Does a general structure exist for adaptation/learning algorithms?

Ioan Doré Landau, Tudor-Bogdan Airimitoaie

Does a general structure exist for adaptation/learning algorithms?

Ioan Doré Landau, Tudor-Bogdan Airimitoaie

Abstract— **There are many parameter adaptation/learning algorithms (PALA) used in adaptive control, system identification and neural networks (Nesterov, Conjugate gradients, Momentum back propagation, Averaged gradient, Integral+proportional+derivative, ...). For most of these algorithms unfortunately there are no results available for the choice of the various coefficients (weights) allowing to guarantee the stability of the parameter estimator for any value of the learning rate and for any initial conditions. All these algorithms are in fact particular cases of a general structure for the PALA which is introduced in this paper. This structure is characterized by the presence of an embedded ARMA (Auto Regressive Moving Average) filter. Taking into account the inherent feedback structure of these adaptation/learning algorithms, the passivity approach is used for addressing the stability issue. Conditions which will assure the stability of this general structure will be provided and then particularized for the specific algorithms described in the paper. The impact of the MA and AR terms of the embedded filter upon the performance of the algorithms will be emphasized through simulation.**

I. INTRODUCTION

With the booming of neural networks [1], [2], there was an explosion of the number of adaptation/learning algorithms which have been proposed. Some of these algorithms are inspired from optimization techniques [3], [4]. In most of the cases only a qualitative analysis of these "new" algorithms is provided. The field becomes a kind of "fiddler's paradise". The papers [5] and [6] give a comprehensive review of current used algorithms. For most of these algorithms, unfortunately, there are no results available for the choice of the various coefficients (weights) allowing to guarantee the asymptotic stability of the parameter estimator for any value of the learning rate and for any initial conditions of the estimated parameters.

In fact, it can be shown that one has to deal with a dynamic system with a feedback structure. This approach has been developed in the field of adaptive control. See for example [7], [8]. The paper will show that many adaptation/learning algorithms (maybe all?) are particular forms of a general structure for PALA characterized by the presence of an embedded ARMA (poles-zeros) filter acting on the partial gradient of a criterion to be minimized with respect to the parameters to be tuned. Taking into account the inherent feedback structure of the PALA (parameter aldaptation/learning algorithms) and using passivity arguments, an answer can be provided to the question of stability of the parameter estimator for any value of the adaptation gain/learning rate and any initial conditions. The basic answer is that the embedded filter should be characterized by a positive real discrete time transfer function. The contributions of the paper can be summarized as follows:

- A general form for the PALA algorithms is introduced and conditions for assuring stability of the algorithms for any positive value of the adaptation gain/learning rate are provided.
- A review of a number of PALA from this unified perspective is provided.
- A discrete time version of the I+P+D (integral + proportional + derivative) algorithm is presented.
- A new PALA algorithm characterized by a 2nd order ARIMA filter acting on the gradient is introduced.
- Illustration of the effect of the various coefficients upon the performance of the algorithms is provided by simulation.

II. REVISITING THE GRADIENT ALGORITHM – FEEDBACK INTERPRETATION AND STABILITY ISSUES

A. Basic Gradient Algorithm

The aim of the gradient parameter adaptation/learning algorithm is to drive the parameters of an adjustable model in order to minimize a quadratic criterion in terms of the prediction error (difference between real data and the output of the model used for prediction). To formalize the problem, following [5] and [6], one considers the discrete-time model described by:

$$y(t+1) = -a_1 y(t) - a_2 y(t-1) - \ldots + b_1 u(t) + b_2 u(t-2) + \ldots = \theta^T \phi(t), \quad (1)$$

where the unknown parameters a_i and b_i form the components of the *parameter vector* θ:

$$\theta^T = [a_1, a_2, \ldots, a_{n_a}, b_1, b_2, \ldots, b_{n_B}] \quad (2)$$

and

$$\phi^T(t) = [-y(t), -y(t-1), \ldots, u(t), u(t-1), \ldots] \quad (3)$$

is the *measurement vector*.[1] The adjustable prediction model will be described in this case by:

$$\hat{y}^0(t+1) = \hat{y}[(t+1)|\hat{\theta}(t)] = \hat{\theta}^T(t)\phi(t), \quad (4)$$

[1]Ioan Doré Landau, is with the Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France ioan-dore.landau@gipsa-lab.grenoble-inp.fr,

[2]Tudor-Bogdan Airimitoaie is with the Univ. Bordeaux, CNRS, Bordeaux INP, IMS, 33405 Talence, France tudor-bogdan.airimitoaie@u-bordeaux.fr

[1] $u(t), y(t) \in R^1, \theta, \phi \in R^n, n = n_a + n_b$, R^n is the real n-dimensional Euclidean space

where $\hat{y}^0(t+1)$ is termed the *a priori* predicted output depending upon the values of the estimated parameter at instant t given by:

$$\hat{\theta}^T(t) = [\hat{a}_1(t), \hat{a}_2(t), \ldots, \hat{a}_{n_A(t)}, \hat{b}_1(t), \hat{b}_2(t), \ldots, \hat{b}_{n_B(t)}]. \quad (5)$$

It is very useful to consider also the *a posteriori* predicted output computed on the basis of the new estimated parameter vector at $t+1$, $\hat{\theta}(t+1)$, which will be available somewhere between $t+1$ and $t+2$. The *a posteriori* predicted output will be given by:

$$\hat{y}(t+1) = \hat{y}[(t+1)|\hat{\theta}(t+1)] = \hat{\theta}^T(t+1)\phi(t). \quad (6)$$

One defines an *a priori* prediction error as:

$$\varepsilon^0(t+1) = y(t+1) - \hat{y}^0(t+1) \quad (7)$$

and an *a posteriori* prediction error as:

$$\varepsilon(t+1) = y(t+1) - \hat{y}(t+1) = (\theta - \hat{\theta}(t+1))^T\phi(t). \quad (8)$$

The objective is to find a recursive parameter adaptation algorithm with memory. The structure of such an algorithm is:

$$\hat{\theta}(t+1) = \hat{\theta}(t) + \Delta\hat{\theta}(t+1) = \hat{\theta}(t) + f[\hat{\theta}(t), \phi(t), \varepsilon^0(t+1)] \quad (9)$$

The correction term $f[\hat{\theta}(t), \phi(t), \varepsilon^0(t+1)]$ must depend solely on the information available at the instant $(t+1)$ when $y(t+1)$ is acquired (last measurement $y(t+1)$, $\hat{\theta}(t)$, and a finite amount of information at times $t, t-1, t-2, \ldots, t-n$). The correction term must enable to minimize the following criterion at each step[2]

$$\min_{\hat{\theta}(t+1)} J(t+1) = [\varepsilon(t+1)]^2 \quad (10)$$

A solution can be provided by the gradient technique. The corresponding PALA will have the form:

$$\hat{\theta}(t+1) = \hat{\theta}(t) - F \bigtriangledown_\theta J(t+1) = \hat{\theta}(t) - F\frac{\partial J(t+1)}{\delta\hat{\theta}(t)}, \quad (11)$$

where $F = \alpha I(\alpha > 0)$ is the matrix adaptation gain/learning rate (I - unitary diagonal matrix) and $\partial J(t+1)/\partial\hat{\theta}(t)$ is the partial gradient of the criterion given in (10) with respect to $\hat{\theta}(t)$. At this stage it is interesting to point out already that this is a dynamic system with input the gradient and output the estimated parameter vector, i.e (11) can be expressed also as:

$$\hat{\theta}(t+1) = H_{PAA}(q^{-1}) \bigtriangledown_\theta J(t+1), \quad (12)$$

where[3] $H_{PAA}(q^{-1})$ is a MIMO diagonal transfer operator having identical terms. All the diagonal terms are identical and are described in this case by:

$$H^{ii}(q^{-1}) = \frac{1}{1-q^{-1}}. \quad (13)$$

[2]Using the criterion $\min_{\hat{\theta}(t)} J(t+1) = [\varepsilon^0(t+1)]^2$ will not allow to guarantee the stability of the PALA for any value of the adaptation gain/learning rate. See [8] for details.

[3]The unit delay operator q^{-1} will be used for describing the system's behavior in the time domain and the complex variable z^{-1} will be used for characterizing the system's behavior in the frequency domain.

The estimated parameter vector $\hat{\theta}$ can be viewed as the output of a discrete time integrator filter whose input is the gradient (or in general a correcting term related to the gradient). Note also that the operator (13) is characterized by a positive real transfer function (it is a passive system).

From Eqs (10) and (11), one obtains (for details see [8]):

$$\hat{\theta}(t+1) = \hat{\theta}(t) + F\phi(t)\varepsilon(t+1) \quad (14)$$

where F is the matrix adaptation gain[4]. The algorithm has memory (for $\varepsilon(t+1) = 0$, $\hat{\theta}(t+1) = \hat{\theta}(t)$). There are two possible choices for the matrix adaptation gain/learning rate: (i) $F = \alpha I$; $\alpha > 0$; (ii) $F > 0$ (positive definite matrices). For the remaining of the paper we will use the option $F = \alpha I$. The term *adaptation gain* or *learning rate* is used for characterizing α.

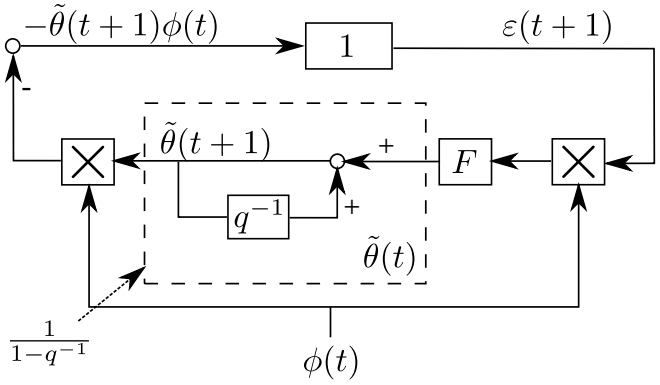

Fig. 1. Feedback structure of gradient adaptation/learning algorithm.

Consider Eq. (14), subtracting θ from both sides of (14) and then multiplying with $\phi(t)^T$) one gets:

$$\phi(t)^T\tilde{\theta}(t+1) = \phi(t)^T\tilde{\theta}(t) + \phi(t)^TF\phi(t)\varepsilon(t+1), \quad (15)$$

where $\tilde{\theta}(t) = \hat{\theta}(t) - \theta$ is the parameter error. Eqs (8) and (15) define a feedback system shown in Fig. 1.

Since it is a feedback structure, stability of the system is a key issue. Using passivity arguments (see [8]) it can be shown that the feedback path is passive and since the feedforward transfer function is 1 (a particular strictly positive real transfer function), the system will guarantee $\lim_{t\to\infty}\varepsilon(t+1) = 0$ for any initial conditions $\theta(0), \varepsilon(0)$ and any value of the adaptation gain $\alpha > 0$ (or any positive definite matrix F). Furthermore, examining the equivalent feedback path one observes that there is an embedded integrator filter which is characterized by a positive real transfer function.

III. A GENERAL FORM FOR ADAPTATION/LEARNING ALGORITHMS

For stability reasons, it is therefore crucial that the equivalent feedback path be passive. However, passivity of the equivalent feedback path can be guaranteed if one replaces

[4]For the effective implementation, $\varepsilon(t+1)$ is given by $\varepsilon(t+1) = \frac{\varepsilon^0(t+1)}{1+\phi^T(t)F\phi(t)}$.

the integrator filter (in fact a multi-input, multi-output filter) by any positive real transfer matrix[5] (of appropriate dimension) with a pole at $z = 1$, in order to have memory, or without a pole at $z = 1$, if we do not want to have memory. For details, see [8]. This allows on one hand to generate an infinite number of adaptation/learning algorithms and on the other hand it allows to analyze adaptation/learning algorithms which have been generated from different points of view. Therefore, one can consider to replace the integrator by a more general passive linear system leading to a PALA of the form ([8])

$$x(t+1) = Ax(t) + B\phi(t)\varepsilon(t+1), \tag{16}$$
$$\hat{\theta}(t+1) = Cx(t) + D\phi(t)\varepsilon(t+1), \tag{17}$$

where $x(t)$ is the state of the passive linear filter and the input is the inverse of the gradient, in our case $\phi(t)\varepsilon(t+1)$. The system $[A, B, C, D]$ is characterized also by the matrix transfer function:

$$H_{PAA}(z) = C(zI - A)^{-1}B + D. \tag{18}$$

The particular case of integral adaptation/learning corresponds to: $A = I$, $B = D = F$, $C = I$.
The algorithm (16) and (17) can also be expressed as:

$$\hat{\theta}(t+1) = H_{PAA}(q^{-1})\phi(t)\varepsilon(t+1). \tag{19}$$

One has the following result:

Theorem 1 *For the system described by Eqs* (1) *through* (8) *using the PALA of Eqs* (16) *and* (17) *or of Eq.* (19) *one has* $\lim_{t\to\infty}\varepsilon(t+1) = 0$ *for any initial conditions* $\theta(0), \varepsilon(0)$ if $H_{PAA}(q^{-1})$ is a positive real transfer matrix[6] with a pole at $z = 1$.

The proof of Theorem 1 uses the results of [8, Theorem 3.1] for proving the passivity of the equivalent feedback path and the results of [8, Theorem 3.2] to conclude upon the stability of the full system.

For the purpose of this paper, it is convenient to particularize $H_{PAA}(q^{-1})$ as a MIMO diagonal transfer operator having identical terms. All the diagonal terms are identical and are described by:

$$H^{ii}(q^{-1}) = \frac{1 + c_1q^{-1} + c_2q^{-2} + .. + c_{n_C}q^{-n_C}}{1 - d_1q^{-1} - d_2q^{-2} - .. - d_{n_D}q^{-n_D}} = \frac{C(q^{-1})}{D(q^{-1})} \tag{20}$$

and the passivity condition of Theorem 1 implies that $H^{ii}(z^{-1})$ should be a positive real transfer function with a pole at $z = 1$ if we want memory. F is the adaptation gain/learning rate which is a positive definite matrix. For the remaining of the paper it will be considered that $F = \alpha I; \alpha > 0$. The explicit form of the algorithm is:

$$\begin{aligned}\hat{\theta}(t+1) &= d_1\hat{\theta}(t) + d_2\hat{\theta}(t-1) + ... + d_{n_D}\hat{\theta}(t-n_D)\\ &+ F[\phi(t)\varepsilon(t+1) + c_1\phi(t-1)\varepsilon(t) + c_2\phi(t-2)\varepsilon(t-1)\\ &+ \ldots + c_{n_C}\phi(t-n_C)\varepsilon(t-n_C+1)].\end{aligned} \tag{21}$$

The algorithm given in Eq. (21) will be termed *Auto Regressive Moving Average (ARMA)* adaptation/learning algorithm and if it has an integrator, it will be termed *Auto Regressive with Integrator Moving Average (ARIMA)* adaptation/learning algorithm. One can see that the current parameter estimates depend upon the previous parameter estimations over a certain horizon (auto regressive) and upon the current and past values of the gradient over a certain horizon (moving average). The *ARIMA* adaptive/learning algorithms are characterized by an embedded filter of the form:

$$\begin{aligned}H^{ii}(q^{-1}) &= \frac{1 + c_1q^{-1} + c_2q^{-2} + .. + c_{n_C}q^{-n_C}}{(1-q^{-1})(1 - d'_1q^{-1} - d'_2q^{-2} - ..d'_{n_{D'}}q^{-n'_D})}\\ &= \frac{C(q^{-1})}{(1-q^{-1})D'(q^{-1})} = \frac{C(q^{-1})}{D(q^{-1})}\end{aligned} \tag{22}$$

where $\frac{C(q^{-1})}{D'(q^{-1})}$ is a *dynamic adaptation gain* and the relation with the coefficients of Eqs (20) and (21) is given by:

$$d_i = (d'_i - d'_{i-1}) \quad ; i = 1, ...n_D; d'_0 = -1,\ d'_{n_D} = 0 \tag{23}$$

To implement the algorithm one needs a computable expression for $\varepsilon(t+1)$. One defines[7]: $\hat{y}^0(t+1) = \hat{\theta}_0^T(t)\phi(t)$ where

$$\begin{aligned}\hat{\theta}_0(t) &= d_1\hat{\theta}(t) + d_2\hat{\theta}(t-1) + \ldots\\ &+ F[c_1\phi(t-1)\varepsilon(t) + c_2\phi(t-2)\varepsilon(t-1) + \ldots]\end{aligned} \tag{24}$$

The *a posteriori* adaptation/prediction error can be written:

$$\begin{aligned}\varepsilon(t+1) &= y(t+1) \pm \hat{\theta}_0^T(t)\phi(t) - \hat{\theta}^T(t+1)\phi(t)\\ &= \varepsilon^0(t+1) - [\hat{\theta}(t+1) - \hat{\theta}_0(t)]^T\phi(t)\\ &= \varepsilon^0(t+1) - \phi(t)^TF\phi(t)\varepsilon(t+1),\end{aligned} \tag{25}$$

which leads to:

$$\varepsilon(t+1) = \frac{\varepsilon^0(t+1)}{1 + \phi^T(t)F\phi(t)}. \tag{26}$$

For small adaptation gains the passivity/stability condition can be relaxed using *averaging* [9]. Using the results of [10], under the hypothesis of an input signal spanning all the frequencies up to half of the sampling frequency, passivity in the average will be assured if the frequency interval where H is not positive real is smaller than the frequency interval where H is positive real.

It will be shown subsequently on one hand, that a number of well known adaptation/learning algorithms are particular cases of the *ARIMA* adaptation/learning algorithm and on the other hand, sufficient conditions for the stability of these algorithms independently of the value of the adaptation gain/learning rate will be provided.

[5]A positive real discrete-time transfer matrix is characterized by the following properties:

1) All elements of $H(z)$ are analytic outside the unit circle (i.e. they do not have poles in $\mid z \mid > 1$).
2) The eventual poles of any element of $H(z)$ on $\mid z \mid = 1$ are simple and the associated residue matrix is a positive semidefinite Hermitian.
3) The matrix $H(z) + H^T(z^{-1})$ is a positive semidefinite Hermitian for all $\mid z \mid = 1$, which are not a pole of $H(z)$.

[6]Or equivalently the system [A,B,C,D] is passive.

[7]$\hat{\theta}_0(t)$ is the best prediction of $\theta(t+1)$ based on the information available at instant t (can be denoted also as $\hat{\theta}_0(t) = \hat{\theta}(t+1/t)$).

IV. A REVIEW OF VARIOUS ADAPTATION/LEARNING ALGORITHMS

A. "Integral + Proportional" Parameter Adaptation Algorithm

A first particularization of the above results is obtained for the integral + proportional adaptation/learning algorithm [11], [8], [12], [13], [14]. The algorithm is in general written under the form:

$$\hat{\theta}_I(t+1) = \hat{\theta}_I(t) + F_I\phi(t)\varepsilon(t+1)\ ;\ F_I > 0 \tag{27}$$

$$\hat{\theta}_P(t+1) = F_p\phi(t)\varepsilon(t+1)\ ; \tag{28}$$

$$\hat{\theta}(t+1) = \hat{\theta}_I(t+1) + \hat{\theta}_P(t+1), \tag{29}$$

where F_I is called the *integral adaptation gain* and F_p *the proportional adaptation gain*. The *a priori* adjustable predictor has the form:

$$\hat{y}^0(t+1) = \hat{\theta}_I^T(t)\phi(t);\ y(t+1) = \hat{\theta}^T(t+1)\phi(t) \tag{30}$$

The *a posteriori* adaptation (prediction) error is given by:

$$\varepsilon(t+1) = \frac{\varepsilon^0(t+1)}{1+\phi(t)^T(F_I+F_P)\phi(t)} \tag{31}$$

The associated state-space representation (16) and (17) of the embedded filter is obtained with: $A = I\ ;\ B = F_I\ ;\ C = I\ ;\ D = F_I+F_P$ which corresponds to an associated transfer matrix:

$$H_{PAA}(z^{-1}) = \frac{1}{1-z^{-1}}F_I + F_P \tag{32}$$

The associated passivity conditions on the matrices F_I and F_p resulting from the use of the *"positive real lemma"* take the form [8]:

$$F_I > 0;\quad F_P = \beta F_I\ ;\ \beta \geq -0.5 \tag{33}$$

For the case of diagonal matrices with identical terms, the embedded transfer operator can be alternatively expressed as Eq. (22).

Proportional + Integral PALA with *positive* proportional gain leads to the improvement of the convergence of the adaptation error. Small *negative* proportional adaptation gain improves in general the convergence of the parameters. This is illustrated in [8, pg. 93], [15], [14] among other references.

B. "Integral+Proportional+Derivative" parameter adaptation algorithm

This algorithm has been introduced in [11] with a continuous time formulation. The corresponding discrete-time structure of the algorithm is as follows:

$$\hat{\theta}(t+1) = \hat{\theta}_I(t+1) + \hat{\theta}_P(t+1) + \hat{\theta}_D(t+1) \tag{34}$$

where $\hat{\theta}_I(t+1)$ and $\hat{\theta}_P(t+1)$ are given by Eqs (27) and (28), respectively, and $\hat{\theta}_D(t+1)$ is given by:

$$\hat{\theta}_D(t+1) = F_D[\phi(t)\varepsilon(t+1) - \phi(t-1)\varepsilon(t)]. \tag{35}$$

For the case of diagonal matrices with identical terms, the embedded transfer operator can be alternatively expressed as:

$$H^{ii}(q^{-1}) = \frac{\alpha_I}{1-q^{-1}} + \alpha_P + \alpha_D(1-q^{-1}), \tag{36}$$

which can be reformulated as Eq. (22). The *a priori* predicted output is given by:

$$\hat{y}^0(t+1) = \hat{\theta}_0^T(t)\phi(t);\ \hat{\theta_0}(t) = \hat{\theta_I}(t) - F_D\phi(t-1)\varepsilon(t)] \tag{37}$$

and the *a posteriori* adaptation (prediction) error is given by:

$$\varepsilon(t+1) = \frac{\varepsilon^0(t+1)}{1+\phi(t)^T(F_I+F_P+F_D)\phi(t)}. \tag{38}$$

C. Averaged gradient algorithms

The basic idea is to use an average of the current and of previous gradients over a certain horizon (see [16], [17]). A general formulation in the present context can be:

$$\hat{\theta}(t+1) = \hat{\theta}(t) + F\sum_{i=0}^{n} c_i\phi(t-i)\varepsilon(t+1-i); c_0 = 1. \tag{39}$$

The associated embedded adaptation filter will be:

$$H(q^{-1}) = \frac{1+c_1q^{-1}+c_2q^{-2}+\dots}{(1-q^{-1})}. \tag{40}$$

Of course the c_i should be chosen such that the transfer function associated to the transfer operator given in Eq. (40) is positive real.

Note that I+P and I+P+D adaptation/learning algorithms (see Eq. (36)) for $F_I = \alpha_I I, F_P = \alpha_P I, F_D = \alpha_D I$ can be viewed as a particular form of this algorithm with $c_1 = \frac{-(\alpha_P+2\alpha_D)}{\alpha_I+\alpha_P+\alpha_D}$, $c_2 = \frac{\alpha_D}{\alpha_I+\alpha_P+\alpha_D}$ and $F = (\alpha_I+\alpha_P+\alpha_D)I = \alpha I$. To compute the *a posteriori* adaptation error, one uses Eqs (21) and (26) for $n_D = 1,\ d_1 = 1,\ n_C = n$.

D. The Nesterov algorithm

The Nesterov algorithm [3], [5] has been developed in the field of optimization in order to improve under certain conditions the convergence rate of the basic gradient algorithm. Based on [5], the Nesterov algorithm can be written in the present context as :

$$\hat{\theta}(t+1) = \rho(t) + \alpha\phi(t)\varepsilon(t+1), \tag{41}$$

$$\rho(t) = \hat{\theta}(t) + \beta[\hat{\theta}(t) - \hat{\theta}(t-1)]. \tag{42}$$

Combining Eqs (41) and (42), one gets:

$$\hat{\theta}(t+1) = (1+\beta)\hat{\theta}(t) - \beta\hat{\theta}(t-1) + \alpha\phi(t)\varepsilon(t+1). \tag{43}$$

This is equivalent to say that $\hat{\theta}(t+1)$ is the output of a MIMO diagonal transfer operator and the diagonal terms are characterized by

$$H^{ii}(q^{-1}) = \frac{\alpha}{1-(1+\beta)q^{-1}+\beta q^{-2}} = \frac{\alpha}{(1-q^{-1})(1-\beta q^{-1})}, \tag{44}$$

whose input is $\phi(t)\varepsilon(t+1)$. This transfer operator has a pole at $z = 1$ assuring the memory of the algorithm. However, in order to lead to a stable algorithm H^{ii} should be a positive real transfer operator. Basic calculus allows to find that the positive real condition (which implies also the stability of the algorithm for any finite learning rate) is $-1 \leq \beta \leq 0.33$ (see [11, pg. 160] for details).

To implement the algorithm, one uses Eqs (24) and (26) with $d_1 = 1+\beta;\ d_2 = -\beta;\ c_i = 0;\ i = 1, 2, ..;\ F = \alpha I$.

E. Momentum back propagation algorithm

This algorithm has been proposed in [18], [19]. Following [6], it can be expressed as:

$$\hat{\theta}(t+1) = \hat{\theta}(t) + m[\hat{\theta}(t) - \hat{\theta}(t-1)] + (1-m)\alpha\phi(t)\varepsilon(t+1), \quad (45)$$

where m is called momentum and it can be rewritten as:

$$\hat{\theta}(t+1) = (1+m)\hat{\theta}(t) - m\hat{\theta}(t-1) + (1-m)\alpha\phi(t)\varepsilon(t+1). \quad (46)$$

Comparing with the Nesterov algorithm given in Eq. (43), it results that the only difference is the term $(1-m)$ multiplying the adaptation gain/learning rate. The equivalent filter is the one of Eq. (44) except that the numerator is $(1-m)\alpha$ instead of α. The same conditions are imposed on m in order to guarantee the passivity of the embedded filter: $-1 \leq m \leq 0.33$. Implementation is similar to the Nesterov algorithm except that $\beta = m$ and $\alpha = (1-m)\alpha$.

F. Conjugate gradient algorithm

Conjugate gradient methods [20], [21], [4] are efficient methods for large scale optimization problems. Following [6], this algorithm can be expressed as follows:

$$\hat{\theta}(t+1) = \hat{\theta}(t) + \alpha d(t), \quad (47)$$

$$d(t) = \beta d(t-1) + \phi(t)\varepsilon(t+1); d(0) = \phi(0)\varepsilon(1). \quad (48)$$

Combining Eqs (47) and (48), one gets:

$$\hat{\theta}(t+1) = (1+\beta)\hat{\theta}(t) - \beta\hat{\theta}(t-1) + \alpha\phi(t)\varepsilon(t+1). \quad (49)$$

Eq. (49) has the same form as the Nesterov algorithm and same passivity/stability condition applies.

V. Second order ARIMA algorithm (new)

The algorithms presented above can be divided into two classes: 1) ARI (Integral adaptation, Nesterov, Momentum back propagation, Conjugate gradient) and 2) IMA (I+P, I+P+D, Averaged gradient).

The general form for the ARIMA adaptation/learning algorithms and the examination of various existing algorithms done above suggests to introduce a new algorithm (which can be viewed as a combination of the Nesterov/Conjugate gradient algorithms with the average gradient/I+P+D type algorithms):

$$\hat{\theta}(t+1) = d_1\hat{\theta}(t) + d_2\hat{\theta}(t-1) + F[\phi(t)\varepsilon(t+1) + c_1\phi(t-1)\varepsilon(t) + c_2\phi(t-2)\varepsilon(t-1)]. \quad (50)$$

Taking

$$d_1 = (1+d_1'); \; d_2 = -d_1', \quad (51)$$

one assures the presence of an integrator. The weights d_1', c_1, c_2 should be chosen such that the transfer function:

$$H(q^{-1}) = \frac{1 + c_1q^{-1} + c_2q^{-2}}{1 - d_1q^{-1} - d_2q^{-2}} \quad (52)$$

be positive real.

This algorithm can be also interpreted as an *Integral + Proportional + Filtered derivative* algorithm, i.e the associated transfer operator has the form

$$H(q^{-1}) = \frac{\alpha_I}{1-q^{-1}} + \alpha_P + \alpha_D\frac{(1-q^{-1})}{(1-d_1'q^{-1})}, \quad (53)$$

with:

$$\alpha_I > 0; \quad \alpha_P > -0.5\alpha_I; \quad \alpha_D > 0; \quad -1 < d_1' < 1. \quad (54)$$

From Eq. (54), one obtains the equivalent coefficients $c_1,\ c_2,\ d_1'$ used in Eq. (22). The expression of the *a posteriori* adaptation/prediction error is given by Eqs (24) and (26) particularized for $n_C = n_D = 2$.

VI. Simulation Results

The second order ARIMA algorithm has been chosen to illustrate the properties of the various PALA algorithms. The system under consideration is characterized by

$$S = \frac{q^{-2} + 0.5q^{-3}}{1 - 1.5q^{-1} + 0.7q^{-2}}, \quad (55)$$

whose input is a PRBS with $N = 255$ samples and its parameters will be estimated.

A. Performance

For a given adaptation gain $\alpha = 0.1$, the performance of the adaptation will be evaluated with respect to the choice of the coefficients $c_1,\ c_2,\ d_1'$. To asses the performance, the following indicators will be used:
The sum of the squared *a posteriori* prediction errors:

$$J_\varepsilon(N) = \sum_{t=0}^{N} \varepsilon^2(t+1). \quad (56)$$

The sum of the parametric distance:

$$J_D(N) = \sum_{0}^{N} D(t+1),$$
$$D(N) = \left\{[\theta - \hat{\theta}(t)]^T[\theta - \hat{\theta}(t)]\right\}^{1/2}. \quad (57)$$

Table I summarizes the performance of the 2nd order ARIMA algorithm and of the various particular cases. The table provides the best performance for each configuration. Fig. 2 shows the evolution of the parametric distance. Clearly the 2nd order ARIMA algorithm provides a significant performance improvement with respect to the various particular cases.

Algorithm	c_1	c_2	d_1'	$J_D(N)$	$J_\varepsilon(N)$
Integral	0	0	0	**46.99**	**13.32**
Conj.Gr/Nest..	0	0	0.5	37.86	12.09
I+D	0	0.99	0	34.41	11.95
I+P	0.667	0	0	40.45	12.45
I+P+D/Av.Gr	−0.05	0.99	0	34.475	11.87
ARIMA 2	−0.5	0.4	0.7	**29.42**	**9.67**

TABLE I

Performance of 2nd order ARIMA algorithm.

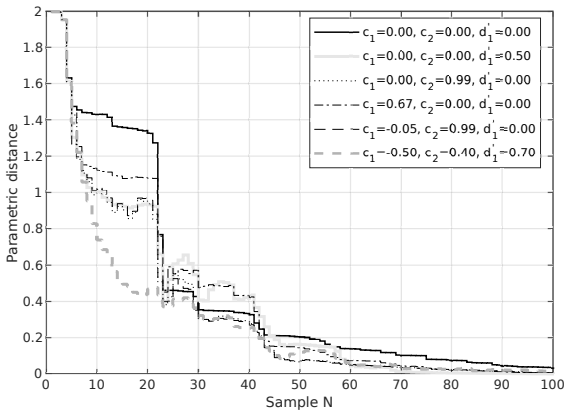

Fig. 2. Evolution of the parametric distance $D(N)$.

B. Stability

Two set of coefficients are considered. As shown in Fig. 3 for the configuration $c_1 = 0.667; \quad c_2 = 0; \quad d_1' = 0$, the corresponding embedded filter is positive real. For the second configuration $c_1 = -0.5; \ c_2 = 0.4; \ d_1' = 0.7$, the embedded filter is not positive real in the region up to $0.17 f_s$ but the positive real condition on the *average* is satisfied for small adaptation gains. Simulations have shown that for the first configuration the algorithm is stable for an adaptation gain of 0.1 and 1000 while for the second case, the algorithm is stable and converges towards the exact values of the parameters for an adaptation gain of 0.1 (see Table I) but the adaptation process is unstable for an adaptation gain/learning rate of 1000.

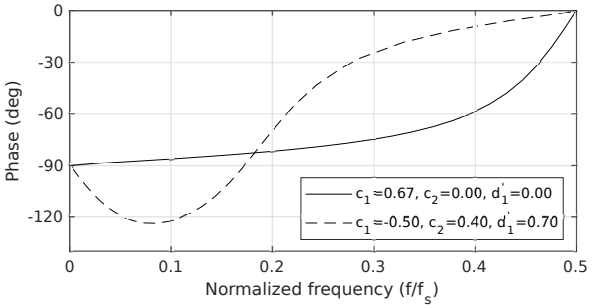

Fig. 3. Phase of the embedded filter for two configurations.

VII. Conclusion

A parameter adaptation/learning algorithm characterized by the presence of an embedded IIR (ARMA) filter has been introduced. The positive realness of the embedded filter transfer function guarantees the stability of the algorithm for any positive value of the adaptation gain/learning rate. The parameter adaptation/learning algorithms reviewed in this paper appear to be particular cases of this algorithm. For some choices of the embedded filter parameters, the performances are improved with respect to the performance of the parameter adaptation/learning algorithm reviewed in the paper.

References

[1] S. Haykin, *Neural Networks*. Prentice Hall, 1999.

[2] K. Narendra and K. Parthasaraty, "Gradient methods for the optimization of dynamical systems containing neural networks ." *IEEE Transactions on Neural Networks*, vol. 2, no. 2, pp. 252–262, March 1991.

[3] Y. Nesterov, "A method for solving a convex programming problem with convergence rate 0(1/k2)," *Soviet Mathematics Docklady*, vol. 27, pp. 372–376, Winter 1983.

[4] R. Fletcher and C. Reeves, "Function minimization by conjugate gradients." *Computer Journal,*, no. 7, pp. 149–154, July 1964.

[5] J. E. Gaudio, T. E. Gibson, A. M. Annaswamy, and M. A. B. E. Lavretsky, "Connections between adaptive control and optimization in machine learning," in *Proceedings of the IEEE CDC Conference,Dec.2019, Nice, France*, 2019, pp. 4563–4568.

[6] I. Livieris and R. Pintelas, "A survey on algorithms for training artificial neural networks," *University of Patras Report*, no. TR08-01, 2014.

[7] P. Ioannou and J. Sun, *Robust Adaptive Control*. Englewood Cliffs, N.J.: Prentice Hall, 1996.

[8] I. D. Landau, R. Lozano, M. M'Saad, and A. Karimi, *Adaptive control*, 2nd ed. London: Springer, 2011.

[9] B. Anderson, R. Bitmead, C. Johnson, P. Kokotovic, R. Kosut, I. Mareels, L. Praly, and B. Riedle, *Stability of adaptive systems*. Cambridge Massachusetts , London, England: The M.I.T Press, 1986.

[10] I. D. Landau, M. Alma, and T.-B. Airimitoaie, "Adaptive feedforward compensation algorithms for active vibration control with mechanical coupling," *Automatica*, vol. 47, no. 10, pp. 2185 – 2196, 2011.

[11] I. Landau, *Adaptive control : the model reference approach*. New York: Marcel Dekker, 1979.

[12] I. D. Landau, "Analyse et synthèse des commandes adaptatives à l'aide d'un modèle par des méthodes d'hyperstabilité," *Automatisme 14, 301-309 (1969).*, vol. 14, pp. 301–309, 1969.

[13] J. W. Gilbart, R. V. Monopoli, and C. F. Price, "Improved convergence and increased flexibility in the design of model reference adaptive control systems," in *Proceedings of IEEE Symposium on Adaptive Processes*, University of Texas, Austin, December 1970.

[14] I. Landau, "Unbiased recursive identification using model reference adaptive techniques," *Automatic Control, IEEE Transactions on*, vol. 21, no. 2, pp. 194 – 202, apr 1976.

[15] T.-B. Airimitoaie and I. D. Landau, "Improving adaptive feedforward vibration compensation by using integral+proportional adaptation," *Automatica*, vol. 49, no. 5, pp. 1501–1505, 2013.

[16] M.Schmidt, "Stochastic average gradient," *University of British Columbia.*, Winter 2018.

[17] S.Pouyanfar, S. Sadiq, Y. Yan, and H. Tian, "A survey on deep learning: Algorithms, techniques, and applications," *ACM Computing Surveys, Vol. 1, No. 1, Article 1. Publication date: January 2017.*, vol. 1, January 2017.

[18] D. Rumelhart, G. Hinton, and R. Williams, *Learning internal representations by error propagation*. Boston, MA: MIT, Cambridge, Massachusetts, US, 1986, pp. 318–362.

[19] R. Jacobs, "Increased rates of convergence through learning rate adaptation." *Neural Networks*, no. 1, pp. 295–307, Winter 1988.

[20] E. Polak and G. Ribière., "Note sur la convergence de methods de directions conjuguees." *Revue Francaise d'Informatique et de Recherche Operationnelle,*, vol. 16, pp. 35–43, 1964.

[21] M. Hestenes and E. Stiefel, "Methods for conjugate gradients for solving linear systems." *Journal of Research of the National Bureau of Standards,*, vol. 49, pp. 409–436, JUly 1952.