Online Sparse System Identification and Signal Reconstruction using Projections onto Weighted ℓ_1 Balls

Yannis Kopsinis, *Member, IEEE,* Konstantinos Slavakis, *Member, IEEE,* and Sergios Theodoridis, *Fellow, IEEE*

Abstract

This paper presents a novel projection-based adaptive algorithm for sparse signal and system identification. The sequentially observed data are used to generate an equivalent sequence of closed convex sets, namely hyperslabs. Each hyperslab is the geometric equivalent of a cost criterion, that quantifies "data mismatch". Sparsity is imposed by the introduction of appropriately designed weighted ℓ_1 balls and the related projection operator is also derived. The algorithm develops around projections onto the sequence of the generated hyperslabs as well as the weighted ℓ_1 balls. The resulting scheme exhibits linear dependence, with respect to the unknown system's order, on the number of multiplications/additions and an $\mathcal{O}(L \log_2 L)$ dependence on sorting operations, where L is the length of the system/signal to be estimated. Numerical results are also given to validate the performance of the proposed method against the LASSO algorithm and two very recently developed adaptive sparse schemes that fuse arguments from the LMS / RLS adaptation mechanisms with those imposed by the lasso rational.

Index Terms

Adaptive filtering, sparsity, projections, compressive sensing.

Y. Kopsinis and S. Theodoridis are with the University of Athens, Department of Informatics and Telecommunications, Ilissia, Athens 15784, Greece. Emails: kopsinis@ieee.org, stheodor@di.uoa.gr. Tel: +30.210.727.5328, Fax: +30.210.727.5337.

K. Slavakis is with the University of Peloponnese, Department of Telecommunications Science and Technology, Karaiskaki St., Tripolis 22100, Greece. Email: slavakis@uop.gr. Tel: +30.2710.37.2204, Fax: +30.2710.37.2160.

I. INTRODUCTION

Sparsity is the key characteristic of systems whose impulse response consists of only a few nonzero coefficients, while the majority of them retain values of negligible size. Similarly, any signal comprising a small number of nonzero samples is also characterized as being a sparse one. The exploitation of sparsity is recently attracting an interest of exponential growth under the Compressed Sensing (CS) framework [1]–[3]. In principle, CS allows the estimation of sparse signals and systems using a smaller number of measurements than that which was previously thought to be necessary. Moreover, identification/reconstruction can be realized with efficient constrained minimization schemes. Indeed, it has been shown that sparsity is favored by ℓ_1 constrained solutions [4], [5].

With only a few recent exceptions, i.e., [6]–[10], the majority of the proposed, so far, CS techniques are appropriate for batch mode operation. In other words, one has to wait until a fixed and predefined number of measurements is available prior to application of CS processing methods, in order to recover the corresponding signal/system estimate. Dynamic online operation for updating and improving estimates, as new measurements become available, is inefficient and impractical when batch processing methods are used. The development of efficient, online adaptive CS techniques is of great importance, especially for cases where the signal or system under consideration is time-varying and/or if the available storage resources are limited.

The basic idea in [6], [7] is to use ℓ_1 *regularization*, i.e., to add to a loss function an extra penalty term expressed by means of a ℓ_1 norm of the unknown system/signal coefficients. Such an approach has been adopted for the classical LMS [6], and for the LS type [7] minimization problems. The resulting recursions for the time update use the current estimate and the information residing in the subgradient of the cost function (due to the non-differentiability of the ℓ_1 norm) to provide the next estimate. Alternatively, a sequence of convex optimisation problems, one per measurement, have to be solved [11].

This paper evolves along a different rationale compared to [6], [7], [11], and introduces a projection-based algorithm for sparse system identification and sparse signal reconstruction. The kick-off point is the set theoretic estimation approach, e.g., [12]. Instead of a single optimum, we search for a set of points that are in *agreement* with the available information, which resides in the training data set (measurements) as well as in the available constraints (the ℓ_1 ball, in this case). To this end, as each new set of measurements is received, a closed convex set is constructed, which defines the region in the solution space that is in "agreement" with the current measurement. In the context of the current paper, the shape of these convex sets is chosen to be a hyperslab. The resulting problem is a convex feasibility task, with an *infinite*

number of convex sets. The fundamental tool of projections onto closed convex sets is used to tackle the problem, following the recent advances on adaptive projection algorithms [13]–[15]. Instead of using the information associated with the subgradient of the ℓ_1 norm, the ℓ_1 constraint is imposed on our solution via the exact projection mapping onto a weighted ℓ_1 ball. The algorithm consists of a sequence of projections onto the generated hyperslabs as well as the weighted ℓ_1 balls. The associated complexity is of order $\mathcal{O}(qL)$ multiplications/additions and $\mathcal{O}(L\log_2 L)$ sorting operations, where L is the length of the system/signal to be identified and q is a user-defined parameter, that controls convergence speed and it defines the number of measurements that are processed, concurrently, at each time instant. The resulting algorithm enjoys a clear geometric interpretation.

Matlab implementations of all the proposed algorithms can be found in
www.di.uoa.gr/~stheodor/adaptivelearning.html

The paper is organized as follows. In Section II the problem under consideration is described and in Section III some definitions and related background are provided. Section IV presents the proposed algorithm. The derivation and discussion of the projection mapping onto the weighted ℓ_1 ball are treated in Section V. The adopted mechanism for weighting the ℓ_1 ball is discussed in Section VI. In Section VII, the convergence properties of the algorithm are derived and discussed. It must be pointed out that this section comprises one of the main contributions of the paper, since the existing, so far, theory cannot cover the problem at hand and has to be extended. In Section VIII, the performance of the proposed algorithmic scheme is evaluated for both, time-invariant and time-varying scenarios. Section IX addresses issues related to the sensitivity of the methods, used in the simulations, to non-ideal parametrization and, finally, the conclusions are provided in Section X. The Appendices offer a more detailed tour to the necessary, for the associated theory, proofs.

II. Problem description

We will denote the set of all integers, non-negative integers, positive integers, and real numbers by $\mathbb{Z}$, $\mathbb{Z}_{\geq 0}$, $\mathbb{Z}_{>0}$, and $\mathbb{R}$, respectively. Given two integers $j_1, j_2 \in \mathbb{Z}$, such that $j_1 \leq j_2$, let $\overline{j_1, j_2} := \{j_1, j_1+1, \ldots, j_2\}$.

The stage of discussion will be the Euclidean space $\mathbb{R}^L$, of dimension $L \in \mathbb{Z}_{>0}$. Its norm will be denoted by $\|\cdot\|$. The superscript symbol $(\cdot)^T$ will stand for vector transposition. The ℓ_1 norm of a vector $\boldsymbol{h} = [h_1, \ldots, h_L]^T \in \mathbb{R}^L$ is defined as the quantity $\|\boldsymbol{h}\|_{\ell_1} := \sum_{i=1}^{L} |h_i|$. The support of a vector $\boldsymbol{h}$ is defined as $\text{supp}(\boldsymbol{h}) := \{i \in \overline{1, L} : \ h_i \neq 0\}$. The ℓ_0-norm of $\boldsymbol{h}$ is defined as the cardinality of its support, i.e., $\|\boldsymbol{h}\|_{\ell_0} := \#\,\text{supp}(\boldsymbol{h})$.

Put in general terms, the problem to solve is to estimate a vector $\boldsymbol{h}_*$, based on measurements that are sequentially generated by the (unknown) linear regression model:

$$y_n = \boldsymbol{x}_n^T \boldsymbol{h}_* + v_n, \quad \forall n \in \mathbb{Z}_{\geq 0}, \tag{1}$$

where the model outputs $(y_n)_{n\in\mathbb{Z}_{\geq 0}} \subset \mathbb{R}$ and the model input vectors $(\boldsymbol{x}_n)_{n\in\mathbb{Z}_{\geq 0}} \subset \mathbb{R}^L$ comprise the measurements and $(v_n)_{n\in\mathbb{Z}_{\geq 0}}$ is the noise process. Furthermore, the unknown vector $\boldsymbol{h}_*$ is S-sparse, meaning that it has S non-zero terms only, with S being small compared to L, i.e., $S := \|\boldsymbol{h}_*\|_{\ell_0} \ll L$.

For a finite number of measurements N, the previous data generation model can be written compactly in the following matrix-vector form,

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{h}_* + \boldsymbol{v}, \tag{2}$$

where the input matrix $\boldsymbol{X} \in \mathbb{R}^{N\times L}$ has as its rows the input measurement vectors, $\boldsymbol{y} := [y_1, y_2, \ldots, y_N]^T$, and $\boldsymbol{v} := [v_1, v_2, \ldots, v_N]^T$.

Depending on the physical quantity that $\boldsymbol{h}_*$ represents, the model in (2) suits to both sparse signal reconstruction and linear sparse system identification:

1) **Sparse signal reconstruction problem:** The aim is to estimate an unknown sparse signal, $\boldsymbol{h}_*$, based on a set of measurements (training data), that are obtained as inner products of the unknown signal with appropriately selected input vectors, $\boldsymbol{x}_n$, according to (1). The elements of the input vectors are often selected to be independent and identically distributed (i.i.d.) random variables following, usually, a zero-mean normal or a Bernoulli distribution [5].
2) **System identification problem:** The unknown sparse system with impulse response $\boldsymbol{h}_*$ is probed with an input signal x_n, $n \in \mathbb{Z}_{\geq 0}$ yielding the output values y_n as the result of convolution of the input signal with the (unknown) impulse response of the system. In agreement to the model of (1), the measurement (input) vector, at time n, is given by $\boldsymbol{x}_n := [x_n, x_{n-1}, \ldots, x_{n-L+1}]^T$. In the matrix-vector formulation, and for a finite number of measurements, the corresponding measurement matrix $\boldsymbol{X}$ is a (partial) Toeplitz one having as entries the elements $\boldsymbol{X}_{i,j} = x_{i+L-j}$, where $i \in \overline{1, N}$ and $j \in \overline{1, L}$. The input signal vector, $\boldsymbol{x}$, usually consists of i.i.d. normally distributed samples. The study of Toeplitz matrices, with respect to their potential to serve as CS measurement matrices, has been recently intensified, e.g., [16], [17], partially due to their importance in sparse channel estimation applications [18].

A batch approach to estimating a sparse $\boldsymbol{h}_*$ based on a limited number of measurements $N < L$, is provided by the Least-Absolute Shrinkage and Selection Operator (LASSO):

$$\boldsymbol{h}_* = \arg\min_{\boldsymbol{h}\in\mathbb{R}^L:\ \|\boldsymbol{h}\|_{\ell_1}\leq\delta} \|\boldsymbol{X}\boldsymbol{h} - \boldsymbol{y}\|^2. \tag{3}$$

In this case, $\boldsymbol{h}_*$ is assumed to be stationary and the total number of measurements, N, needs to be available prior to solution of the LASSO task.

In the current study, we will assume that $\boldsymbol{h}_*$ is not only sparse but it is also allowed to be time-varying. This poses certain distinct differences with regard to the standard compressive sampling scenario. The major objective is no longer to estimate the sparse signal or system, based on a limited number of measurements. The additional requirement, which is often more hard to cope with, is the capability of the estimator to track possible variations of the unknown signal or system. Moreover, this has to take place at an affordable computational complexity, as required by most real time applications, where online adaptive estimation is of interest. Consequently, the batch *sparsity aware* techniques developed under the CS framework, solving LASSO or one of its variants, become unsuitable under time-varying scenarios. The focus now becomes to develop techniques that a) exploit the sparsity b) exhibit fast convergence to error floors that are as close as possible to those obtained by their batch counterparts c) offer good tracking performance and d) have low computational demands in order to meet the stringent time constraints that are imposed by most real time operation scenarios.

III. Online estimation under the sparsity constraint

The objective of online techniques is the generation of a sequence of estimates, $(\boldsymbol{h}_n)_{n\in\mathbb{Z}_{\geq 0}}$, as time, n, evolves, which converge to a value that *"best approximates"*, in some sense, the unknown sparse vector $\boldsymbol{h}_*$. The classical approach to this end is to adopt a loss function and then try to minimize it in a time recursive manner. A more recent approach is to achieve the goal via set theoretic arguments by exploiting the powerful tool of projections.

A. Loss function minimization approach.

A well-known approach to quantify the "best approximation" term is the minimization of a user-defined loss function

$$\forall n \in \mathbb{Z}_{\geq 0}, \forall \boldsymbol{h} \in \mathbb{R}^L, \quad \Theta_n(\boldsymbol{h}) := \mathcal{L}_r^{(n)}(\boldsymbol{h}) + \gamma_n \mathcal{L}_s^{(n)}(\boldsymbol{h}), \tag{4}$$

where $\mathcal{L}_r^{(n)}$ is computed over the training (observed) data set and accounts for the data mismatch, between measured and desired responses, and $\mathcal{L}_s^{(n)}$ accounts for the "size" of the solution, and in the current context is the term that imposes *sparsity*. The sequence of user-defined parameters $(\gamma_n)_{n\in\mathbb{Z}_{\geq 0}}$ accounts for the relative contribution of $\mathcal{L}_r^{(n)}, \mathcal{L}_s^{(n)}$ to the cost in (4). Usually, both functions $\mathcal{L}_r^{(n)}, \mathcal{L}_s^{(n)}$ are chosen to be convex, due to the powerful tools offered by the convex analysis theory.

For example, the study in [6] chooses $\mathcal{L}_r^{(n)}(\boldsymbol{h}) := \frac{1}{2}|y_n - \boldsymbol{h}^T\boldsymbol{x}_n|^2$, where $\mathcal{L}_s^{(n)}(\boldsymbol{h}) := \|\boldsymbol{h}\|_{\ell_1}$, $\boldsymbol{h} \in \mathbb{R}^L$, in order to obtain the ZA-LMS algorithm. The RZA-LMS scheme is obtained in [6] when setting $\mathcal{L}_s^{(n)}(\boldsymbol{h}) := \sum_{i=1}^{L} \log(1 + \frac{|h_i|}{\eta})$, $\boldsymbol{h} \in \mathbb{R}^L$, while keeping the same $\mathcal{L}_r^{(n)}$. In [7], the sum Least Squares with a forgetting factor β is used in place of $\mathcal{L}_r^{(n)}$ and the ℓ_1 norm in $\mathcal{L}_s^{(n)}$.

B. Set theoretic approach.

In this paper, a different path is followed. Instead of attempting to minimize, recursively, a cost function that is defined over the entire observations' set, our goal becomes to find a *set* of solutions that is in *agreement* with the available observations as well as the constraints. To this end, at each time instant, n, we require our estimate $\boldsymbol{h}_n$ to lie within an appropriately defined closed convex set, which is a subset of our solutions space and it is also known as *property set*. Any point that lies within this set is said to be in agreement with the current measurement pair $(\boldsymbol{x}_n, y_n)$. The "shape" of the property set is dictated by a "local" loss function, which is assumed to be convex. In the context of the current paper, we have adopted property sets that are defined by the following criterion

$$S_n[\epsilon] := \{\boldsymbol{h} \in \mathbb{R}^L : \ |\boldsymbol{h}^T\boldsymbol{x}_n - y_n| \leq \epsilon\}, \ n \in \mathbb{Z}_{\geq 0}, \tag{5}$$

for some user-defined tolerance $\epsilon \geq 0$. Such criteria have extensively been used in the context of robust statistics cost functions. Eq. (5) defines a *hyperslab*, which is indeed a closed convex set. Any point that lies in the hyperslab generated at time n is in agreement with the corresponding measurement at the specific time instance. The parameter ϵ determines the width of the hyperslabs. Fig. 1 shows two hyperslabs defined at two successive instants, namely, n and $n-1$.

Having associated each measurement pair with a hyperslab, our goal, now, becomes to find a point in $\mathbb{R}^L$ that lies in the intersection of these hyperslabs, provided that this is nonempty. We will come back to this point when discussing the convergence issues of our algorithm. For a recent review of this algorithmic family the reader may consult [19].

To exploit sparsity, we adopt the notion of the weighted ℓ_1 norm. Given a vector $\boldsymbol{w}_n \in \mathbb{R}^L$ with positive components, i.e., $w_{n,i} > 0$, $\forall i \in \overline{1, L}$, the *weighted* ℓ_1 *ball of radius* $\delta > 0$ is defined as [20]

$$B_{\ell_1}[\boldsymbol{w}_n, \delta] := \{\boldsymbol{h} \in \mathbb{R}^L : \ \sum_{i=1}^{L} w_{n,i}|h_i| \leq \delta\}. \tag{6}$$

For more flexibility, we let the weight vector depend on the time instant n, hence the notation $\boldsymbol{w}_n$ has been adopted. We will see later on that such a strategy speeds up convergence and decreases the misadjustment error of the algorithm. The well-known unweighted ℓ_1 *ball* is nothing but $B_{\ell_1}[\mathbf{1}, \delta]$, where $\mathbf{1} \in \mathbb{R}^L$ is a

vector with 1s in all of its components. Note that all the points that lie inside a weighted ℓ_1 norm form a closed convex set.

Having defined the weighted ℓ_1 ball, which is the sparsity related constraint, our task now is to search for a point $\boldsymbol{h}$ in $\mathbb{R}^L$ that lies in the intersection of the hyperslabs as well as the weighted ℓ_1 balls, i.e., for some $z_0 \in \mathbb{Z}_{\geq 0}$,

$$\text{find an } \boldsymbol{h} \in \bigcap_{n \geq z_0} \left(S_n[\epsilon] \cap B_{\ell_1}[\boldsymbol{w}_n, \delta] \right). \tag{7}$$

As it will become clear later on, when discussing the convergence issues of the algorithm, the existence of z_0 in (7) allows for a finite number of property sets not to share intersection with the rest.

IV. Proposed algorithmic framework

The solution to the problem of finding a point lying in the intersection of a number of closed convex sets traces its origins back in the classical POCS theory [21]–[24]. However, in contrast to POCS where only a finite number of sets is involved, our methodology deals with an *infinite number* of sets, which was originally proposed in [13] and extended [14]. The basic idea is very elegant: Keep projecting, according to an appropriate rule, on the involved convex sets; then this sequence of projections will, finally, take you to a point in their intersection. Hence, for our problem, metric projection mapping operators for, both, the hyperslabs as well as the weighted ℓ_1 balls have to be used. Projection operators for hyperslabs are already known and widely used, e.g., [19], [25]. The metric projection mapping onto a weighted ℓ_1 norm will be derived here, and it was presented for a first time, to the best of our knowledge, in [10].

Each time instant, n, a new pair of training data $(\boldsymbol{x}_n, y_n)$ becomes available, and a corresponding hyperslab is formed according to (5). This is used to update the currently available estimate $\boldsymbol{h}_n$. However, in order to speed up convergence, the update mechanism can also involve previously defined hyperslabs; for example, the hyperslabs formed at time instants $\overline{n-q+1, n}$, for some $q \in \mathbb{Z}_{>0}$. Then, in order to obtain $\boldsymbol{h}_{n+1}$, an iteration scheme consisting of three basic steps is adopted: a) the current estimate $\boldsymbol{h}_n$ is projected onto each one of the q hyperslabs, b) these projections are in turn combined as a weighted sum and c) the result of the previous step is subsequently projected onto the weighted ℓ_1 ball. This is according to the concepts introduced in [13] and followed in [14], [25], [26]. Schematically, the previous procedure is illustrated in Fig. 1, for the case of $q = 2$.

In detail, the algorithm is mathematically described as follows:

Algorithm. Let $q \in \mathbb{Z}_{>0}$, and define the following sliding window on the time axis, of size at most q (to account for the initial period where $n < q - 1$), in order to indicate the hyperslabs to be processed

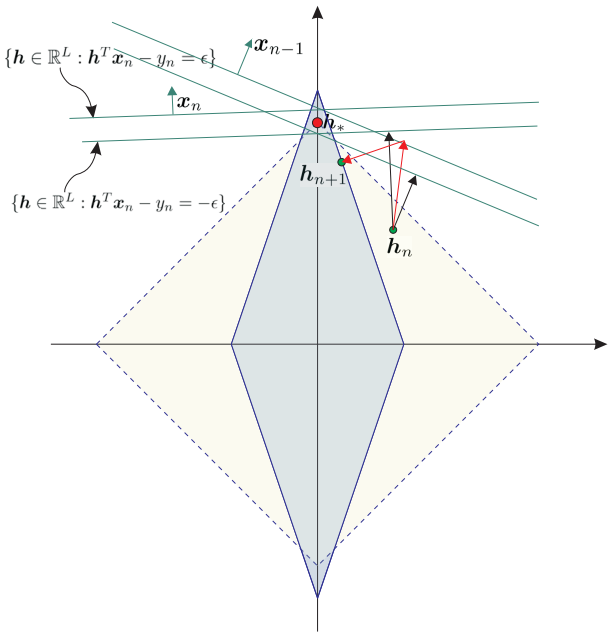

Fig. 1. The ℓ_1 ball is shown with dotted lines. At time n, the estimate $\boldsymbol{h}_n$ is available. For $q = 2$, two hyperslabs are involved in the update recursion, those associated with time instants n and $n-1$. The new update, $\boldsymbol{h}_{n+1}$, results by first projecting $\boldsymbol{h}_n$ onto the hyperslabs, then combining the resulting projections and finally projecting onto the weighted ℓ_1 norm, that is defined at time n and which is drawn using the full line.

at each time instant:

$$\mathcal{J}_n := \overline{\max\{0, n-q+1\}, n}, \quad \forall n \in \mathbb{Z}_{\geq 0},$$

i.e., the hyperslabs $\{S_j[\epsilon]\}_{j\in\mathcal{J}_n}$ are to be concurrently processed at time instant n.

Given the current estimate $\boldsymbol{h}_n \in \mathbb{R}^L$ of the sparse system under study, we identify the *active* hyperslabs, among $\{S_j[\epsilon]\}_{j\in\mathcal{J}_n}$, which provide new "information" to our learning algorithm, i.e., those hyperslabs $S_j[\epsilon]$ for which $\boldsymbol{h}_n \notin S_j[\epsilon]$:

$$\mathcal{I}_n := \{j \in \mathcal{J}_n : \ \boldsymbol{h}_n \notin S_j[\epsilon]\}. \tag{8}$$

The way to identify $\mathcal{I}_n$ is easy; notice that $\boldsymbol{h}_n \notin S_j[\epsilon] \Leftrightarrow \boldsymbol{h}_n \neq P_{S_j[\epsilon]}(\boldsymbol{h}_n)$, where $P_{S_j[\epsilon]}$ stands for the metric projection mapping onto $S_j[\epsilon]$.

Having the knowledge of $\mathcal{I}_n$, we define the set of weights $\{\omega_j^{(n)}\}_{j\in\mathcal{I}_n} \subset (0, 1]$ such that $\sum_{j\in\mathcal{I}_n} \omega_j^{(n)} = 1$. Each $\omega_j^{(n)}$ quantifies the contribution of the j-th hyperslab into a weighted combination of all the hyperslabs indicated by $\mathcal{I}_n$.

Given an arbitrary initial point $\boldsymbol{h}_0 \in \mathbb{R}^L$, the following recursion generates the sequence of estimates

$(\boldsymbol{h}_n)_{n\in\mathbb{Z}_{\geq 0}}$; $\forall n \in \mathbb{Z}_{\geq 0}$,

$$\boldsymbol{h}_{n+1} := P_{B_{\ell_1}[\boldsymbol{w}_n,\delta]}\left(\boldsymbol{h}_n + \mu_n\left(\sum_{j\in\mathcal{I}_n}\omega_j^{(n)} P_{S_j[\epsilon]}(\boldsymbol{h}_n) - \boldsymbol{h}_n\right)\right), \tag{9}$$

where $P_{S_j[\epsilon]}$ and $P_{B_{\ell_1}[\boldsymbol{w}_n,\delta]}$ denote the metric projection mappings onto the hyperslab, defined by the j-th data pair, and onto the (currently available) weighted ℓ_1 ball, respectively. As it will be shown in the analysis of the algorithm in Appendix B, in order to guarantee convergence, the extrapolation parameter μ_n takes values within the interval $(0, 2\mathcal{M}_n)$, where $\mathcal{M}_n$ is computed by

$$\mathcal{M}_n := \begin{cases} \frac{\sum_{j\in\mathcal{I}_n}\omega_j^{(n)}\|\boldsymbol{h}_n - P_{S_j[\epsilon]}(\boldsymbol{h}_n)\|^2}{\|\sum_{j\in\mathcal{I}_n}\omega_j^{(n)}(\boldsymbol{h}_n - P_{S_j[\epsilon]}(\boldsymbol{h}_n))\|^2}, & \text{if } \sum_{j\in\mathcal{I}_n}\omega_j^{(n)}(\boldsymbol{h}_n - P_{S_j[\epsilon]}(\boldsymbol{h}_n)) \neq \boldsymbol{0}, \\ 1, & \text{otherwise.} \end{cases} \tag{10}$$

Notice that the convexity of the function $\|\cdot\|^2$ implies that $\mathcal{M}_n \geq 1$, $\forall n \in \mathbb{Z}_{\geq 0}$. To avoid any ambiguities in (9) and (10), in cases where $\mathcal{I}_n = \emptyset$, we let $\sum_{j\in\emptyset}\omega_j^{(n)}(\boldsymbol{h}_n - P_{S_j[\epsilon]}(\boldsymbol{h}_n)) := \boldsymbol{0}$. ■

It is interesting to point out that the algorithm is compactly encoded into a single equation! Also, note that projection onto the hyperslabs $\{S_j[\epsilon]\}_{j\in\mathcal{I}_n}$ can take place concurrently and this can be exploited if computations are carried out in a parallel processing environment. Moreover, q can be left to vary from iteration to iteration. The dependence of the algorithm's performance on q will be discussed in Section VIII.

It turns out that the projection mappings involved in (9) and (10) have computationally simple forms and are given in (12) and Section V-B. The algorithm amounts to a computational load of order, at most, $\mathcal{O}(qL)$ multiplications/additions and $\mathcal{O}(L\log_2 L)$ sorting operations. The dependence on q is relaxed in a parallel processing environment.

Having disclosed the algorithmic scheme for the update of our estimate at each iteration step, as measurements are received sequentially, there are a number of issues, yet, to be resolved. First, the involved projection mappings have to be explicitly provided/derived. Second, a strategy for the selection of the weights in the weighted ℓ_1 norm need to be decided. Third, the convergence of the algorithm has to be established. Although the algorithm stands on the shoulders of the theory developed in previously published papers, e.g., [13], [14], [27], the developed, so far, theory is not enough to cover the current algorithm. Since we do not use the ℓ_1 norm, but its weighted version, the projection mapping $P_{B_{\ell_1}[\boldsymbol{w}_n,\delta]}$ in (9) is *time-varying* and it also depends on the obtained estimates. Convergence has to be proved for such a scenario and this is established in Appendix B.

V. Projections onto Closed Convex Sets

A subset C of $\mathbb{R}^L$ will be called convex if every line segment $\{\lambda \boldsymbol{h} + (1-\lambda)\boldsymbol{h}' : \lambda \in [0,1]\}$, with endpoints any $\boldsymbol{h}, \boldsymbol{h}' \in C$, lies in C.

Given any set $C \subset \mathbb{R}^L$, define the *(metric) distance function* $d(\cdot, C) : \mathbb{R}^L \to \mathbb{R}$ *to* C as follows: $\forall \boldsymbol{x} \in \mathbb{R}^L$, $d(\boldsymbol{x}, C) := \inf\{\|\boldsymbol{x} - \boldsymbol{f}\| : \ \boldsymbol{f} \in C\}$. If we assume now that C is closed and convex, then the *(metric) projection onto* C is defined as the mapping $P_C : \mathbb{R}^L \to C$ which maps to an $\boldsymbol{x} \in \mathbb{R}^L$ the unique $P_C(\boldsymbol{x}) \in C$ such that $\|\boldsymbol{x} - P_C(\boldsymbol{x})\| = d(\boldsymbol{x}, C)$. A well-known property of the metric projection mapping P_C onto a closed convex set C, which will be used in the sequel, is the following [23], [24]:

$$\forall \boldsymbol{x} \in \mathbb{R}^L, \forall \boldsymbol{f} \in C, \quad \|\boldsymbol{x} - P_C(\boldsymbol{x})\|^2 \leq \|\boldsymbol{x} - \boldsymbol{f}\|^2 - \|P_C(\boldsymbol{x}) - \boldsymbol{f}\|^2. \tag{11}$$

A. Projecting onto a hyperslab.

The metric projection operator $P_{S_n[\epsilon]}$ onto the hyperslab (5) takes the following simple analytic form [23], [24]:

$$\forall \boldsymbol{h} \in \mathbb{R}^L, \quad P_{S_n[\epsilon]}(\boldsymbol{h}) = \boldsymbol{h} + \begin{cases} \frac{y_n - \epsilon - \boldsymbol{h}^T \boldsymbol{x}_n}{\|\boldsymbol{x}_n\|^2} \boldsymbol{x}_n, & \text{if } y_n - \epsilon > \boldsymbol{h}^T \boldsymbol{x}_n, \\ \boldsymbol{0}, & \text{if } |\boldsymbol{h}^T \boldsymbol{x}_n - y_n| \leq \epsilon, \\ \frac{y_n + \epsilon - \boldsymbol{h}^T \boldsymbol{x}_n}{\|\boldsymbol{x}_n\|^2} \boldsymbol{x}_n, & \text{if } y_n + \epsilon < \boldsymbol{h}^T \boldsymbol{x}_n. \end{cases} \tag{12}$$

B. Projecting onto the weighted ℓ_1 ball.

The following theorem computes, in a finite number of steps, the exact projection of a point onto a weighted ℓ_1 ball. The result generalizes the projection mapping computed for the case of the classical unweighted ℓ_1 ball in [28], [29]. In words, the projection mapping exploits the part of the weighted ℓ_1 ball that lies in the non-negative hyperorthant of the associated space. This is because the projection of a point onto the weighted ℓ_1 ball lies always in the same hyperorthant as the point itself. Hence, one may always choose to map the problem on the non-negative hyperorthant, work there, and then return to the original hyperorthant of the space, where the point lies. The part of the weighted ℓ_1 norm, that lies in the non-negative hyperorthant, can be seen as the intersection of a closed halfspace and the non-negative hyperorthant, see Fig. 2. It turns out that if the projection of a point on this specific halfspace has all its components positive, e.g., point $\boldsymbol{x}_1$ in Fig. 2, then the projection on the halfspace and the projection of the point on the weighted ℓ_1 ball coincide. If, however, some of the components of the projection onto the halfspace are non-positive, e.g., point $\boldsymbol{x}_2$ in Fig. 2, the corresponding dimensions are ignored and a projection takes place in the resulting lower dimensional space. It turns out that this projection coincides

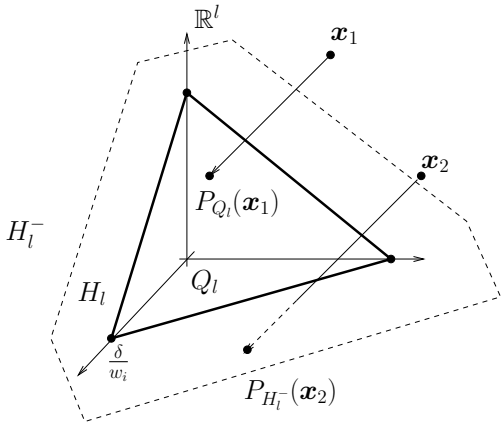

Fig. 2. This figure illustrates the geometry of the weighted ℓ_1 ball $B_{\ell_1}[\boldsymbol{w}, \delta]$, and more specifically its intersection with the non-negative hyperorthant of $\mathbb{R}^l$. The reason for studying only the non-negative hyperorthant is justified by Lemma 1 of appendix A. Two points $\boldsymbol{x}_1, \boldsymbol{x}_2$ of $\mathbb{R}^l$ are considered to demonstrate the concepts introduced in the manuscript. Notice that $P_{H_l^-}(\boldsymbol{x}_1) > \boldsymbol{0}$, which implies by Lemma 2.1 of appendix A that $P_{Q_l}(\boldsymbol{x}_1) = P_{H_l^-}(\boldsymbol{x}_1)$. Point $\boldsymbol{x}_2$ demonstrates the case where some of the components of $P_{H_l^-}(\boldsymbol{x}_2)$ take non-positive values. These components are ignored and projection takes place in a lower dimensional space.

with the projection of the point on the weighted ℓ_1 ball. The previous procedure is formally summarized next.

Theorem 1. Given an $\boldsymbol{h} \in \mathbb{R}^L \setminus B_{\ell_1}[\boldsymbol{w}_n, \delta]$, the following recursion computes, in a finite number of steps (at most L), the projection of $\boldsymbol{h}$ onto the ball $B_{\ell_1}[\boldsymbol{w}_n, \delta]$, i.e., the (unique) vector $P_{B_{\ell_1}[\boldsymbol{w}_n,\delta]}(\boldsymbol{h}) \in \mathbb{R}^L$. The case of $\boldsymbol{h} \in B_{\ell_1}[\boldsymbol{w}_n, \delta]$ is trivial, since $P_{B_{\ell_1}[\boldsymbol{w}_n,\delta]}(\boldsymbol{h}) = \boldsymbol{h}$.

1) Form the vector $[|h_1|/w_{n,1}, \ldots, |h_L|/w_{n,L}]^T \in \mathbb{R}^L$.
2) Sort the previous vector in a non-ascending order (this takes $\mathcal{O}(L \log_2 L)$ computations), so that $[|h_{\tau(1)}|/w_{n,\tau(1)}, \ldots, |h_{\tau(L)}|/w_{n,\tau(L)}]^T$, with $|h_{\tau(1)}|/w_{n,\tau(1)} \geq \cdots \geq |h_{\tau(L)}|/w_{n,\tau(L)}$, is obtained. The notation τ stands for the permutation, which is implicitly defined by the sorting algorithm. Keep in memory the inverse τ^{-1} which moves the sorted elements back to the original positions.
3) Let $r_1 := L$.
4) Let $l = 1$. While $l \leq L$, do the following.
   a) Let $\lambda_* := l$.
   b) Find the maximum j_* among those $j \in \overline{1, r_l}$ such that $\frac{|h_{\tau(j)}|}{w_{n,\tau(j)}} > \frac{\sum_{i=1}^{r_l} w_{n,\tau(i)}|h_{\tau(i)}| - \delta}{\sum_{i=1}^{r_l} w_{n,\tau(i)}^2}$.
   c) If $j_* = r_l$ then break the loop.
   d) Otherwise, set $r_{l+1} := j_*$.

e) Increase l by 1, and go back to Step 4a.

5) Form the vector $\hat{\boldsymbol{p}} \in \mathbb{R}^{r_{\lambda_*}}$ whose j-th component is given by $\hat{p}_j := |h_{\tau(j)}| - \frac{\sum_{i=1}^{r_{\lambda_*}} w_{n,\tau(i)}|h_{\tau(i)}| - \delta}{\sum_{i=1}^{r_{\lambda_*}} w_{n,\tau(i)}^2} w_{n,\tau(j)}$.
6) Use the inverse mapping τ^{-1}, met in step 2, to insert the number $\hat{p}_j$ into the $\tau^{-1}(j)$ position of the L-dimensional vector $\boldsymbol{p}$, i.e., $p_{\tau^{-1}(j)} := \hat{p}_j$, $\forall j \in \overline{1, r_{\lambda_*}}$, and fill in the remaining $L - r_{\lambda_*}$ positions of $\boldsymbol{p}$ with zeros.
7) The desired projection is $P_{B_{\ell_1}[\boldsymbol{w}_n, \delta]}(\boldsymbol{h}) = [\mathrm{sgn}(h_1)p_1, \ldots, \mathrm{sgn}(h_L)p_L]^T \in \mathbb{R}^L$, where the symbol $\mathrm{sgn}(\cdot)$ stands for the sign of a real number.

Proof: The proof is given in Appendix A. It follows a geometric approach, instead of the Lagrange multipliers methodology, which was followed in [28] for the case of the unweighted ℓ_1 norm. ■

VI. Weighting the ℓ_1 Ball

Motivated by the strategy adopted in [20], the sequence of weights $(\boldsymbol{w}_n)_{n \in \mathbb{Z}_{\geq 0}}$ is designed as follows; let the i-th component of the vector $\boldsymbol{w}_n$ be given by

$$w_{n,i} := \frac{1}{|h_{n,i}| + \epsilon'_n}, \ \forall i \in \overline{1, L}, \ \forall n \in \mathbb{Z}_{\geq 0}, \tag{13}$$

where $(\epsilon'_n)_{n \in \mathbb{Z}_{\geq 0}}$ is a sequence of small positive parameters, which are used in order to avoid division by zero. An illustration of the induced geometry can be seen in Fig. 1. A way to design the parameters $(\epsilon'_n)_{n \in \mathbb{Z}_{\geq 0}}$ will be given in the next section. The corresponding algorithm will be referred to as the Adaptive Projection-based Algorithm using Weighted ℓ_1 balls (APWL1). The unweighted case, i.e., when $\boldsymbol{w}_n := \boldsymbol{1}$, $\forall n \in \mathbb{Z}_{\geq 0}$, will be also considered and is denoted as APL1.

Remark 1. The radius δ of the ℓ_1 norm, on which we project, depends on whether the unweighted or the weighted version is adopted. In the unweighted ℓ_1 norm case, the optimum value of the radius is apparently $\delta := \|\boldsymbol{h}_*\|_{\ell_1}$. However, in the weighted case, δ is set equal to $S = \|\boldsymbol{h}_*\|_{\ell_0}$. The reason for this is the following.

Consider the desirable situation where our sequence of estimates $(\boldsymbol{h}_n)_{n \in \mathbb{Z}_{\geq 0}}$ converges to $\boldsymbol{h}_*$, i.e., $\lim_{n \to \infty} \boldsymbol{h}_n = \boldsymbol{h}_*$. Moreover, let $\epsilon'_n \geq \epsilon' > 0$, $\forall n \in \mathbb{Z}_{\geq 0}$, where ϵ' is a user-defined parameter. Then, $\sum_{i=1}^L w_{n,i}|h_{n,i}| \leq \sum_{i=1}^L \frac{|h_{n,i}|}{|h_{n,i}| + \epsilon'}$, $\forall n \in \mathbb{Z}_{\geq 0}$, and thus,

$$\begin{aligned} \limsup_{n \to \infty} \sum_{i=1}^L w_{n,i}|h_{n,i}| &\leq \limsup_{n \to \infty} \sum_{i=1}^L \frac{|h_{n,i}|}{|h_{n,i}| + \epsilon'} = \lim_{n \to \infty} \sum_{i=1}^L \frac{|h_{n,i}|}{|h_{n,i}| + \epsilon'} \\ &= \sum_{i \in \mathrm{supp}(\boldsymbol{h}_*)} \frac{|h_{*,i}|}{|h_{*,i}| + \epsilon'} + \sum_{i \notin \mathrm{supp}(\boldsymbol{h}_*)} \frac{|h_{*,i}|}{|h_{*,i}| + \epsilon'} < \sum_{i \in \mathrm{supp}(\boldsymbol{h}_*)} \frac{|h_{*,i}|}{|h_{*,i}|} = \|\boldsymbol{h}_*\|_{\ell_0}. \end{aligned}$$

The previous strict inequality and the definition of $\limsup$ suggest that there exists an $m_1 \in \mathbb{Z}_{\geq 0}$ such that $\forall n \geq m_1$ we have $\sum_{i=1}^{L} w_{n,i}|h_{n,i}| \leq \|\boldsymbol{h}_*\|_{\ell_0}$. In other words, we obtain that $\forall n \geq m_1$, $\boldsymbol{h}_n \in B_{\ell_1}[\boldsymbol{w}_n, \|\boldsymbol{h}_*\|_{\ell_0}]$. Hence, a natural choice for δ in the design of the constraint set $B_{\ell_1}[\boldsymbol{w}_n, \delta]$ is $\|\boldsymbol{h}_*\|_{\ell_0}$. At least, such a choice is justified $\forall n \geq m_1$, since it becomes a necessary condition for having $(\boldsymbol{h}_n)_{n\in\mathbb{Z}_{\geq 0}}$ to converge to the desirable $\boldsymbol{h}_*$.

VII. Convergence Properties of the Algorithm

It can be shown that, under certain assumptions, the previous algorithm produces a sequence of estimates $(\boldsymbol{h}_n)_{n\in\mathbb{Z}_{\geq 0}}$, which converges to a point located arbitrarily close to the intersection as in (7). The convergence of the algorithm is guaranteed even if a finite number of closed convex sets do not share any nonempty intersection with the rest of the convex constraints in (7). This is important, since it allows for a finite number of data outliers not to disturb the convergence of the algorithm.

Assumptions.

1) Define $\forall n \in \mathbb{Z}_{\geq 0}$, $\Omega_n := B_{\ell_1}[\boldsymbol{w}_n, \delta] \cap \Big(\bigcap_{j\in\mathcal{I}_n} S_j[\epsilon]\Big)$, i.e., the set Ω_n is defined as the intersection of the weighted ℓ_1 ball and the hyperslabs that are considered at time n. Assume that there exists a $z_0 \in \mathbb{Z}_{\geq 0}$ such that $\Omega := \bigcap_{n\geq z_0} \Omega_n \neq \emptyset$. That is, with the exception of a finite number of Ω_ns, the rest of them have a nonempty intersection. This is an assumption that is often adopted in convex feasibility problems [23].
2) Choose a sufficiently small $\epsilon'' > 0$, and let $\forall n \in \mathbb{Z}_{\geq 0}$, $\frac{\mu_n}{\mathcal{M}_n} \in [\epsilon'', 2 - \epsilon'']$.
3) The interior of Ω is nonempty, i.e., $\mathrm{int}(\Omega) \neq \emptyset$. For the definition of $\mathrm{int}(\cdot)$ see Fact 2 in Appendix B.
4) Assume that $\breve{\omega} := \inf\{\omega_j^{(n)} : \ j \in \mathcal{I}_n, n \in \mathbb{Z}_{\geq 0}\} > 0$. In words, none of the weights, used to combine the projections onto the hyperslabs, will fade away as time n advances.

Theorem 2 (Convergence analysis of the Algorithm)**.** Under the previously adopted assumptions, the following properties can be established.

1) Every update takes us closer to the intersection Ω. In other words, the convergence is monotonic, that is, $\forall n \geq z_0$, $d(\boldsymbol{h}_{n+1}, \Omega) \leq d(\boldsymbol{h}_n, \Omega)$.
2) Asymptotically, the distance of the obtained estimates from the respective hyperslabs tends to zero. That is, $\lim_{n\to\infty} \max\{d(\boldsymbol{h}_n, S_j[\epsilon]) : \ j \in \mathcal{J}_n\} = 0$.
3) Similarly, the distance of the obtained estimates from the respective weighted ℓ_1 balls tends asymptotically to zero. That is, $\lim_{n\to\infty} d(\boldsymbol{h}_n, B_{\ell_1}[\boldsymbol{w}_n, \delta]) = 0$.

4) Finally, there exists an $\tilde{\boldsymbol{h}}_* \in \mathbb{R}^L$ such that the sequence of estimates $(\boldsymbol{h}_n)_{n\in\mathbb{Z}_{\geq 0}}$ converges to, i.e., $\lim_{n\to\infty} \boldsymbol{h}_n = \tilde{\boldsymbol{h}}_*$, and that

$$\tilde{\boldsymbol{h}}_* \in \left(\overline{\liminf_{n\to\infty} B_{\ell_1}[\boldsymbol{w}_n, \delta]}\right) \cap \left(\overline{\liminf_{n\to\infty} S_n[\epsilon]}\right).$$

Here, $\liminf_{n\to\infty} C_n := \bigcup_{n\geq 0} \bigcap_{m\geq n} C_m$, for any sequence $(C_n)_{n\in\mathbb{Z}_{\geq 0}} \subset \mathbb{R}^L$, and the overline denotes the closure of a set. In other words, the algorithm converges to a point that lies arbitrarily close to an intersection of the involved property sets.

Proof: The proof of these results, several auxiliary concepts, as well as details on which assumptions are activated, in order to prove each result, can be found in Appendix B. ■

Remark 2. Regarding Assumption 3, the condition $\operatorname{int} \bigcap_{n\in\mathbb{Z}_{\geq 0}} B_{\ell_1}[\boldsymbol{w}_n, \delta] \neq \emptyset$ can be easily satisfied. Choose any sufficiently small $\epsilon' > 0$ such that $\epsilon'_n \geq \epsilon'$, $\forall n \in \mathbb{Z}_{\geq 0}$, in (13). Let any $\boldsymbol{h} \in B(\boldsymbol{0}, \frac{\delta\epsilon'}{L})$. Then, clearly, $\forall i \in \overline{1, L}$, $|h_i| \leq \frac{\delta\epsilon'}{L}$. Notice, now, that $\forall n \in \mathbb{Z}_{\geq 0}$, $\sum_{i=1}^L w_{n,i}|h_i| = \sum_{i=1}^L \frac{|h_i|}{|h_{n,i}|+\epsilon'_n} \leq \sum_{i=1}^L \frac{|h_i|}{\epsilon'} \leq \sum_{i=1}^L \frac{\delta\epsilon'}{L\epsilon'} = \delta$. In other words, $B(\boldsymbol{0}, \frac{\delta\epsilon'}{L}) \subset B_{\ell_1}[\boldsymbol{w}_n, \delta]$, $\forall n \in \mathbb{Z}_{\geq 0}$, which implies that $B(\boldsymbol{0}, \frac{\delta\epsilon'}{L}) \subset \bigcap_{n\in\mathbb{Z}_{\geq 0}} B_{\ell_1}[\boldsymbol{w}_n, \delta]$. That is, $\boldsymbol{0} \in \operatorname{int} \bigcap_{n\in\mathbb{Z}_{\geq 0}} B_{\ell_1}[\boldsymbol{w}_n, \delta] \neq \emptyset$.

VIII. Performance evaluation

In this section, the performance of the proposed algorithms is evaluated against both time-invariant and time-varying signals and systems. It is also compared to a number of other online algorithms such as the Zero-Attracting LMS (ZA-LMS) [6] and the Reweighted ZA-LMS (RZA-LMS) [6]. Moreover, adaptive implementations of the LASSO algorithm are considered such as the Time-Weighted LASSO (TWL) as well as the Time and Norm Weighted LASSO (TNWL) [11]. Note that the performance of TWL is the same as that of the recursive LASSO (RLASSO) [7] so the performance curves of the latter algorithm are not shown. Details on complexity issues follow when appropriate. In the cases where the system under consideration is not time-varying then the performance of the LASSO and the weighted LASSO [30], solved with batch methods [31], [32], are given instead of the TWL and TNWL. All the performance curves are the result from ensemble averaging of 100 independent runs. Moreover, for all the projection based algorithms tested, in all simulation examples, μ_n was set equal to $\mathcal{M}_n$ and the hyperslabs parameter ϵ was set equal to $1.3 \times \sigma$, with σ being the noise standard deviation. Even though such a choice may not be necessarily optimal, the proposed algorithms turn out to be relatively insensitive to the values of these parameters. Finally, $\omega_j^{(n)}$ of (9) are set equal to $1/q$, $\forall j \in \mathcal{I}_n$, $\forall n \in \mathbb{Z}_{\geq 0}$.

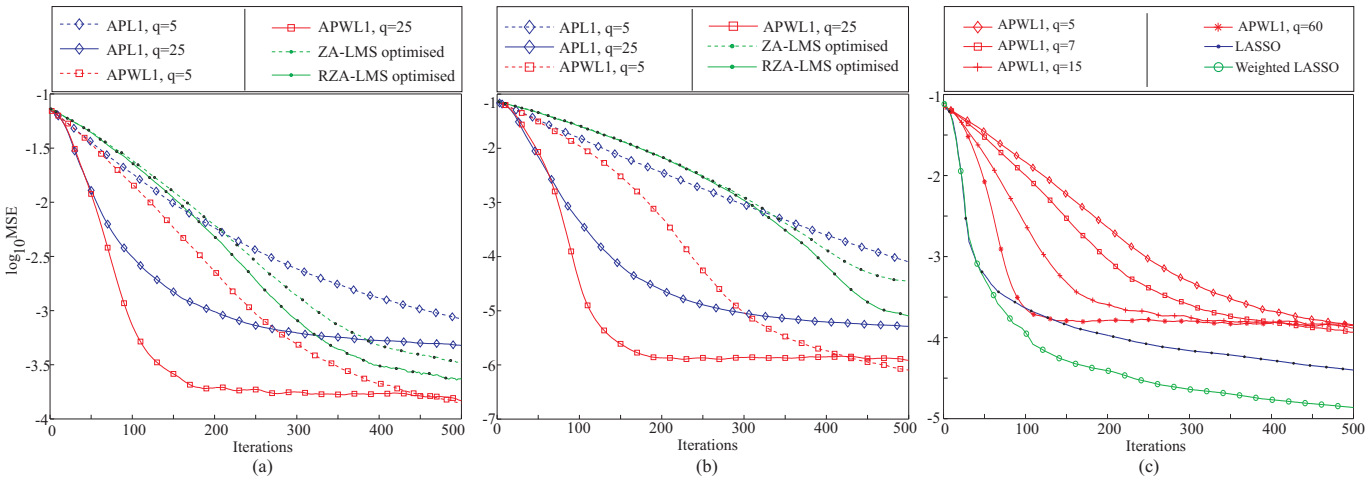

Fig. 3. Sparse system identification example with $L = 100$ and $S = 5$. (a) and (b) shows the performance comparison of the proposed techniques with the LMS-based methods for high and low noise respectively. (c) shows the effect of different q values in comparison to batch LASSO-type algorithms.

A. *Time-invariant case.*

In this simulation example, a time-invariant system having $L = 100$ coefficients is used. The system is sparse with $S = 5$, i.e., it has only five nonzero coefficients, which are placed in arbitrary positions. The input signal $\boldsymbol{x}$ is formed with entries drawn from a zero-mean normal distribution with variance 1.

In Figs. 3(a) and 3(b) the performance of the new algorithm is compared with that obtained by the LMS-based methods, in different noise levels. The noise variance was set equal to two different values, i.e., $\sigma_n^2 = 0.1$ and $\sigma_n^2 = 0.001$ corresponding to SNR values of approximately -3dB and 17dB, respectively. Two different values of the parameter q have been considered, namely 5 an 25. Moreover, with respect to the ZA-LMS and the RZA-LMS, the "optimized" tag indicates that the free parameters μ and ρ were optimized, in order to give the best performance at the 450th iteration. A different parameter setup could lead to faster convergence of both LMS-based methods, albeit at the expense of higher error-floors. In Fig. 3(a) we observe that APWL1 exhibits the best performance both with respect to convergence speed as well as steady-state error floor. In fact, the larger the value of q is the faster the convergence becomes. However, when the unweighted ℓ_1 ball is used (APL1), the method assumes relatively high error-floors, worse than both the LMS-based methods. It must be pointed out that using values of q less than 3, pushes the performance of APWL1 close to that of LMS as expected.

In all the cases, unless the contrary is explicitly stated, the adopted values for δ were: $\delta := \|\boldsymbol{h}_*\|_{\ell_1}$ and $\delta := S$ for the APL1 and the APWL1, respectively. The sensitivity of these methods, on using different

values of δ, will be discussed in section IX-A. Moreover, the adaptation strategy of ϵ'_n in (13) was decided upon the following observation. A very small ϵ'_n, in the range of $[0.001, 0.01]$, leads to low error-floors but the convergence speed is compromised. On the other hand, when ϵ'_n is relatively large, e.g., $\epsilon'_n \geq 0.1$, then fast convergence speed is favored at the expense of a higher steady state error floor. In order to tackle this issue efficiently, ϵ'_n can start with a high value and then getting gradually smaller. Although other scenarios may be possible, in all the time invariant examples, we have chosen: $\epsilon'_n := \epsilon' + \frac{0.2}{n+1}$, $\forall n \in \mathbb{Z}_{\geq 0}$, where ϵ' is a user-defined positive constant which can assume arbitrarily small values.

Fig. 3(b) corresponds to the low noise level, where the improved performance of the proposed algorithm, compared to that of LMS-based algorithms, is even more enhanced.

It suffices to say that this enhanced performance is achieved at the expense of higher complexity. The LMS-based algorithms require $\mathcal{O}(L)$ multiply/add operations, while the APWL1 demands q times more multiply/add operations. However, in a parallel processing environment, the dependence on q can be relaxed.

In Fig. 3(c) the performance improvement of APWL1, as the q value is increasing, is examined and compared to the performance of the batch LASSO algorithm using the true delta value, i.e., $\delta := \|\boldsymbol{h}_*\|_{\ell_1}$. Moreover, the batch weighted LASSO solution [30] is also given. The test is performed for $\sigma_n^2 = 0.1$ Clearly, the convergence speed rapidly improves as q increases, and the rate of improvement is more noticeable in the range of small values of q.

Observe that for large values of q, the performance gets "closer" to the one obtained by the LASSO method. Of course, the larger the q the "heavier" the method becomes from a computationally point of view. However, even for the value of $q = 60$ the complexity remains much lower than any adaptive implementation of LASSO. For example, TWL requires the solution of a convex program per measurement[1]. On the other hand the complexity of the RLASSO implementation rises up to the order of $\mathcal{O}\left(rL^2\right)$, where r is the number of iterations for the cost function minimization in [7, (7)] which, in the specific example, needed to be larger than L in order the method to converge for all the realizations that were involved. Simplified algorithms for the TWL and TNWL, based on the coordinate descent rational, of complexity $\mathcal{O}\left(L^2\right)$, are presented in [11].

In the sequel, we turn our attention to the estimation of a large vector. We will realize it in the context of a signal reconstruction task. We assume a sparse signal vector of 2000 components with $S = 20$

[1]As it was suggested in [11], the SeDuMi package [33] was used for the solution of the convex programs. This results to $\mathcal{O}\left(L^3\right)$ complexity.

arbitrarily positioned nonzero components having values drawn from a zero-mean normal distribution of unit variance. In this case, the observations are obtained from inner products of the unknown signal with independent random measurement vectors, having values distributed according to zero-mean normal distribution of unit variance. The results, are depicted in Fig. 4 for $\sigma_n^2 = 0.1$ (SNR=-10dB), and $\sigma_n^2 = 0.001$ (SNR=10dB), drawn with solid and dashed lines, respectively. It is readily observed that the same trend, which was discussed in the previous experiments, holds true for this example. It must be pointed out that in the signal reconstruction task, the input signal may not necessarily have the shift invariance property [34], [35]. Hence, techniques that build around this property and have extensively been used in order to reduce complexity in the context of LS algorithms, are not applicable for such a problem. Both, LMS and the proposed algorithmic scheme do not utilize this property.

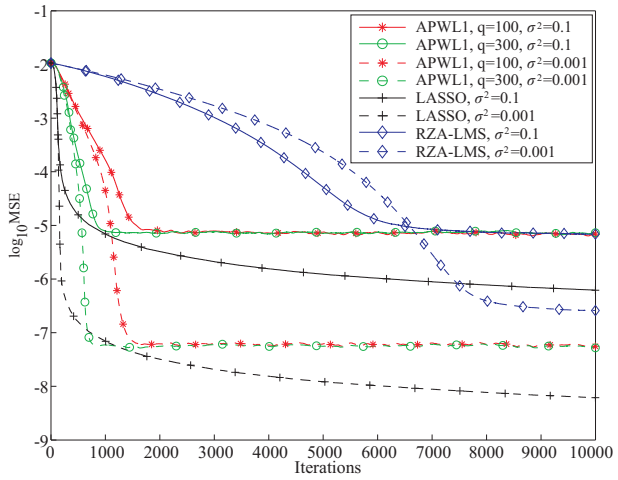

Fig. 4. Sparse signal reconstruction example with $L = 2000$ and $S = 20$, for high and low noise levels.

B. Time-varying case.

It is by now well established in the adaptive filtering community, e.g., [34], that convergence speed and tracking ability of an algorithm do not, necessarily, follow the same trend. An algorithm may have good converging properties, yet its tracking ability to time variations may not be good, or vice versa. There are many cases where LMS tracks better than the RLS. Although the theoretical analysis of the tracking performance is much more difficult, due to the non-stationarity of the environment, related simulated examples are always needed to demonstrate the performance of an adaptive algorithm in such environments. To this end, in this section, the performance of the proposed method to track time-varying

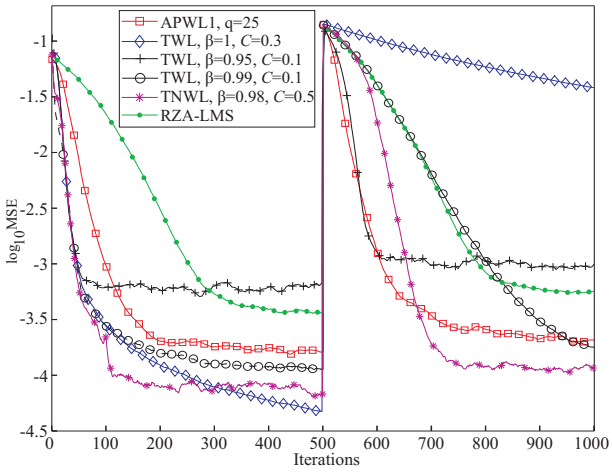

Fig. 5. Time-varying sparse system identification example. The system impulse response changes abruptly at iteration #501.

sparse systems is investigated. Both, the number of nonzero elements of $\boldsymbol{h}_*$ as well as the values of the system's coefficients are allowed to undergo sudden changes. This is a typical scenario used in adaptive filtering in order to study the tracking performance of an algorithm in practice. The system used in the experiments is 100 coefficients long. The system change is realized as follows: For the first 500 time instances, the first 5 coefficients are taking values drawn from a zero-mean normal distribution with variance 1. Then, at time instance $n = 501$ the second and the fourth coefficients are changed to zero, and all the odd coefficients from #7 to #15 are taking random values from the same distribution. Note that the sparsity level, S, also changes at time instance $n = 501$, and it becomes 8 instead of 5.

In the specific time-varying setting, apart from APWL1 and RZA-LMS, the performance of TWL and TNWL will be also studied. As far as the adaptation law of the regularisation parameter λ_n of TWL and TNWL is concerned, we have adopted a similar to the one used in [11]:

$$\lambda_n = \sqrt{2\sigma^2 C \log L}\sqrt{\sum_{i=1}^{n} \beta^{2(n-i)}}. \tag{14}$$

Particularly, the only difference from [11], is that an extra constant parameter C is present, which controls the rate of change of λ_n. We have observed that the TWL and TNWL performance can be enhanced by properly choosing C. Unless otherwise stated, all the curves related to the TWL and TNWL are optimised with respect to parameter C and the specific C value used is shown in the corresponding figure legend.

The results of the ensemble average of 100 independent runs are shown in Fig. 5 with the noise variance being set equal to 0.1. The curve indicated with squares corresponds to the proposed, APWL1 method with $q = 25$ which apparently performs better than the RZA-LMS method (curve denoted by

dots). The performance of the TWL scheme with forgetting factor $\beta = 1$ is denoted by diamonds. It can be observed that TWL outperforms both APWL1 and RZA-LMS up to time instance 500. This is expected, since LS-type of algorithms are known to have fast converging properties. Note that up to this time instant, the example is similar to that shown with solid curves in Fig. 3(c). However, the algorithm lucks the "agility" of fast tracking the changes that take place after convergence, due to its long memory.

In order to make it track faster, the forgetting factor β has to be decreased, in order to "forget" the remote past. However, this affects its (initial) converging properties and in particular the corresponding error floor. When $\beta = 0.95$ (curve denoted by crosses), the tracking speed of the TWL after $n = 501$ coincides with that of APWL1, albeit at the expense of significantly increased error floor. A similar error floor increase is also noticed in the first period, where TWL converges fast, yet to a steady state of increased misadjustment error, which is even larger than that of RZA-LMS. Adjusting the β parameter to get lower error floors, one has to sacrifice tracking speed. Indeed, for $\beta = 0.99$, (dashed curve denoted by circles) TWL enjoys steady state performance slightly better than that of APWL1. However, the cost one has to pay for this error floor reduction is high; Tracking speed as slow as that exhibited by the RZA-LMS method.

An other point that should be stressed out is the sensitivity of TWL in the selection of the β parameter. Comparing the curves indicated with circles and crosses corresponding to $\beta = 0.99$ and $\beta = 0.95$ respectively, we arrive to the conclusion that just a slight variation of the β value (difference of only 0.04) causes a substantial difference in the method behaviour. There is always a question on the way the β is chosen in practical time-varying conditions.

The TNWL, with $\beta = 0.98$, attains the best steady state performance after the abrupt system change showing with the curve denoted by stars. Note, however, that the tracking speed is slower than that of APWL1. Like TWL, the TNWL method demonstrates high sensitivity to the β values however, for brevity TNWL curves with different β are not shown. It will be discussed next, that both TWL and TNWL exhibit a relatively high sensitivity with respect to the specific choice of λ_n adaptation law, i.e., in our case this is translated to sensitivity with respect to the choice of parameter C.

In Fig. 6 the performance curves denoted by squares, circles and asterisks are exactly the same as those shown in Fig. 5 corresponding to APWL1, TWL with $\beta = 0.99$, $C = 0.1$ and TNWL with $\beta = 0.98$, $C = 0.5$, respectively. Studying the sensitivity of TWL with respect to C, we set $C = 1$ instead of 0.1, which performed the best. The corresponding curve is the one denoted with diamonds demonstrating a significant performance degradation. We should mention that when $C = 1$, λ_n follows exactly the adaptation law suggested in [11]. However, in our case, that the nonzero system components are not

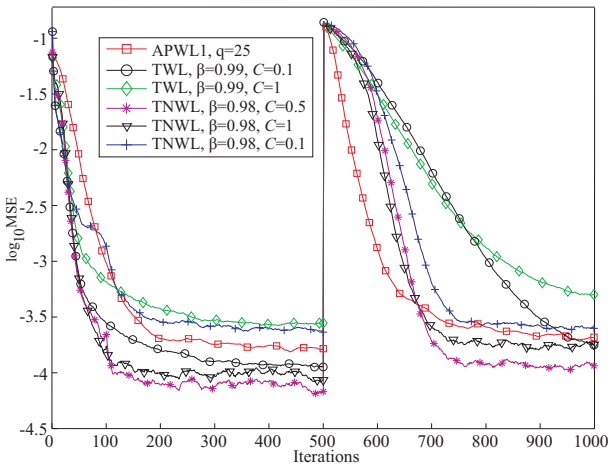

Fig. 6. Sensitivity of TWL and TNWL to the C parameter.

simply set to a high value but they are normally distributed, $C = 1$ is not the optimum value.

In a similar fashion a comparison between the curve indicated with asterisks (optimised, C=0.5) with those indicated with crosses (C=0.1) and triangles (C=1), demonstrates a sensitivity of the TNWL method on the λ_n adaptation. Particularly, both non-optimum C cases lead to performance degradation which brings the performance curves of TNWL to closer or even at higher levels compared to the proposed method.

There are two issues related to the proposed method that have to be discussed for the time-varying case. The first concerns the value of δ and the other the adaptation strategy of ϵ'_n. Physical reasoning suggests that δ, for the weighted ℓ_1 ball, should be set equal to 5 for the first 500 iterations and then take the value 8. However, the actual sparsity levels can not be known in advance. As a result, in the example of Fig. 5, δ was fixed to 9. As it will be discussed soon, the method is rather insensitive against overestimated δ values. Concerning ϵ'_n, the adaptation strategy discussed in the previous section, needs a slight modification. Due to the fact that the system undergoes changes, the algorithm has to be alert to track changes. In order to achieve this, the algorithm must have the ability to monitor abrupt changes of the orbit $(\boldsymbol{h}_n)_{n\in\mathbb{Z}_{\geq 0}}$. Whenever the estimated impulse response changes considerably, and such a change also appears in the orbit $(\boldsymbol{h}_n)_{n\in\mathbb{Z}_{\geq 0}}$, ϵ'_n in (13) is reset to $\epsilon' + 0.2$ and it is gradually reduced adopting a mechanism similar to the one used in the previous example.

IX. Sensitivity of APWL1 to Non ideal Parameter Setting

The robustness of any technique is affected by its sensitivity to non "optimized" configurations. In this section, the sensitivity of APWL1 on δ and ϵ is examined. The sensitivity of APWL1 is compared to the sensitivity that LASSO and LMS-based algorithms exhibit with respect to their associated parameters.

A. *Comparing to LASSO.*

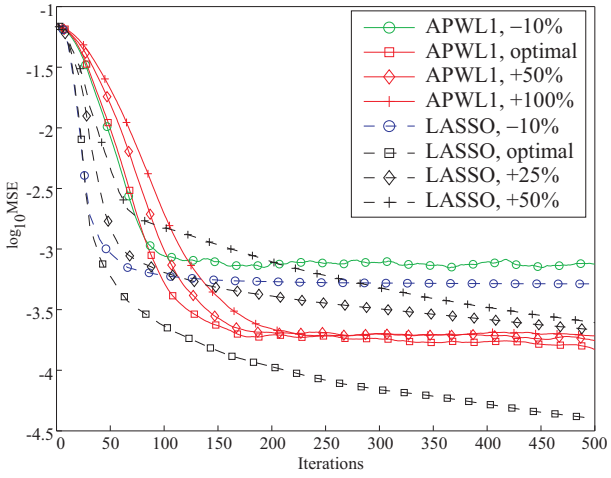

Fig. 7. Sensitivity of APWL1 and LASSO to the δ parameter.

In Fig. 7, the solid lines indicated by diamonds, crosses and circles correspond to the performance of the APWL1, with $q = 30$, when the true δ parameter is overestimated by 50%, 100% or underestimated by 10%, respectively. The system $\boldsymbol{h}_*$ under consideration has $L = 100$, $S = 5$ and $\sigma_n^2 = 0.1$. The best performance, drawn with the solid curve indicated with squares, is achieved when the APWL1 method is supplied with the true δ value, i.e., when $\delta = S$. We observe that the tolerance in δ underestimation is very limited, since even an underestimation by 10% leads to a significant performance degradation. On the other hand, APWL1 is rather insensitive to overestimation. Indeed, overestimation even by 100%, compared to the true value, leads to acceptable results. For comparison, the sensitivity of the standard LASSO is presented with dashed lines. In this case, the optimized δ value equals to $\|\boldsymbol{h}_*\|_{\ell_1}$. The sensitivity of LASSO is clearly higher, particularly to the steady-state region. Observe that only a 25% deviation from the optimum value (dashed line with diamonds) causes enough performance degradation to bring LASSO at a higher MSE regime, compared APWL1. Moreover, LASSO, similarly to APWL1, exhibits limited tolerance in underestimated δ values.

B. Comparing to LMS-based techniques.

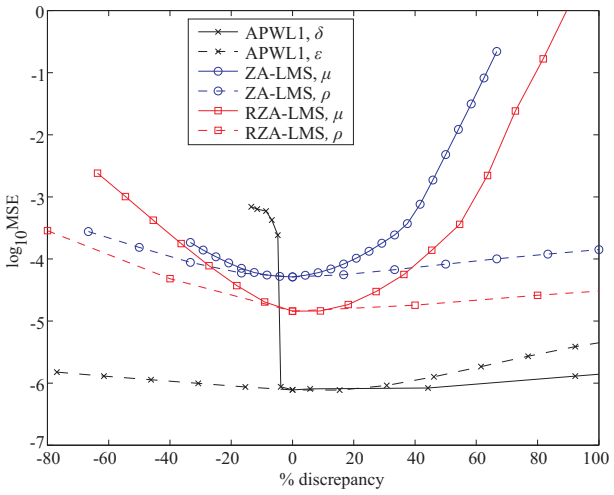

Fig. 8. Sensitivity of the LMS-based methods on the μ and ρ parameters compared to the sensitivity of APWL1 to δ and ϵ.

Besides the δ parameter, APWL1 also needs specification of the hyperslabs width, i.e., the parameter ϵ. On the other hand, LMS-based methods need the specification of μ and ρ. Fig. 8 shows the performance degradation of APWL1 (curves with x-crosses), ZA-LMS (curves with circles) and RZA-LMS (curves with rectangles), when they are configured with parameter values which deviate from the "optimum" ones. The x-axis indicates deviation, from the "optimal" values, in percentage. The problem setting is the one shown in Fig. 3(b) and the reported MSE is always evaluated at time instance 450, where convergence is assumed to have been achieved. The 0% discrepancy point coincides with the best achieved performance of each method. For the LMS-based methods, the solid and dashed curves correspond to μ and ρ, respectively. For the APWL1, the dashed and the solid curves correspond to ϵ and δ, respectively. Starting with the latter parameter, as expected from the discussion in section IX-A, even a slight underestimation, i.e., negative deviation from the optimum, leads to a sudden performance degradation. On the positive side, the method exhibits a very low sensitivity. With respect to ϵ, the sensitivity of APWL1 is similar to the sensitivity exhibited by the LMS-based methods on the ρ parameter. However, LMS-based methods show an increased sensitivity on the μ parameter for both negative and positive deviation. In addition, the optimum μ value depends on the length of $\boldsymbol{h}_*$, as it is the case with the standard non-regularized LMS [35].

X. Conclusions

A novel efficient algorithm, of linear complexity, for sparse system adaptive identification was presented, based on set theoretic estimation arguments. Sparsity was exploited by the introduction of a sequence of weighted ℓ_1 balls. The algorithm consists of a sequence of projections on hyperslabs, that measure data mismatch with respect to the training data, and on the weighted ℓ_1 balls. The projection mapping on a weighted ℓ_1 ball has been derived and a full convergence proof of the algorithm has been established. A comparative performance analysis, using simulated data, was performed against the recently developed online sparse LMS and sparse LS-type of algorithms.

Appendix A

The metric projection mapping onto the weighted ℓ_1 ball $B_{\ell_1}[\boldsymbol{w}, \delta]$

The results in this section are stated for any Euclidean space $\mathbb{R}^l$, where $l \in \overline{1, L}$. Moreover, given two vectors $\boldsymbol{x} := [x_1, \ldots, x_l]^T, \boldsymbol{y} := [y_1, \ldots, y_l]^T \in \mathbb{R}^l$, then the notation $\boldsymbol{x} \leq (<) \boldsymbol{y}$ means that $\forall i \in \overline{1, l}$, $x_i \leq (<) y_i$.

Define $Q_l := B_{\ell_1}[\boldsymbol{w}, \delta] \cap \mathbb{R}^l_{\geq 0}$, where $\mathbb{R}^l_{\geq 0}$ stands for the non-negative hyperorthant of $\mathbb{R}^l$ (see Fig. 2). Define also the following closed halfspace: $H_l^- := \{\boldsymbol{u} \in \mathbb{R}^l : \sum_{i=1}^l w_i u_i = \boldsymbol{w}^T \boldsymbol{u} \leq \delta\}$. Clearly the boundary of H_l^- is the hyperplane: $H_l := \{\boldsymbol{u} \in \mathbb{R}^l : \sum_{i=1}^l w_i u_i = \boldsymbol{w}^T \boldsymbol{u} = \delta\}$. It is easy to verify that $Q_l = H_l^- \cap \mathbb{R}^l_{\geq 0}$.

Lemma 1. 1) For any $\boldsymbol{x} \in \mathbb{R}^l$, the projection $P_{B_{\ell_1}[\boldsymbol{w},\delta]}(\boldsymbol{x})$ belongs to the same hyperorthant as $\boldsymbol{x}$ does, i.e., if $\boldsymbol{x}_* := P_{B_{\ell_1}[\boldsymbol{w},\delta]}(\boldsymbol{x})$, then $\mathrm{sgn}(x_{*,i}) = \mathrm{sgn}(x_i)$, $\forall i \in \overline{1, l}$.

2) Define the mapping $\mathrm{abs} : \boldsymbol{x} = [x_1, \ldots, x_l]^T \mapsto [|x_1|, \ldots, |x_l|]^T$, $\forall \boldsymbol{x} \in \mathbb{R}^l$. It can be easily verified that abs is an one-to-one mapping of any hyperorthant of $\mathbb{R}^l$ onto $\mathbb{R}^l_{\geq 0}$, i.e., it is a bijection. Fix arbitrarily an $\boldsymbol{x} \in \mathbb{R}^l$. Consider the mapping abs which bijectively maps the hyperorthant, in which $\boldsymbol{x}$ is located, to $\mathbb{R}^l_{\geq 0}$. Then, $P_{B_{\ell_1}[\boldsymbol{w},\delta]}(\boldsymbol{x}) = \mathrm{abs}^{-1}\left(P_{B_{\ell_1}[\boldsymbol{w},\delta]}(\mathrm{abs}(\boldsymbol{x}))\right)$, where abs^{-1} stands for the inverse mapping of abs. In other words, in order to calculate the projection mapping onto $B_{\ell_1}[\boldsymbol{w}, \delta]$, it is sufficient to study only the case of $\mathbb{R}^l_{\geq 0}$.

Proof:

1) Without any loss of generality, assume that $\boldsymbol{x}$ belongs to the non-negative hyperorthant of $\mathbb{R}^l$. We will show that every component of $\boldsymbol{x}_*$ is, also, non-negative. In order to derive a contradiction, assume that there exist some negative components of $\boldsymbol{x}_*$. To make the proof short, and with no loss of generality, assume that the only negative component of $\boldsymbol{x}_*$ is $x_{*,1}$. Define the vector $\boldsymbol{u}_*$ such that

$u_{*,1} := 0$ and $u_{*,i} := x_{*,i}$, $\forall i \in \overline{2,l}$. Since $\boldsymbol{x}_* \in B_{\ell_1}[\boldsymbol{w}, \delta]$, we have that $\sum_{i=1}^{l} w_i|x_{*,i}| \leq \delta$, which easily leads to $\sum_{i=1}^{l} w_i|u_{*,i}| = \sum_{i=2}^{l} w_i|x_{*,i}| \leq \sum_{i=1}^{l} w_i|x_{*,i}| \leq \delta$, i.e., $\boldsymbol{u}_* \in B_{\ell_1}[\boldsymbol{w}, \delta]$. Moreover, notice that since $x_{*,1} < 0 = u_{*,1}$, then $x_1 - x_{*,1} > x_1 - u_{*,1} = x_1 \geq 0$. Hence, $\|\boldsymbol{x} - \boldsymbol{u}_*\|^2 < (x_1 - x_{*,1})^2 + \sum_{i=2}^{l}(x_i - x_{*,i})^2 = \|\boldsymbol{x} - \boldsymbol{x}_*\|^2$. This contradicts the fact that $\boldsymbol{x}_* = P_{B_{\ell_1}[\boldsymbol{w},\delta]}(\boldsymbol{x})$, and establishes Lemma 1.1.

2) Fix arbitrarily an $\boldsymbol{x} \in \mathbb{R}^l$. As we have seen before, $P_{B_{\ell_1}[\boldsymbol{w},\delta]}(\boldsymbol{x})$ will be located in the same hyperorthant as $\boldsymbol{x}$. Let any $\boldsymbol{u} \in S_l$, where S_l stands for the intersection of $B_{\ell_1}[\boldsymbol{w}, \delta]$ with the same hyperorthant where $\boldsymbol{x}$ belongs to. As a result, we have that $\operatorname{sgn}(x_i) = \operatorname{sgn}(u_i)$, $\forall i \in \overline{1,l}$, and after a bit of algebra [36] we get $\|\boldsymbol{x} - \boldsymbol{u}\|^2 = \sum_{i=1}^{l}(\operatorname{sgn}(x_i)|x_i| - \operatorname{sgn}(x_i)|u_i|)^2 = \|\operatorname{abs}(\boldsymbol{x}) - \operatorname{abs}(\boldsymbol{u})\|^2$. Notice here that abs is a bijection from S_l to Q_l, so that the previous equality results into the following:

$$\|\operatorname{abs}(\boldsymbol{x}) - P_{B_{\ell_1}[\boldsymbol{w},\delta]}(\operatorname{abs}(\boldsymbol{x}))\| = \min_{\boldsymbol{u}' \in Q_l} \|\operatorname{abs}(\boldsymbol{x}) - \boldsymbol{u}'\| = \min_{\boldsymbol{u} \in S_l} \|\operatorname{abs}(\boldsymbol{x}) - \operatorname{abs}(\boldsymbol{u})\|$$

$$= \min_{\boldsymbol{u} \in S_l} \|\boldsymbol{x} - \boldsymbol{u}\| = \|\boldsymbol{x} - P_{B_{\ell_1}[\boldsymbol{w},\delta]}(\boldsymbol{x})\| = \|\operatorname{abs}(\boldsymbol{x}) - \operatorname{abs}\left(P_{B_{\ell_1}[\boldsymbol{w},\delta]}(\boldsymbol{x})\right)\|.$$

Hence, by the uniqueness of the projection, $\operatorname{abs}\left(P_{B_{\ell_1}[\boldsymbol{w},\delta]}(\boldsymbol{x})\right) = P_{B_{\ell_1}[\boldsymbol{w},\delta]}(\operatorname{abs}(\boldsymbol{x}))$, and Lemma 1.2 is established.

■

Lemma 2. Let an $\boldsymbol{x} \in \mathbb{R}^l_{\geq 0} \setminus Q_l$, and

$$\boldsymbol{x}_* := P_{H_l^-}(\boldsymbol{x}) = \boldsymbol{x} - \frac{\max\{0, \boldsymbol{x}^T\boldsymbol{w} - \delta\}}{\|\boldsymbol{w}\|^2}\boldsymbol{w}. \tag{15}$$

1) Assume that $\boldsymbol{x}_* > \boldsymbol{0}$. Then, $P_{Q_l}(\boldsymbol{x}) = P_{H_l^-}(\boldsymbol{x})$.
2) Make the following partitions $\boldsymbol{x} = \left[\begin{smallmatrix}\hat{\boldsymbol{x}}\\ \tilde{\boldsymbol{x}}\end{smallmatrix}\right]$, $\boldsymbol{x}_* = \left[\begin{smallmatrix}\hat{\boldsymbol{x}}_*\\ \tilde{\boldsymbol{x}}_*\end{smallmatrix}\right]$, where $\hat{l}, \tilde{l} \in \overline{1,l}$, $\hat{l} + \tilde{l} = l$, and $\hat{\boldsymbol{x}}, \hat{\boldsymbol{x}}_* \in \mathbb{R}^{\hat{l}}$, $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}_* \in \mathbb{R}^{\tilde{l}}$. Assume, now, that there exists an $\tilde{l} \in \overline{1,l}$ such that $\tilde{\boldsymbol{x}}_* \leq \boldsymbol{0}$. Then,

$$P_{Q_l}(\boldsymbol{x}) = [P_{Q_{\hat{l}}}(\hat{\boldsymbol{x}})^T,\ \boldsymbol{0}^T]^T.$$

Proof:

1) Since $\boldsymbol{x}_* := P_{H_l^-}(\boldsymbol{x}) > \boldsymbol{0}$, it is clear that $\boldsymbol{x}_* \in H_l^- \cap \mathbb{R}^l_{\geq 0} = Q_l$. Hence,

$$\min_{\boldsymbol{u} \in Q_l} \|\boldsymbol{x} - \boldsymbol{u}\| \leq \|\boldsymbol{x} - \boldsymbol{x}_*\| = \|\boldsymbol{x} - P_{H_l^-}(\boldsymbol{x})\| = \min_{\boldsymbol{u} \in H_l^-} \|\boldsymbol{x} - \boldsymbol{u}\| \leq \min_{\boldsymbol{u} \in Q_l} \|\boldsymbol{x} - \boldsymbol{u}\|,$$

where the last inequality comes from $Q_l \subset H_l^-$. Thus, $\|\boldsymbol{x} - P_{H_l^-}(\boldsymbol{x})\| = \min_{\boldsymbol{u} \in Q_l} \|\boldsymbol{x} - \boldsymbol{u}\|$. Therefore, by the uniqueness of the projection, $P_{Q_l}(\boldsymbol{x}) = P_{H_l^-}(\boldsymbol{x})$, and Lemma 2.1 is established.

2) Since H_l is a hyperplane, $\forall \boldsymbol{u} \in H_l$, $(\boldsymbol{u}-\boldsymbol{x}_*)^T(\boldsymbol{x}-\boldsymbol{x}_*)=0$, which implies, of course, that $\forall \boldsymbol{u} \in H_l \cap \mathbb{R}^l_{\geq 0}$, $(\boldsymbol{u}-\boldsymbol{x}_*)^T(\boldsymbol{x}-\boldsymbol{x}_*)=0$. Thus, $\forall \boldsymbol{u} \in H_l \cap \mathbb{R}^l_{\geq 0}$,

$$\|\boldsymbol{u}-\boldsymbol{x}\|^2_{\mathbb{R}^l} = \|\boldsymbol{u}-\boldsymbol{x}_*\|^2_{\mathbb{R}^l} + \|\boldsymbol{x}_*-\boldsymbol{x}\|^2_{\mathbb{R}^l} = \|\hat{\boldsymbol{u}}-\hat{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\hat{l}}} + \|\tilde{\boldsymbol{u}}-\tilde{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\tilde{l}}} + \|\boldsymbol{x}_*-\boldsymbol{x}\|^2_{\mathbb{R}^l}. \tag{16}$$

This in turn implies that

$$\begin{aligned} P_{Q_l}(\boldsymbol{x}) &= \arg\min\{\|\boldsymbol{u}-\boldsymbol{x}\|^2_{\mathbb{R}^l} :\ \boldsymbol{u} \in H_l \cap \mathbb{R}^l_{\geq 0}\} \\ &= \arg\min\{\|\hat{\boldsymbol{u}}-\hat{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\hat{l}}} + \|\tilde{\boldsymbol{u}}-\tilde{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\tilde{l}}} :\ \hat{\boldsymbol{u}} \in \mathbb{R}^{\hat{l}}_{\geq 0}, \tilde{\boldsymbol{u}} \in \mathbb{R}^{\tilde{l}}_{\geq 0}, \hat{\boldsymbol{u}}^T\hat{\boldsymbol{w}} + \tilde{\boldsymbol{u}}^T\tilde{\boldsymbol{w}} = \delta\}. \end{aligned} \tag{17}$$

By our initial assumption $\tilde{\boldsymbol{x}}_* \leq \boldsymbol{0}$. Hence, it is easy to verify that $\forall \hat{\boldsymbol{u}} \in \mathbb{R}^{\hat{l}}_{\geq 0}$, $\forall \tilde{\boldsymbol{u}} \in \mathbb{R}^{\tilde{l}}_{\geq 0} \setminus \{\boldsymbol{0}\}$, $\|\hat{\boldsymbol{u}}-\hat{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\hat{l}}} + \|\boldsymbol{0}-\tilde{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\tilde{l}}} < \|\hat{\boldsymbol{u}}-\hat{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\hat{l}}} + \|\tilde{\boldsymbol{u}}-\tilde{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\tilde{l}}}$, which evidently suggests that

$$\begin{aligned} &\arg\min\{\|\hat{\boldsymbol{u}}-\hat{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\hat{l}}} + \|\tilde{\boldsymbol{u}}-\tilde{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\tilde{l}}} :\ \hat{\boldsymbol{u}} \in \mathbb{R}^{\hat{l}}_{\geq 0}, \tilde{\boldsymbol{u}} \in \mathbb{R}^{\tilde{l}}_{\geq 0}, \hat{\boldsymbol{u}}^T\hat{\boldsymbol{w}} + \tilde{\boldsymbol{u}}^T\tilde{\boldsymbol{w}} = \delta\} \\ &= \arg\min\{\|\hat{\boldsymbol{u}}-\hat{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\hat{l}}} + \|\tilde{\boldsymbol{u}}-\tilde{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\tilde{l}}} :\ \hat{\boldsymbol{u}} \in \mathbb{R}^{\hat{l}}_{\geq 0}, \hat{\boldsymbol{u}}^T\hat{\boldsymbol{w}} = \delta, \tilde{\boldsymbol{u}} = \boldsymbol{0}\} \end{aligned} \tag{18}$$

Now, since $\boldsymbol{x} \in \mathbb{R}^l_{\geq 0} \setminus Q_l$, it is clear by the geometry of the Q_l that $P_{Q_l}(\boldsymbol{x})$ will be located on $H_l \cap \mathbb{R}^l_{\geq 0}$. Hence, by (16), (17), and (18), and after a bit of algebra [36], one can verify the following:

$$\begin{aligned} P_{Q_l}(\boldsymbol{x}) &= \arg\min\{\|\hat{\boldsymbol{u}}-\hat{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\hat{l}}} + \|\tilde{\boldsymbol{u}}-\tilde{\boldsymbol{x}}_*\|^2_{\mathbb{R}^{\tilde{l}}} :\ \hat{\boldsymbol{u}} \in \mathbb{R}^{\hat{l}}_{\geq 0}, \hat{\boldsymbol{u}}^T\hat{\boldsymbol{w}} = \delta, \tilde{\boldsymbol{u}} = \boldsymbol{0}\} \\ &= \arg\min\{\|\hat{\boldsymbol{u}}-\hat{\boldsymbol{x}}\|^2_{\mathbb{R}^l} :\ \hat{\boldsymbol{u}} \in \mathbb{R}^{\hat{l}}_{\geq 0}, \hat{\boldsymbol{u}}^T\hat{\boldsymbol{w}} = \delta, \tilde{\boldsymbol{u}} = \boldsymbol{0}\} = [P_{Q_{\hat{l}}}(\hat{\boldsymbol{x}})^T,\ \boldsymbol{0}^T]^T. \end{aligned}$$

This establishes Lemma 2.2.

■

Lemma 3. Assume an $\boldsymbol{x} \in \mathbb{R}^l_{\geq 0}$ such that $\forall i \in \overline{1, l-1}$, $\frac{x_i}{w_i} \geq \frac{x_{i+1}}{w_{i+1}}$. Moreover, let $\boldsymbol{x}_* := P_{H_l^-}(\boldsymbol{x})$. Assume that there exists an $i_0 \in \overline{1, l}$ such that $x_{*,i_0} \leq 0$. Then, $\forall i \geq i_0$, $x_{*,i} \leq 0$.

Proof: Here we consider only the case where $\boldsymbol{x} \in \mathbb{R}^l_{\geq 0} \setminus Q_l$, i.e., $\boldsymbol{x}^T\boldsymbol{w} - \delta > 0$. Notice by (15) that

$$x_{*,i} \leq 0 \Leftrightarrow \frac{x_i}{w_i} \leq \frac{\boldsymbol{x}^T\boldsymbol{w}-\delta}{\|\boldsymbol{w}\|^2}. \tag{19}$$

Now, notice also that by the construction of $\boldsymbol{x}$ and by our initial assumption, we have that

$$\forall i \geq i_0, \quad \frac{x_i}{w_i} \leq \frac{x_{i_0}}{w_{i_0}} \leq \frac{\boldsymbol{x}^T\boldsymbol{w}-\delta}{\|\boldsymbol{w}\|^2}.$$

However, by (19), this is equivalent to $x_{*,i} \leq 0$, $\forall i \geq i_0$, which establishes Lemma 3. ■

A. The proof of Theorem 1.

Notice that Step 1 is due to Lemma 1. Step 4b refers to the attempt of the algorithm to locate the non-positive components of a vector, according to Lemma 3. Step 4c refers to Lemma 2.1, while Step 4d corresponds to Lemma 2.2.

APPENDIX B
THE PROOF OF THEOREM 2

A. Preliminaries.

Definition 1 (Subgradient and subdifferential [37])**.** Given a convex function $\Theta : \mathbb{R}^L \to \mathbb{R}$, a subgradient $\Theta'(\boldsymbol{x})$ of Θ at $\boldsymbol{x} \in \mathbb{R}^L$ is an element of $\mathbb{R}^L$, which satisfies the following property: $\Theta'(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}) + \Theta(\boldsymbol{x}) \leq \Theta(\boldsymbol{y})$, $\forall \boldsymbol{y} \in \mathbb{R}^L$. The set of all the subgradients of Θ at the point $\boldsymbol{x}$ will be called the subdifferential of Θ at $\boldsymbol{x}$, and will be denoted by $\partial\Theta(\boldsymbol{x})$. Notice that if Θ is (Gâteaux) differentiable at $\boldsymbol{x}$, then the only subgradient of Θ at $\boldsymbol{x}$ is its differential.

Fact 1. The subdifferential of the metric distance function $d(\cdot, C)$ to a closed convex set $C \subset \mathbb{R}^L$ is given as follows [37]:

$$\partial d(\boldsymbol{x}, C) = \begin{cases} N_C(\boldsymbol{x}) \cap B[\boldsymbol{0}, 1], & \boldsymbol{x} \in C, \\ \frac{\boldsymbol{x} - P_C(\boldsymbol{x})}{d(\boldsymbol{x}, C)}, & \boldsymbol{x} \notin C, \end{cases}$$

where $N_C(\boldsymbol{x}) := \{\boldsymbol{y} \in \mathbb{R}^L : \ \boldsymbol{y}^T(\boldsymbol{f} - \boldsymbol{x}) \leq 0, \forall \boldsymbol{f} \in C\}$, and $B[\boldsymbol{0}, 1] := \{\boldsymbol{y} \in \mathbb{R}^L : \ \|\boldsymbol{y}\| \leq 1\}$. Notice that $\forall \boldsymbol{x} \in \mathbb{R}^L$, $\|d'(\boldsymbol{x}, C)\| \leq 1$, where $d'(\boldsymbol{x}, C)$ stands for any subgradient in $\partial d(\boldsymbol{x}, C)$.

We will give, now, an equivalent description of the Algorithm in (9), which will help us in proving several properties of the algorithm.

Lemma 4 (Equivalent description of the Algorithm in (9))**.** Given the sequence of estimates $(\boldsymbol{h}_n)_{n \in \mathbb{Z}_{\geq 0}}$, define the following non-negative functions:

$$\forall n \in \mathbb{Z}_{\geq 0}, \forall \boldsymbol{x} \in \mathbb{R}^L, \quad \Theta_n(\boldsymbol{x}) := \begin{cases} \sum_{j \in \mathcal{I}_n} \frac{\omega_j^{(n)} d(\boldsymbol{h}_n, S_j[\epsilon])}{L_n} d(\boldsymbol{x}, S_j[\epsilon]), & \text{if } \mathcal{I}_n \neq \emptyset, \\ 0, & \text{if } \mathcal{I}_n = \emptyset, \end{cases} \tag{20}$$

where $L_n := \sum_{j \in \mathcal{I}_n} \omega_j^{(n)} d(\boldsymbol{h}_n, S_j[\epsilon])$. Then, (9) takes the following form:

$$\forall n \in \mathbb{Z}_{\geq 0}, \quad \boldsymbol{h}_{n+1} := \begin{cases} P_{B_{\ell_1}[\boldsymbol{w}_n, \delta]} \left(\boldsymbol{h}_n - \lambda_n \frac{\Theta_n(\boldsymbol{h}_n)}{\|\Theta_n'(\boldsymbol{h}_n)\|^2} \Theta_n'(\boldsymbol{h}_n) \right), & \text{if } \Theta_n'(\boldsymbol{h}_n) \neq 0, \\ P_{B_{\ell_1}[\boldsymbol{w}_n, \delta]}(\boldsymbol{h}_n), & \text{if } \Theta_n'(\boldsymbol{h}_n) = 0, \end{cases} \tag{21}$$

where $\lambda_n \in (0,2)$, $\forall n \in \mathbb{Z}_{\geq 0}$, and $\Theta_n'(\boldsymbol{h}_n)$ is a subgradient of Θ_n at $\boldsymbol{h}_n$.

Proof: First, a few comments regarding L_n in (20) are in order. It can be easily verified by the definition of $\mathcal{I}_n$ in (8) that $\forall j \in \mathcal{I}_n$, $d(\boldsymbol{h}_n, S_j[\epsilon]) > 0$. Hence, $L_n > 0$, and (20) is well-defined. The reason for introducing L_n in the design is to give the freedom to the extrapolation parameter μ_n in (9) to be able to take values greater than or equal to 2; recall that $\mu_n \in (0, 2\mathcal{M}_n)$ and $\mathcal{M}_n \geq 1$, $\forall n \in \mathbb{Z}_{\geq 0}$, in (10).

Basic calculus on subdifferentials [37] suggests that

$$\forall \boldsymbol{x} \in \mathbb{R}^L, \quad \partial\Theta_n(\boldsymbol{x}) := \begin{cases} \sum_{j\in\mathcal{I}_n} \frac{\omega_j^{(n)} d(\boldsymbol{h}_n, S_j[\epsilon])}{L_n} \partial d(\boldsymbol{x}, S_j[\epsilon]), & \text{if } \mathcal{I}_n \neq \emptyset, \\ \{\mathbf{0}\}, & \text{if } \mathcal{I}_n = \emptyset. \end{cases}$$

Hence, in the case where $\mathcal{I}_n \neq \emptyset$, Fact 1 implies that

$$\Theta_n'(\boldsymbol{h}_n) = \sum_{j\in\mathcal{I}_n} \frac{\omega_j^{(n)} d(\boldsymbol{h}_n, S_j[\epsilon])}{L_n} \frac{\boldsymbol{h}_n - P_{S_j[\epsilon]}(\boldsymbol{h}_n)}{d(\boldsymbol{h}_n, S_j[\epsilon])} = \frac{1}{L_n} \sum_{j\in\mathcal{I}_n} \omega_j^{(n)} (\boldsymbol{h}_n - P_{S_j[\epsilon]}(\boldsymbol{h}_n)).$$

In the case where $\mathcal{I}_n = \emptyset$, it is clear by (8) that $\boldsymbol{h}_n \in \bigcap_{j\in\mathcal{J}_n} S_j[\epsilon]$, which results into $\boldsymbol{h}_n - P_{S_j[\epsilon]}(\boldsymbol{h}_n) = \mathbf{0}$, $\forall j \in \mathcal{J}_n$. Thus, it is natural to adopt the following convention: $\sum_{j\in\emptyset} \omega_j^{(n)} (\boldsymbol{h}_n - P_{S_j[\epsilon]}(\boldsymbol{h}_n)) = \mathbf{0}$. Notice by (20), that in the case where $\mathcal{I}_n = \emptyset$, then $\Theta_n'(\boldsymbol{h}_n) = \mathbf{0}$. To conclude,

$$\forall n \in \mathbb{Z}_{\geq 0}, \quad \Theta_n'(\boldsymbol{h}_n) = \frac{1}{L_n} \sum_{j\in\mathcal{I}_n} \omega_j^{(n)} (\boldsymbol{h}_n - P_{S_j[\epsilon]}(\boldsymbol{h}_n)). \tag{22}$$

If we substitute (22) in (21), and if we define $\mu_n := \lambda_n \mathcal{M}_n$, $\forall n \in \mathbb{Z}_{\geq 0}$, where $\mathcal{M}_n$ is given in (10), then we obtain (9).

Let's define, now, $\forall n \in \mathbb{Z}_{\geq 0}$, $\operatorname{lev}_{\leq 0} \Theta_n := \{\boldsymbol{y} \in \mathbb{R}^L : \ \Theta_n(\boldsymbol{y}) \leq 0\}$. In the case where $\mathcal{I}_n \neq \emptyset$, and if $\bigcap_{j\in\mathcal{I}_n} S_j[\epsilon] \neq \emptyset$, then it is easy to verify that $\operatorname{lev}_{\leq 0} \Theta_n = \bigcap_{j\in\mathcal{I}_n} S_j[\epsilon]$. It is also easy to see that in the case where $\bigcap_{j\in\mathcal{I}_n} S_j[\epsilon] = \emptyset$, then $\operatorname{lev}_{\leq 0} \Theta_n = \emptyset$. Let us demonstrate this. Take any $\boldsymbol{x} \in \mathbb{R}^L$. Since $\boldsymbol{x} \notin \bigcap_{j\in\mathcal{I}_n} S_j[\epsilon] = \emptyset$, there exists a $j_0 \in \mathcal{I}_n$ such that $d(\boldsymbol{x}, S_{j_0}[\epsilon]) > 0$. Hence, $\Theta_n(\boldsymbol{x}) \geq \frac{\omega_{j_0}^{(n)} d(\boldsymbol{h}_n, S_{j_0}[\epsilon])}{L_n} d(\boldsymbol{x}, S_{j_0}[\epsilon]) > 0$. In other words, $\forall \boldsymbol{x} \in \mathbb{R}^L$, $\Theta_n(\boldsymbol{x}) > 0 \Rightarrow \operatorname{lev}_{\leq 0} \Theta_n = \emptyset$. If we also make the convention that $\bigcap_{j\in\emptyset} S_j[\epsilon] := \mathbb{R}^L$, to cover the case of $\mathcal{I}_n = \emptyset$, then we can see that

$$\forall n \in \mathbb{Z}_{\geq 0}, \quad \operatorname{lev}_{\leq 0} \Theta_n = \bigcap_{j\in\mathcal{I}_n} S_j[\epsilon].$$

Notice, also, that

$$\boldsymbol{h}_n \in \bigcap_{j\in\mathcal{I}_n} S_j[\epsilon] \Leftrightarrow \left(\omega_j^{(n)} \boldsymbol{h}_n = \omega_j^{(n)} P_{S_j[\epsilon]}(\boldsymbol{h}_n), \ \forall j \in \mathcal{I}_n\right) \Rightarrow \sum_{j\in\mathcal{I}_n} \omega_j^{(n)} (\boldsymbol{h}_n - P_{S_j[\epsilon]}(\boldsymbol{h}_n)) = \Theta_n'(\boldsymbol{h}_n) = \mathbf{0}.$$

In the previous relation, the symbol $\Rightarrow$ becomes $\Leftrightarrow$, if we assume that $\bigcap_{j\in\mathcal{I}_n} S_j[\epsilon] \neq \emptyset$ [23, Proposition 2.12]. Hence, in the case where $\bigcap_{j\in\mathcal{I}_n} S_j[\epsilon] \neq \emptyset$,

$$\boldsymbol{h}_n \in \mathrm{lev}_{\leq 0}\,\Theta_n \Leftrightarrow \Theta_n'(\boldsymbol{h}_n) = \boldsymbol{0}. \tag{23}$$

■

Definition 2 (Subgradient projection mapping [38])**.** Given a convex function $\Theta : \mathbb{R}^L \to \mathbb{R}$, such that $\mathrm{lev}_{\leq 0}\,\Theta \neq \emptyset$, define the *subgradient projection mapping* $T_\Theta : \mathbb{R}^L \to \mathbb{R}^L$ *with respect to* Θ as follows:

$$T_\Theta(\boldsymbol{x}) := \begin{cases} \boldsymbol{x} - \frac{\Theta(\boldsymbol{x})}{\|\Theta'(\boldsymbol{x})\|^2}\Theta'(\boldsymbol{x}), & \text{if } \boldsymbol{x} \notin \mathrm{lev}_{\leq 0}\,\Theta, \\ \boldsymbol{x}, & \text{if } \boldsymbol{x} \in \mathrm{lev}_{\leq 0}\,\Theta, \end{cases}$$

where $\Theta'(\boldsymbol{x})$ stands for an arbitrarily fixed subgradient of Θ at $\boldsymbol{x}$. If I stands for the identity mapping in $\mathbb{R}^L$, the mapping $T_\Theta^{(\lambda)} := I + \lambda(T_\Theta - I)$, $\lambda \in (0,2)$, will be called the *relaxed subgradient projection mapping*. Moreover, similarly to (11), an important property of $T_\Theta^{(\lambda)}$ is the following [38]:

$$\forall \boldsymbol{x} \in \mathbb{R}^L, \forall \boldsymbol{f} \in \mathrm{lev}_{\leq 0}\,\Theta, \quad \frac{2-\lambda}{\lambda}\|\boldsymbol{x} - T_\Theta^{(\lambda)}(\boldsymbol{x})\|^2 \leq \|\boldsymbol{x} - \boldsymbol{f}\|^2 - \|T_\Theta^{(\lambda)}(\boldsymbol{x}) - \boldsymbol{f}\|^2. \tag{24}$$

Now, (11) and (24) can be combined as follows.

Lemma 5. Let a closed convex set $C \subset \mathbb{R}^L$, and a convex function $\Theta : \mathbb{R}^L \to \mathbb{R}$ such that $C \cap \mathrm{lev}_{\leq 0}\,\Theta \neq \emptyset$. Then,

$$\forall \boldsymbol{x} \in \mathbb{R}^L, \forall \boldsymbol{f} \in C \cap \mathrm{lev}_{\leq 0}\,\Theta, \quad \frac{2-\lambda}{2}\|\boldsymbol{x} - P_C T_\Theta^{(\lambda)}(\boldsymbol{x})\|^2 \leq \|\boldsymbol{x} - \boldsymbol{f}\|^2 - \|P_C T_\Theta^{(\lambda)}(\boldsymbol{x}) - \boldsymbol{f}\|^2.$$

Proof: This is a direct consequence of [13, Proposition 1]. ■

Fact 2 ([13])**.** Let a sequence $(\boldsymbol{x}_n)_{n\in\mathbb{Z}_{\geq 0}} \subset \mathbb{R}^L$, and a closed convex set $C \subset \mathbb{R}^L$. Assume that

$$\exists \kappa > 0 : \ \forall \boldsymbol{f} \in C, \ \forall n \in \mathbb{Z}_{\geq 0}, \quad \kappa\|\boldsymbol{x}_{n+1} - \boldsymbol{x}_n\|^2 \leq \|\boldsymbol{x}_n - \boldsymbol{f}\|^2 - \|\boldsymbol{x}_{n+1} - \boldsymbol{f}\|^2.$$

Assume, also, that there exists a hyperplane Π such that the relative interior of the set C with respect to Π is nonempty, i.e., $\mathrm{ri}_\Pi\, C \neq \emptyset$. Then, $\exists \boldsymbol{x}_* \in \mathbb{R}^L : \ \boldsymbol{x}_* = \lim_{n\to\infty} \boldsymbol{x}_n$.

Here, given any $\Upsilon \subset \mathbb{R}^L$, $\mathrm{ri}_\Upsilon\, C := \{\boldsymbol{y} \in \mathbb{R}^L : \ \exists \rho > 0, (B(\boldsymbol{y},\rho) \cap \Upsilon) \subset C\}$. As a byproduct of this definition, the interior of C is defined as $\mathrm{int}\, C := \mathrm{ri}_{\mathbb{R}^L}\, C$. Hence, it becomes clear that if $\mathrm{int}\, C \neq \emptyset$, then we can always find a hyperplane $\Pi \subset \mathbb{R}^L$ such that $\mathrm{ri}_\Pi\, C \neq \emptyset$. This fact will be used in the proof of Theorem 2.4.

Fact 3 ([13])**.** Let $C \subset \mathbb{R}^L$ be a nonempty closed convex set. Assume also an $\mathring{\boldsymbol{f}} \in \operatorname{int} C$, i.e., $\exists \rho > 0$ such that $B(\mathring{\boldsymbol{f}}, \rho) \subset C$. Assume, now, an $\boldsymbol{x} \in \mathbb{R}^L \setminus C$, and a $t \in (0,1)$ such that $\mathring{\boldsymbol{f}} + t(\boldsymbol{x} - \mathring{\boldsymbol{f}}) \notin C$. Then, $d(\boldsymbol{x}, C) > \rho \frac{1-t}{t}$.

Lemma 6. The set of all subgradients of the collection of convex functions $(\Theta_n)_{n \in \mathbb{Z}_{\geq 0}}$, defined in (20), is bounded, i.e., $\forall n \in \mathbb{Z}_{\geq 0}$, $\forall \boldsymbol{x} \in \mathbb{R}^L$, $\|\Theta_n'(\boldsymbol{x})\| \leq 1$.

Proof: Fix arbitrarily an $n \in \mathbb{Z}_{\geq 0}$. Here we deal only with the case $\mathcal{I}_n \neq \emptyset$, since otherwise, the function Θ_n becomes everywhere zero, and for such a function, Lemma 6 holds trivially.

By (20), Fact 1, and some calculus on subdifferentials [36], [37], we obtain that $\forall \boldsymbol{x} \in \mathbb{R}^L$, the norm of any subgradient $\Theta_n'(\boldsymbol{x})$ satisfies the following:

$$\begin{aligned}\|\Theta_n'(\boldsymbol{x})\| &= \|\sum_{j \in \mathcal{I}_n} \frac{\omega_j^{(n)} d(\boldsymbol{h}_n, S_j[\epsilon])}{L_n} d'(\boldsymbol{x}, S_j[\epsilon])\| \\ &\leq \sum_{j \in \mathcal{I}_n:\ \boldsymbol{x} \notin S_j[\epsilon]} \frac{\omega_j^{(n)} d(\boldsymbol{h}_n, S_j[\epsilon])}{L_n} \frac{\|\boldsymbol{x} - P_{S_j[\epsilon]}(\boldsymbol{x})\|}{d(\boldsymbol{x}, S_j[\epsilon])} + \sum_{j \in \mathcal{I}_n:\ \boldsymbol{x} \in S_j[\epsilon]} \frac{\omega_j^{(n)} d(\boldsymbol{h}_n, S_j[\epsilon])}{L_n} = 1.\end{aligned}$$

This establishes Lemma 6. ■

B. The proof of Theorem 2.

1) Assumption 1, Definition 2, and (23) suggest that (21) can be equivalently written as follows: $\forall n \geq z_0$, $\boldsymbol{h}_{n+1} = P_{B_{\ell_1}[\boldsymbol{w}_n, \delta]} T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n)$, where $T_{\Theta_n}^{(\lambda_n)}$ stands for the relaxed subgradient projection mapping with respect to Θ_n. Notice here that $\forall n \geq z_0$, $\operatorname{lev}_{\leq 0} \Theta_n = \bigcap_{j \in \mathcal{I}_n} S_j[\epsilon]$. Thus, by Assumption 1 and Lemma 5, we have that $\forall n \geq z_0, \forall \boldsymbol{f} \in \Omega$,

$$\begin{aligned}0 \leq \frac{2 - \lambda_n}{2} \|\boldsymbol{h}_n - \boldsymbol{h}_{n+1}\|^2 &= \frac{2 - \lambda_n}{2} \|\boldsymbol{h}_n - P_{B_{\ell_1}[\boldsymbol{w}_n, \delta]} T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n)\|^2 \\ &\leq \|\boldsymbol{h}_n - \boldsymbol{f}\|^2 - \|P_{B_{\ell_1}[\boldsymbol{w}_n, \delta]} T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n) - \boldsymbol{f}\|^2 = \|\boldsymbol{h}_n - \boldsymbol{f}\|^2 - \|\boldsymbol{h}_{n+1} - \boldsymbol{f}\|^2 \qquad (25) \\ &\Rightarrow \|\boldsymbol{h}_{n+1} - \boldsymbol{f}\| \leq \|\boldsymbol{h}_n - \boldsymbol{f}\|. \qquad (26)\end{aligned}$$

If we apply $\inf_{\boldsymbol{f} \in \Omega}$ on both sides of (26), we establish our original claim.

2) The next claim is to show that under Assumption 1, the sequence $(\|\boldsymbol{h}_n - \boldsymbol{f}\|)_{n \in \mathbb{Z}_{\geq 0}}$ converges $\forall \boldsymbol{f} \in \Omega$. To this end, fix arbitrarily $\boldsymbol{f} \in \Omega$. By (26), the sequence $(\|\boldsymbol{h}_n - \boldsymbol{f}\|)_{n \geq z_0}$ is non-increasing, and bounded below. Hence, it is convergent. This establishes the claim.

Let Assumptions 1 and 2 hold true. Then, we will show that $\lim_{n \to \infty} \Theta_n(\boldsymbol{h}_n) = 0$. First, we will prove that

$$\lim_{n \to \infty} \frac{\Theta_n(\boldsymbol{h}_n)}{\|\Theta_n'(\boldsymbol{h}_n)\|} = 0. \qquad (27)$$

We will show this by deriving a contradiction. To this end, assume that there exists a $\delta > 0$ and a subsequence $(n_k)_{k\in\mathbb{Z}_{\geq 0}}$ such that $\forall k \in \mathbb{Z}_{\geq 0}$, $\frac{\Theta_{n_k}(\boldsymbol{h}_{n_k})}{\|\Theta'_{n_k}(\boldsymbol{h}_{n_k})\|} \geq \delta$. We can always choose a sufficiently large k_0 such that $\forall k \geq k_0$, $n_k \geq z_0$.

Let, now, any $\boldsymbol{f} \in \Omega$, and recall that $\Omega \subset B_{\ell_1}[\boldsymbol{w}_{n_k}, \delta]$, $\forall k \geq k_0$. Then, verify that the following holds true $\forall k \geq k_0$:

$$\begin{aligned}
\|\boldsymbol{h}_{n_k+1} - \boldsymbol{f}\|^2 &= \|P_{B_{\ell_1}[\boldsymbol{w}_{n_k},\delta]}\left(\boldsymbol{h}_{n_k} - \lambda_{n_k}\frac{\Theta_{n_k}(\boldsymbol{h}_{n_k})}{\|\Theta'_{n_k}(\boldsymbol{h}_{n_k})\|^2}\Theta'_{n_k}(\boldsymbol{h}_{n_k})\right) - \boldsymbol{f}\|^2 \\
&\leq \|\boldsymbol{h}_{n_k} - \lambda_{n_k}\frac{\Theta_{n_k}(\boldsymbol{h}_{n_k})}{\|\Theta'_{n_k}(\boldsymbol{h}_{n_k})\|^2}\Theta'_{n_k}(\boldsymbol{h}_{n_k}) - \boldsymbol{f}\|^2 \\
&= \|\boldsymbol{h}_{n_k} - \boldsymbol{f}\|^2 + \lambda_{n_k}^2\frac{\Theta_{n_k}^2(\boldsymbol{h}_{n_k})}{\|\Theta'_{n_k}(\boldsymbol{h}_{n_k})\|^2} - 2\lambda_{n_k}\frac{\Theta_{n_k}(\boldsymbol{h}_{n_k})}{\|\Theta'_{n_k}(\boldsymbol{h}_{n_k})\|^2}\Theta'_{n_k}(\boldsymbol{h}_{n_k})^T(\boldsymbol{h}_{n_k} - \boldsymbol{f}), \qquad (28)
\end{aligned}$$

where (11) was used for $P_{B_{\ell_1}[\boldsymbol{w}_{n_k},\delta]}$ in order to derive the previous inequality. By the definition of the subgradient, we have that $\Theta'_{n_k}(\boldsymbol{h}_{n_k})^T(\boldsymbol{f} - \boldsymbol{h}_{n_k}) + \Theta_{n_k}(\boldsymbol{h}_{n_k}) \leq \Theta_{n_k}(\boldsymbol{f}) = 0$. If we merge this into (28), we obtain the following:

$$\begin{aligned}
\|\boldsymbol{h}_{n_k+1} - \boldsymbol{f}\|^2 &\leq \|\boldsymbol{h}_{n_k} - \boldsymbol{f}\|^2 + \lambda_{n_k}^2\frac{\Theta_{n_k}^2(\boldsymbol{h}_{n_k})}{\|\Theta'_{n_k}(\boldsymbol{h}_{n_k})\|^2} - 2\lambda_{n_k}\frac{\Theta_{n_k}^2(\boldsymbol{h}_{n_k})}{\|\Theta'_{n_k}(\boldsymbol{h}_{n_k})\|^2} \\
&= \|\boldsymbol{h}_{n_k} - \boldsymbol{f}\|^2 - \lambda_{n_k}(2 - \lambda_{n_k})\frac{\Theta_{n_k}^2(\boldsymbol{h}_{n_k})}{\|\Theta'_{n_k}(\boldsymbol{h}_{n_k})\|^2}.
\end{aligned}$$

This, in turn, implies that

$$\forall k \geq k_0, \quad 0 < (\epsilon''\delta)^2 \leq \lambda_{n_k}(2 - \lambda_{n_k})\frac{\Theta_{n_k}^2(\boldsymbol{h}_{n_k})}{\|\Theta'_{n_k}(\boldsymbol{h}_{n_k})\|^2} \leq \|\boldsymbol{h}_{n_k} - \boldsymbol{f}\|^2 - \|\boldsymbol{h}_{n_k+1} - \boldsymbol{f}\|^2. \qquad (29)$$

However, as we have already shown before, $(\|\boldsymbol{h}_n - \boldsymbol{f}\|)_{n\in\mathbb{Z}_{\geq 0}}$ is convergent, and hence it is a Cauchy sequence. This implies that $\lim_{k\to\infty}(\|\boldsymbol{h}_{n_k} - \boldsymbol{f}\|^2 - \|\boldsymbol{h}_{n_k+1} - \boldsymbol{f}\|^2) = 0$, which apparently contradicts (29). In other words, (27) holds true.

Notice, now, that for all those $n \in \mathbb{Z}_{\geq 0}$ such that $\Theta'_n(\boldsymbol{h}_n) \neq 0$, we have by Lemma 6 that

$$\Theta_n(\boldsymbol{h}_n) = \|\Theta'_n(\boldsymbol{h}_n)\|\frac{\Theta_n(\boldsymbol{h}_n)}{\|\Theta'_n(\boldsymbol{h}_n)\|} \leq \frac{\Theta_n(\boldsymbol{h}_n)}{\|\Theta'_n(\boldsymbol{h}_n)\|}. \qquad (30)$$

Notice, also, here that for all those $n \geq z_0$ such that $\Theta'_n(\boldsymbol{h}_n) = 0$, and by making use of the well-known fact $\boldsymbol{0} \in \partial\Theta_n(\boldsymbol{h}_n) \Leftrightarrow \boldsymbol{h}_n \in \arg\min\{\Theta_n(\boldsymbol{y}) : \ \boldsymbol{y} \in \mathbb{R}^L\}$, that $\Theta_n(\boldsymbol{h}_n) = 0$. Take $\lim_{n\to\infty}$ on both sides of (30), and use (27) to establish our original claim.

Let now Assumption 1 holds true. Then we show that there exists a $D > 0$ such that $\forall n \in \mathbb{Z}_{\geq 0}$, $L_n \leq D$. Notice, that $\forall n \geq z_0$, $\forall j \in \mathcal{I}_n$, $\forall \boldsymbol{f} \in \Omega$,

$$\begin{aligned}
d(\boldsymbol{h}_n, S_j[\epsilon]) = \|\boldsymbol{h}_n - P_{S_j[\epsilon]}(\boldsymbol{h}_n)\| &\leq \|\boldsymbol{h}_n - \boldsymbol{f}\| + \|\boldsymbol{f} - P_{S_j[\epsilon]}(\boldsymbol{h}_n)\| \\
&\leq 2\|\boldsymbol{h}_n - \boldsymbol{f}\| \leq 2\|\boldsymbol{h}_{z_0} - \boldsymbol{f}\|,
\end{aligned}$$

where we have used (11) and the monotonicity of the sequence $(\|\boldsymbol{h}_n - \boldsymbol{f}\|)_{n\geq z_0}$. Then, by the definition of L_n,

$$\forall n \geq z_0, \quad L_n = \sum_{j\in\mathcal{I}_n} \omega_j^{(n)} d(\boldsymbol{h}_n, S_j[\epsilon]) \leq 2 \sum_{j\in\mathcal{I}_n} \omega_j^{(n)} \|\boldsymbol{h}_{z_0} - \boldsymbol{f}\| = 2\|\boldsymbol{h}_{z_0} - \boldsymbol{f}\|.$$

Choose, now, any $D > \max\{2\|\boldsymbol{h}_{z_0} - \boldsymbol{f}\|, L_0, \ldots, L_{z_0-1}\} \geq 0$, and notice that for such a D the claim holds true.

Let Assumptions 1, 2, and 4 hold true. By (20), we observe that

$$\begin{aligned}\frac{D}{\check{\omega}}\Theta_n(\boldsymbol{h}_n) &= \frac{D}{\check{\omega}} \sum_{j\in\mathcal{I}_n} \frac{\omega_j^{(n)} d^2(\boldsymbol{h}_n, S_j[\epsilon])}{L_n} \geq \frac{D}{\check{\omega}} \sum_{j\in\mathcal{I}_n} \frac{\omega_j^{(n)} d^2(\boldsymbol{h}_n, S_j[\epsilon])}{D} \\ &\geq \frac{D}{\check{\omega}} \frac{\check{\omega}}{D} \sum_{j\in\mathcal{I}_n} d^2(\boldsymbol{h}_n, S_j[\epsilon]) \geq \max\{d^2(\boldsymbol{h}_n, S_j[\epsilon]) : \ j \in \mathcal{I}_n\}.\end{aligned}$$

Hence, if we take $\lim_{n\to\infty}$ on both sides of the previous inequality, we obtain $\lim_{n\to\infty} \max\{d(\boldsymbol{h}_n, S_j[\epsilon]) : j \in \mathcal{I}_n\} = 0$.

Recall, now, (8) and easily verify that $\forall j \in \mathcal{J}_n \setminus \mathcal{I}_n$, $d(\boldsymbol{h}_n, S_j[\epsilon]) = 0$. Hence, $\forall n \in \mathbb{Z}_{\geq 0}$, $\max\{d(\boldsymbol{h}_n, S_j[\epsilon]) : j \in \mathcal{I}_n\} = \max\{d(\boldsymbol{h}_n, S_j[\epsilon]) : j \in \mathcal{J}_n\}$. This and the previous discussion establish Theorem 2.2.

3) Here we establish Theorem 2.3. Let Assumptions 1 and 2 hold true. We utilize first (11) and then (24) in order to obtain the following: $\forall \boldsymbol{f} \in \Omega$,

$$\begin{aligned}&\|(I - P_{B_{\ell_1}[\boldsymbol{w}_n,\delta]})(T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n))\|^2 \leq \|T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n) - \boldsymbol{f}\|^2 - \|P_{B_{\ell_1}[\boldsymbol{w}_n,\delta]} T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n) - \boldsymbol{f}\|^2 \\ &= \|T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n) - \boldsymbol{f}\|^2 - \|\boldsymbol{h}_{n+1} - \boldsymbol{f}\|^2 \\ &\leq \|\boldsymbol{h}_n - \boldsymbol{f}\|^2 - \frac{2-\lambda_n}{\lambda_n}\|\boldsymbol{h}_n - T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n)\|^2 - \|\boldsymbol{h}_{n+1} - \boldsymbol{f}\|^2 \leq \|\boldsymbol{h}_n - \boldsymbol{f}\|^2 - \|\boldsymbol{h}_{n+1} - \boldsymbol{f}\|^2.\end{aligned}$$

Take $\lim_{n\to\infty}$ on both sides of this inequality and recall that the sequence $(\|\boldsymbol{h}_n - \boldsymbol{f}\|)_{n\in\mathbb{Z}_{\geq 0}}$ is convergent, and thus Cauchy, in order to obtain

$$\lim_{n\to\infty} \|(I - P_{B_{\ell_1}[\boldsymbol{w}_n,\delta]})(T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n))\| = \lim_{n\to\infty} d(T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n), B_{\ell_1}[\boldsymbol{w}_n, \delta]) = 0. \tag{31}$$

Moreover, notice that for all such $n \geq z_0$ such that $\boldsymbol{h}_n \notin \operatorname{lev}_{\leq 0}\Theta_n$, by (23) we obtain that

$$\|\boldsymbol{h}_n - T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n)\| = \|\boldsymbol{h}_n - \boldsymbol{h}_n + \lambda_n \frac{\Theta_n(\boldsymbol{h}_n)}{\|\Theta_n'(\boldsymbol{h}_n)\|^2}\Theta_n'(\boldsymbol{h}_n)\| = \lambda_n \frac{\Theta_n(\boldsymbol{h}_n)}{\|\Theta_n'(\boldsymbol{h}_n)\|} \leq 2\frac{\Theta_n(\boldsymbol{h}_n)}{\|\Theta_n'(\boldsymbol{h}_n)\|}.$$

Take $\lim_{n\to\infty}$ on both sides of this inequality, and recall (27) to easily verify that

$$\lim_{n\to\infty} \|\boldsymbol{h}_n - T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n)\| = 0. \tag{32}$$

Notice, now, that $\forall \boldsymbol{f} \in B_{\ell_1}[\boldsymbol{w}_n, \delta]$, the triangle inequality implies that

$$\|\boldsymbol{h}_n - \boldsymbol{f}\| \leq \|\boldsymbol{h}_n - T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n)\| + \|T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n) - \boldsymbol{f}\|.$$

If we take $\inf_{f \in B_{\ell_1}[\boldsymbol{w}_n, \delta]}$ on both sides of the previous inequality, then

$$\forall n \in \mathbb{Z}_{\geq 0}, \quad d(\boldsymbol{h}_n, B_{\ell_1}[\boldsymbol{w}_n, \delta]) \leq \|\boldsymbol{h}_n - T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n)\| + d(T_{\Theta_n}^{(\lambda_n)}(\boldsymbol{h}_n), B_{\ell_1}[\boldsymbol{w}_n, \delta]).$$

Take, now, $\lim_{n\to\infty}$ on both sides of this inequality, and use (31) and (32) to establish Theorem 2.3.

4) Next, let Assumptions 1, 2, and 3 hold true. By (25) notice that $\forall n \geq z_0$, $\forall \boldsymbol{f} \in \Omega$,

$$\frac{\epsilon''}{2}\|\boldsymbol{h}_n - \boldsymbol{h}_{n+1}\|^2 \leq \frac{2-\lambda_n}{2}\|\boldsymbol{h}_n - \boldsymbol{h}_{n+1}\|^2 \leq \|\boldsymbol{h}_n - \boldsymbol{f}\|^2 - \|\boldsymbol{h}_{n+1} - \boldsymbol{f}\|^2.$$

This and Fact 2 suggest that $\exists \tilde{\boldsymbol{h}}_* \in \mathbb{R}^L : \ \lim_{n\to\infty} \boldsymbol{h}_n = \tilde{\boldsymbol{h}}_*$.

Now, in order to establish Theorem 2.4, let Assumptions 1, 2, 3, and 4 hold true. Notice that the existence of $\tilde{\boldsymbol{h}}_*$ is guaranteed by the previously proved claim. To prove Theorem 2.4, we will use contradiction. In other words, assume that $\tilde{\boldsymbol{h}}_* \notin \overline{\liminf_{n\to\infty}} S_n[\epsilon]$. This clearly implies that $\tilde{\boldsymbol{h}}_* \notin \liminf_{n\to\infty} S_n[\epsilon]$. Note that the set $\overline{\liminf_{n\to\infty}} S_n[\epsilon]$ is convex. This comes from the fact that $S_n[\epsilon]$ and $\bigcap_{m\geq n} S_m[\epsilon]$ are convex, and that $\bigcap_{m\geq n} S_m[\epsilon] \subset \bigcap_{m\geq n+1} S_m[\epsilon]$, $\forall n \in \mathbb{Z}_{\geq 0}$.

Since by our initial assumption $\operatorname{int} \bigcap_{n\geq z_0} S_n[\epsilon] \neq \emptyset$, we can always find an $\mathring{\boldsymbol{f}}$ and a $\rho > 0$ such that $B(\mathring{\boldsymbol{f}}, \rho) \subset \bigcap_{n\geq z_0} S_n[\epsilon]$. Hence,

$$\forall n \geq z_0, \quad B(\mathring{\boldsymbol{f}}, \rho) \subset S_n[\epsilon]. \tag{33}$$

Notice, here, that $\mathring{\boldsymbol{f}} \in \bigcap_{n\geq z_0} S_n[\epsilon] \subset \bigcup_{n\in\mathbb{Z}_{\geq 0}} \bigcap_{m\geq n} S_m[\epsilon] =: \liminf_{n\to\infty} S_n[\epsilon] \subset \overline{\liminf_{n\to\infty}} S_n[\epsilon]$. Using this, our initial assumption on $\tilde{\boldsymbol{h}}_*$, and the fact that $\overline{\liminf_{n\to\infty}} S_n[\epsilon]$ is closed and convex, then we can always find a $t \in (0,1)$ such that $\boldsymbol{f}_t := \mathring{\boldsymbol{f}} + t(\tilde{\boldsymbol{h}}_* - \mathring{\boldsymbol{f}}) \notin \overline{\liminf_{n\to\infty}} S_n[\epsilon]$. This implies, by the definition of $\liminf_{n\to\infty} S_n[\epsilon]$, that

$$\forall n \geq z_0, \quad \boldsymbol{f}_t \notin \bigcap_{m\geq n} S_m[\epsilon]. \tag{34}$$

Now, since $\lim_{n\to\infty} \boldsymbol{h}_n = \tilde{\boldsymbol{h}}_*$, there exists a $z_1 \in \mathbb{Z}_{\geq 0}$ such that $\forall n \geq z_1$, $\|\tilde{\boldsymbol{h}}_* - \boldsymbol{h}_n\| < \frac{\rho(1-t)}{2t}$. If we set n equal to $\max\{z_0, z_1\}$ in (33) and (34), then we readily verify that $\exists n_0 \in \mathbb{Z}_{\geq 0}$ such that $n_0 \geq \max\{z_0, z_1\}$, $\boldsymbol{f}_t \notin S_{n_0}[\epsilon]$, and $B(\mathring{\boldsymbol{f}}, \rho) \subset S_{n_0}[\epsilon]$. Hence, Fact 3 suggests that $d(\tilde{\boldsymbol{h}}_*, S_{n_0}[\epsilon]) > \frac{\rho(1-t)}{t}$.

Using the triangle inequality $\|\tilde{\boldsymbol{h}}_* - \boldsymbol{f}\| \leq \|\tilde{\boldsymbol{h}}_* - \boldsymbol{h}_{n_0}\| + \|\boldsymbol{h}_{n_0} - \boldsymbol{f}\|$, $\forall \boldsymbol{f} \in S_{n_0}[\epsilon]$, we obtain the following: $d(\boldsymbol{h}_{n_0}, S_{n_0}[\epsilon]) \geq d(\tilde{\boldsymbol{h}}_*, S_{n_0}[\epsilon]) - \|\tilde{\boldsymbol{h}}_* - \boldsymbol{h}_{n_0}\| > \frac{\rho(1-t)}{t} - \frac{\rho(1-t)}{2t} = \frac{\rho(1-t)}{2t} =: \gamma > 0$. This clearly implies that $\max\{d(\boldsymbol{h}_{n_0}, S_j[\epsilon]) : \ j \in \mathcal{J}_{n_0}\} \geq \gamma > 0$. Set, now, n equal to $n_0 + 1$

in (33) and (34), and verify, as we did previously, that $\exists n_1 \in \mathbb{Z}_{\geq 0}$ such that $\max\{d(\boldsymbol{h}_{n_1}, S_j[\epsilon]) : j \in \mathcal{J}_{n_1}\} \geq \gamma > 0$. Going on this way, we can construct a sequence $(\boldsymbol{h}_{n_k})_{k\in\mathbb{Z}_{\geq 0}}$ such that $\forall k \in \mathbb{Z}_{\geq 0}$, $\max\{d(\boldsymbol{h}_{n_k}, S_j[\epsilon]) : \ j \in \mathcal{J}_{n_k}\} \geq \gamma > 0$. However, this contradicts Theorem 2.2. Since we have reached a contradiction, this means that our initial assumption is wrong, and that $\tilde{\boldsymbol{h}}_* \in \overline{\liminf_{n\to\infty}} S_n[\epsilon]$.

If we follow exactly the same procedure, as we did before, for the case of the sequence of sets $(B_{\ell_1}[\boldsymbol{w}_n, \delta])_{n\in\mathbb{Z}_{\geq 0}}$, then we obtain also $\tilde{\boldsymbol{h}}_* \in \overline{\liminf_{n\to\infty}} B_{\ell_1}[\boldsymbol{w}_n, \delta]$.

REFERENCES

[1] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inform. Theory*, vol. 52, no. 2, pp. 489–509, Feb 2006.

[2] David L. Donoho, "Compressed sensing," *IEEE Trans. Inform. Theory*, vol. 52, pp. 1289–1306, 2006.

[3] E. Candès, "Compressive sampling," in *Proceedings of ICM*, 2006, vol. 3, pp. 1433–1452.

[4] E. J. Candès and T. Tao, "Decoding by linear programming," *IEEE Trans. Inform. Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[5] R. G. Baraniuk, "Compressive sensing [lecture notes]," *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 118–121, August 2007.

[6] Y. Chen, Y. Gu, and A. O. Hero, "Sparse LMS for system identification," in *Proceedings of the IEEE ICASSP*, 2009, pp. 3125–3128.

[7] D. Angelosante and G. B. Giannakis, "RLS-weighted Lasso for adaptive estimation of sparse signals," in *Proceedings of the IEEE ICASSP*, 2009, pp. 3245–3248.

[8] B. Babadi, N. Kalouptsidis, and V. Tarokh, "Asympotic achievability of the Cramer-Rao bound for noisy compressive sampling," *IEEE Trans. Signal Processing*, vol. 57, no. 3, pp. 1233–1236, 2009.

[9] Y. Murakami, M. Yamagishi, M. Yukawa, and I. Yamada, "A sparse adaptive filtering using time-varying soft-thresholding techniques," in *Proceedings of the IEEE ICASSP*, Dallas: USA, March 2010, pp. 3734–3737.

[10] K. Slavakis, Y. Kopsinis, and S. Theodoridis, "Adaptive algorithm for sparse system identification using projections onto weighted ℓ_1 balls," in *Proceedings of IEEE ICASSP*, Dallas: USA, March 2010, pp. 3742–3745.

[11] D. Angelosante, J. A. Bazerque, and G. B. Giannakis, "Online adaptive estimation of sparse signals: Where RLS meets the ℓ_1 -norm," *Signal Processing, IEEE Transactions on*, vol. 58, no. 7, pp. 3436 –3447, July 2010.

[12] P. L. Combettes, "The foundations of set theoretic estimation," *Proc. IEEE*, vol. 81, no. 2, pp. 182–208, 1993.

[13] I. Yamada and N. Ogura, "Adaptive Projected Subgradient Method for asymptotic minimization of sequence of nonnegative convex functions," *Numerical Functional Analysis and Optimization*, vol. 25, no. 7&8, pp. 593–617, 2004.

[14] K. Slavakis, I. Yamada, and N. Ogura, "The Adaptive Projected Subgradient Method over the fixed point set of strongly attracting nonexpansive mappings," *Numerical Functional Analysis and Optimization*, vol. 27, no. 7&8, pp. 905–930, 2006.

[15] K. Slavakis and I. Yamada, "Asymptotic minimization of sequences of loss functions constrained by families of quasi-nonexpansive mappings and its application to online learning," submitted for publication (preprint: http://arxiv.org/abs/1008.5231), Sept. 2010.

[16] H. Rauhut, "Circulant and Toeplitz matrices in compressed sensing," in *Proceedings of Signal Processing with Adaptive Sparse Structured Representations (SPARS) Workshop*, 2009.

[17] W. Bajwa, J. Haupt, G. Raz, S. Wright, and R. Nowak, "Toeplitz-structured compressed sensing matrices," in *Proceedings of IEEE Statistical Signal Processing (SSP) Workshop*, Washington, DC, USA, 2007, pp. 294–298.

[18] W. Bajwa, J. Haupt, A. Sayeed, and R. Nowak, "Compressed channel sensing: a new approach to estimating sparse multipath channels," Proc. IEEE, Jun 2010.

[19] S. Theodoridis, K. Slavakis, and I. Yamada, "Adaptive learning in a world of projections: a unifying framework for linear and nonlinear classification and regression tasks," Accepted for publication in the IEEE Signal Processing Magazine.

[20] E. Candès, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *J. Fourier Anal. Appl.*, vol. 14, pp. 877–905, 2008.

[21] L. M. Bregman, "The method of successive projections for finding a common point of convex sets," *Soviet Math. Dokl.*, vol. 6, pp. 688–692, 1965.

[22] L. G. Gubin, B. T. Polyak, and E. V. Raik, "The method of projections for finding the common point of convex sets," *USSR Comput. Math. Phys.*, vol. 7, pp. 1–24, 1967.

[23] H. H. Bauschke and J. M. Borwein, "On projection algorithms for solving convex feasibility problems," *SIAM Review*, vol. 38, no. 3, pp. 367–426, Sept. 1996.

[24] H. Stark and Y. Yang, *Vector Space Projections: A Numerical Approach to Signal and Image Processing, Neural Nets, and Optics*, John Wiley & Sons, New York, 1998.

[25] K. Slavakis, S. Theodoridis, and I. Yamada, "Adaptive constrained learning in Reproducing Kernel Hilbert Spaces: the robust beamforming case," *IEEE Trans. Signal Processing*, vol. 57, no. 12, pp. 4744–4764, Dec. 2009.

[26] K. Slavakis, S. Theodoridis, and I. Yamada, "Online kernel-based classification using adaptive projection algorithms," *IEEE Trans. Signal Processing*, vol. 56, no. 7, pp. 2781–2796, 2008.

[27] I. Yamada, K. Slavakis, and K. Yamada, "An efficient robust adaptive filtering algorithm based on parallel subgradient projection techniques," *IEEE Trans. Signal Processing*, vol. 50, no. 5, pp. 1091–1101, 2002.

[28] J. Duchi, S. S-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the ℓ_1-ball for learning in high dimensions," in *Proceedings of International Conference on Machine Learning (ICML)*, 2008, pp. 272–279.

[29] I. Daubechies, M. Fornasier, and I. Loris, "Accelerated projected gradient method for linear inverse problems with sparsity constraints," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 764–792, December 2008.

[30] H. Zou, "The adaptive lasso and its oracle properties," *Journal of the American Statistical Association*, vol. 101, pp. 1418–1429, December 2006.

[31] E. van den Berg and M. P. Friedlander, "Probing the pareto frontier for basis pursuit solutions," *SIAM Journal on Scientific Computing*, vol. 31, no. 2, pp. 890–912, 2008.

[32] E. van den Berg and M. P. Friedlander, "SPGL1: A solver for large-scale sparse reconstruction," June 2007, http://www.cs.ubc.ca/labs/scl/spgl1.

[33] J. F. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optim. Meth. Softw.*, vol. 11–12, pp. 625–653, Aug. 1999.

[34] A. H. Sayed, *Fundamentals of Adaptive Filtering*, John Wiley & Sons, New Jersey, 2003.

[35] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, New Jersey, 3rd edition, 1996.

[36] Y. Kopsinis, K. Slavakis, and S. Theodoridis, "Online sparse system identification and signal reconstruction using projections onto weighted ℓ_1 balls," *arXiv:1004.3040v1 [cs.IT]*, 2010.

[37] J-B. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms*, vol. 1, Springer-Verlag, Berlin, 1993.

[38] H. H. Bauschke and P. L. Combettes, "A weak-to-strong convergence principle for Fejér-monotone methods in Hilbert spaces," *Mathematics of Operations Research*, vol. 26, no. 2, pp. 248–264, May 2001.

PLACE PHOTO HERE

Yannis Kopsinis Yannis Kopsinis received the BSc degree from the department of Informatics and Telecommunications, University of Athens, Greece, in 1998 and his PhD degree in 2003 from the same department. From Jan. 2004 to Dec. 2005 he has been a research fellow with the Institute for Digital Communications, School of Engineering and Electronics, the University of Edinburgh, UK. From Jan. 2006 to Sep. 2009 he was a senior researcher in the same department. He is currently a honorary research fellow in the Dept. of Infromatics and Telecommunications, University of Athens, Greece. His current research interests include adaptive signal processing, time-frequency analysis and online compressed sensing.

PLACE PHOTO HERE

Konstantinos Slavakis Konstantinos Slavakis received the M.E. and Ph.D. degrees in electrical and electronic engineering from Tokyo Institute of Technology (TokyoTech), Tokyo, Japan, in 1999 and 2002, respectively. From 1996 to 2002, he was a recipient of the Japanese Government (MEXT) Scholarship. For the period of 2004 to 2006, he was with TokyoTech as a Japan Society for the Promotion of Science (JSPS) PostDoctoral Fellow, and from 2006 till 2007 he was a PostDoctoral Fellow in the Department of Informatics and Telecommunications, University of Athens, Greece. Since September 2007, he is an Assistant Professor for the Department of Telecommunications Science and Technology, University of Peloponnese, Tripolis, Greece. He serves as an Associate Editor of the IEEE Transactions on Signal Processing. His research interests include the applications of convex analysis and computational algebraic geometry to signal processing, machine learning, array, and multidimensional systems problems.

PLACE
PHOTO
HERE

Sergios Theodoridis Sergios Theodoridis is currently Professor of Signal Processing and Communications in the Department of Informatics and Telecommunications of the University of Athens. His research interests lie in the areas of Adaptive Algorithms and Communications, Machine Learning and Pattern Recognition, Signal Processing for Audio Processing and Retrieval. He is the co-editor of the book Efficient Algorithms for Signal Processing and System Identification, Prentice Hall 1993, the co-author of the best selling book Pattern Recognition, Academic Press, 4th ed. 2008, the co-author of the book Introduction to Pattern Recognition: A MATLAB Approach, Academic Press, 2009, and the co-author of three books in Greek, two of them for the Greek Open University.

He is the co-author of six papers that have received best paper awards including the 2009 IEEE Computational Intelligence Society Transactions on Neural Networks Outstanding paper Award. He has served as an IEEE Signal Processing Society Distinguished Lecturer.

He was the general chairman of EUSIPCO-98, the Technical Program co-chair for ISCAS-2006 and co-chairman and co-founder of CIP-2008 and co-chairman of CIP-2010. He has served as President of the European Association for Signal Processing (EURASIP) and as member of the Board of Governors for the IEEE CAS Society. He currently serves as member of the Board of Governors (Member-at-Large) of the IEEE SP Society.

He has served as a member of the Greek National Council for Research and Technology and he was Chairman of the SP advisory committee for the Edinburgh Research Partnership (ERP). He has served as vice chairman of the Greek Pedagogical Institute and he was for four years member of the Board of Directors of COSMOTE (the Greek mobile phone operating company). He is Fellow of IET, a Corresponding Fellow of RSE and a Fellow of IEEE.