


# Large scale nonlinear control system fine-tuning through learning

**Journal Article****Author(s):**

Kosmatopoulos, Elias B.; Kouvelas, Anastasios 

**Publication date:**

2009-06

**Permanent link:**

<https://doi.org/10.3929/ethz-b-000275936>

**Rights / license:**

[In Copyright - Non-Commercial Use Permitted](#)

**Originally published in:**

IEEE Transactions on Neural Networks 20(6), <https://doi.org/10.1109/tnn.2009.2014061>

# Large-Scale Nonlinear Control System Fine-Tuning through Learning

Elias B. Kosmatopoulos and Anastasios Kouvelas

**Abstract**—Despite the continuous advances in the fields of intelligent control and computing, the design and deployment of efficient Large-scale Nonlinear Control Systems (LNCS) requires a tedious fine-tuning of the LNCS parameters prior and during the actual system operation. In the majority of LNCSs the fine-tuning process is performed by experienced personnel based on field observations via experimentation with different combinations of controller parameters, without the use of a systematic approach. The existing adaptive/neural/fuzzy control methodologies cannot be used towards the development of a systematic, automated fine-tuning procedure for general LNCS due to the strict assumptions they impose on the controlled system dynamics; on the other hand, adaptive optimization methodologies fail to guarantee an efficient and safe performance during the fine-tuning process, mainly due to the fact that these methodologies involve the use of random perturbations.

In this paper, we introduce and analyze both by means of mathematical arguments and simulation experiments, a new learning/adaptive algorithm that can provide with convergent, efficient and safe fine-tuning of general LNCS. The proposed algorithm consists of a combination of two different algorithms proposed by the authors in the past [10], [11] and the Incremental-Extreme Learning Machine Neural Networks (IELM-NN). Among the nice properties of the proposed algorithm is that it significantly out-performs the algorithms of [10], [11] as well as other existing adaptive optimization algorithms. Moreover, contrary to the algorithms of [10], [11], the proposed algorithm can operate efficiently in the case where the exogenous system inputs (e.g. disturbances, commands, demand, etc) are unbounded signals.

## Index Terms

**Adaptive Optimization, Simultaneous Perturbation Stochastic Approximation (SPSA), Adaptive Fine-Tuning, Incremental-Extreme Learning Machine Neural Networks, Nonlinear Control Systems**

## I. INTRODUCTION

Despite the continuous advances in the fields of intelligent control and computing, the design and deployment of efficient Large-scale Nonlinear Control Systems (LNCS) remains a particularly challenging problem. This is partly due to the fact that practical control design approaches are often based on simplified models for the system dynamics, leading to LNCS control parameters with suboptimal or even unacceptable performance. On the other hand, the use of more complex models for the design of effective LNCS is often hardly feasible due to the lack of adequate theory and reliable and practicable design approaches. Thus, the use of

simplified models is virtually unavoidable in most complex control system applications. As a result there is a major need for careful and efficient fine-tuning of the LNCS parameters prior to the actual system operation that arises from the use of simplified models and control designs while medium- or long-term unpredictable variations of the system dynamics may call for frequent updates. In the majority of LNCSs the fine-tuning process is performed by experienced personnel based on field observations via experimentation with different combinations of controller parameters, without the use of a systematic approach.

Urban and motorway traffic networks, large chemical processes, sensor networks, advanced computer networks, power networks, Mega City power, water and communication networks are specific examples of large-scale, nonlinear processes that are controlled via corresponding LNCSs and call for a tedious fine-tuning procedure for the calibration of their parameters, whose number may range from dozens to several hundreds or more. Such a fine-tuning procedure may take months, or, in extreme cases, even years, until the LNCS actually reaches the desired performance. In most cases this procedure may lead to an acceptable, but not necessarily optimal, control behavior, while the need for frequent updates calls for a quasi-continuous effort with all related technical and organizational (e.g. change of personnel) risks. In some known cases the outlined procedure has even led to a complete failure, i.e. the use of the LNCS was abandoned after the initial deployment due to the failure of the fine-tuning process to achieve a satisfactory performance (see, e.g. [17]).

Unfortunately, the existing methodologies are unable to provide with a generic, efficient and systematic approach for the automated fine-tuning of LNCS. More precisely:

- Adaptive control as well as a neural network, fuzzy and learning control methodologies are, in general, not applicable to the fine-tuning of control systems that have been developed using non-adaptive techniques. Most importantly, these methodologies suffer from the so-called “loss-of-controllability” problem (see e.g. section 9.7 of [6] and [9]). Roughly speaking, the “loss-of-controllability” problem can be explained as follows: adaptive or adaptive-like techniques make use of an estimation model of the actual system and the adaptive controller is designed as if the estimation model were the actual one; in cases where the estimation model becomes uncontrollable, it is impossible to construct a controller based on this estimation model. To overcome this difficulty most adaptive control designs impose strict assumptions on the controlled system, see e.g. [12], [6];

such assumptions become extraordinarily strict in the case of large-scale systems. On the other hand, removal of these assumptions results in very complicated control designs, which moreover have the disadvantage of very poor transient behavior, see e.g. [9], [23] and the references therein.

- On the other hand, methodologies that are based on Adaptive Optimization (AO) principles (e.g., the Kiefer-Wolfowitz (KW) algorithm [8], [2], the Random Directions Kiefer-Wolfowitz (RDKW) [14], the Simultaneous Perturbation SA (SPSA) [20], [21], etc), although they are applicable to control designs that have been developed using non-adaptive techniques and they do not require any a priori knowledge or assumption on the system dynamics, they do not have any mechanism to incorporate the knowledge captured in the past regarding the dependence of the system performance on the controller parameters and system exogenous inputs; in case where such a dependence is highly nonlinear and complex, the aforementioned algorithms fail to produce any improvement during the fine-tuning process. Such a case is reported in [10] where these techniques failed to produce any fine-tuning improvement for virtually any choice of their design parameters when applied to the fine-tuning of an urban traffic control system. Most importantly, AO methodologies possess the disadvantage of not guaranteeing efficient and safe control behavior during the fine-tuning process: due to the fact that these methodologies involve the use of random perturbations, there is always the possibility that, during the fine-tuning process, the controller exhibits poor or, even worse, unstable performance, see e.g. [11].

Recently, we introduced and analyzed a new family of AO algorithms that can be used towards the development of a generic, efficient and systematic approach for the automated fine-tuning of LNCS [10], [11]. The main attributes of these algorithms may be summarized as follows:

- They are based on Adaptive Optimization (AO) principles and, as a result, they do not require any a priori knowledge or assumption on the system dynamics; moreover, they can be implemented to any type of LNCS regardless of the methodology used for the original design of the LNCS.
- They are robust with respect to exogenous disturbances, noisy measurements, system interactions, component failures, etc.
- They are utterly generic, computationally efficient and straightforward to embed to any type of LNCS, regardless of its size, level of complexity and level of decentralization.
- They incorporate powerful learning and estimation mechanisms which render them adaptable to short-term and long-term variations of system characteristics such as demand long-term variations, system aging etc. Moreover, through these learning and estimation mechanisms, they are capable of incorporating the knowledge captured in the past regarding the dependence of the system

performance on the controller parameters and system exogenous inputs.

- Most importantly, they guarantee a safe and efficient fine-tuning procedure, contrary to other popular AO methods that cannot exclude the possibility of poor or even unstable performance during the automatic fine-tuning process.

The key idea behind the development of these algorithms is to use neural approximators, accompanied with appropriate learning mechanisms for adjusting the neural network weights, in order to approximate (learn) the overall system performance as a function of the controller parameters and the exogenous inputs. In other words, the algorithms of [10], [11] use appropriate neural mechanisms to approximate (learn) the relationship

$$SysPer = J(ConPar, ExInp)$$

where  $J$  is an unknown, highly nonlinear function,  $SysPer$  =System Performance,  $ConPar$  =Controller Parameters and  $ExInp$  are exogenous signals that are either available for measurement (e.g., measurable disturbances, exogenous commands, demand, etc) or can be estimated via appropriate estimation algorithms. The approximation of  $J$  can be then used in one of the following ways:

- (GD) To construct an approximation of the gradient of  $J$  with respect to  $ConPar$ , in which case, standard Gradient-Descent (GD) optimization algorithms can be applied to obtain a – local – minimizer for  $J$ ; it is worth noting that the algorithm of [10] is based on this principle.
- (CP) To check the effect of different  $ConPar$  to  $SysPer$ ; in this case, algorithms like the ones presented and analyzed in [11] can be used which, at each iteration, check the effect of Candidate Perturbed (CP) versions of the current  $ConPar$  and select the “best” among these candidate perturbed versions to be the next  $ConPar$ .

Rigorous mathematical arguments presented in [10], [11] established that the algorithms that are based on the above principles – when used for LNCS fine-tuning – preserve a convergent as well as efficient and safe performance, under the assumption that the system exogenous inputs, disturbances, etc are bounded signals. Unfortunately, in many LNCS applications such an assumption is not realistic; LNCS applications where the exogenous inputs are Gaussian noise signals, LNCS for mitigating the effects of earthquakes in civil structures, LNCS for sensor networks, etc, are few of the many practical LNCS applications where the exogenous inputs may be unbounded<sup>1</sup> signals.

In this paper, we show that an appropriate combination of the principles GD and CP presented above, accompanied with the Incremental-Extreme Learning Machine Neural Network (I-ELM-NN) [4], [5] can guarantee a convergent as well as efficient and safe performance even in the case of unbounded exogenous signals. The proposed algorithm switches – at each iteration – among an algorithm that is based on the GD

<sup>1</sup>The terminology “unbounded signals” is used to denote signals for which it is impossible to *a priori* assess an upper bound of their magnitude.

principle (which is similar to the algorithm proposed in [10]) and a CP-based algorithm; an I-ELM-NN, accompanied with appropriate learning laws, is used to estimate the unknown function  $J$ . The switching among a GD-based and a CP-based algorithms is crucial for the overall algorithm performance, since the CP-based algorithm is needed in order for the neural approximator to be able to efficiently learn the unknown function  $J$ , while the use of the GD-based algorithm is essential for convergence under unbounded exogenous signals.

Simulations experiments performed on a “difficult-to-fine-tune” large scale traffic control system exhibit the efficient performance of the proposed algorithm. More precisely, the proposed algorithm is shown to always guarantee an efficient and safe fine-tuning procedure, while the existing algorithms either totally fail to produce any improvement of the overall control system (this happens in the case of the SPSSA algorithm and the algorithm of [10]) or may lead to poor steady-state performance as it happens in the case of the algorithm of [11].

#### A. Notations

The notation  $\text{vec}(A, B, \dots)$ , where  $A, B, \dots$  are scalars, vectors or matrices, is used to denote a vector whose elements are the entries of  $A, B, C, \dots$  (taken columnwise).  $\mathcal{Z}$  denote the set of nonnegative integers.  $I_n$  denotes the  $n$ -dimensional identity matrix.  $[x]$  denotes the integer part of  $x$ . For a vector  $x \in \mathbb{R}^n$ ,  $|x|$  denotes the Euclidean norm of  $x$  (i.e.  $|x| = \sqrt{x^T x}$ ), while for a matrix  $A \in \mathbb{R}^{n \times 2}$ ,  $|A|$  denotes the induced matrix norm of  $A$ . A function  $f$  is said to be  $C^m$ , where  $m$  is a positive integer, if it is uniformly continuous and its first  $m$  derivatives are uniformly continuous.

A random (or random-like) vector  $x \in \mathbb{R}^n$  is said to be *full-rank*, *zero-mean*,  $\alpha$ -*width*, where  $\alpha$  is a positive real, if  $E[x_i] = 0$ ,  $x \in \{-\alpha, \alpha\}^n$  and, moreover, for any sequence  $x_i, i \in \{1, \dots, n\}$  of such vectors, the following condition is satisfied:

$$\left| [x_1, \dots, x_n]^{-1} \right| \leq \frac{\Xi}{\alpha} \quad (1.1)$$

where  $\Xi$  is a finite positive constant.

Finally, in order to avoid definition of too many variables, constants, etc, we will use the following notation: If  $f_\alpha(x)$  is a function parametrized by the nonnegative parameter  $\alpha$ , we will use the notation  $f_\alpha = O(\alpha)$ , if there exists a strictly increasing scalar, at least  $C^1$ , function  $g$  satisfying  $g(0) = 0, g(\alpha) > 0, \forall \alpha \neq 0$ , such that  $|f_\alpha(x)| \leq g(\alpha), \forall x, \forall \alpha$ . Note that our definition of  $O(\cdot)$  differs from the usual “order of” definition.

#### B. I-ELM Neural Networks

As already mentioned, in this paper we make use of a special family of neural networks, the I-ELM-NNs [4], [5]. For this reason, some preliminaries are needed regarding I-ELM-NNs and their approximation capabilities. More precisely, let  $F : \mathbb{R}^{n_1} \mapsto \mathbb{R}^{n_2}$  be an unknown  $C^m$ ,  $m \geq 1$  function to be approximated and let  $\|\cdot\|_a$  denote one of the following two norms

$$\|F\|_a = \sqrt{\int_{\mathbb{R}^{n_1}} |w(x)F(x)|^2 dx} \text{ or } \|F\|_a = \sup_{x \in \mathbb{R}^{n_1}} |w(x)F(x)|$$

where

$$w(x) = \begin{cases} I_{n_2} & \text{if } |x| \leq a \\ 0 & \text{otherwise} \end{cases}$$

with  $a > 0$ . An I-ELM-NN used for the approximation of  $F$  takes the form

$$\hat{F}(x) = \vartheta^T \phi(x) = \vartheta_1^T \phi_1(x) + \dots + \vartheta_{L_g}^T \phi_{L_g}(x) \quad (1.2)$$

where  $\hat{F}$  denotes the approximation of  $F$ ,  $\vartheta$  denotes the matrix of *parameter estimates*,  $L_g$  denotes the *dimension* of the I-ELM-NN estimator (1.2) and, finally,  $\phi$  denotes the non-linear vector function of I-ELM-NNs *neurons*:

$$\phi_i(x) = S(A_i^T x + b_i) \quad (1.3)$$

where  $S(\cdot)$  is an invertible smooth nonlinear function; finally, the vector  $A_i \in \mathbb{R}^{n_1}$  and the real parameter  $b_i$  are *randomly generated* (with  $A_i, b_i$  being zero-mean).

Let us fix an I-ELM-NN of the form (1.2) with  $L_g$  neurons of the form (1.3) and the constant  $a$ . Then, the *optimal parameter matrix*  $\vartheta^*$  and the *optimal modeling error*  $\nu$  w.r.t.  $L_g, \phi, F$  are defined as follows:

$$\vartheta^* = \mathcal{W}(L_g, \phi, F) = \arg \min_{\vartheta} \|F(x) - \vartheta^T \phi(x)\|_a \quad (1.4)$$

and

$$\nu(x) = \mathcal{N}(L_g, \phi, F)(x) = F(x) - \vartheta^{*T} \phi(x) \quad (1.5)$$

Using the results of [15], [4], it can be seen that I-ELM-NNs satisfy the following property:

- (P1) Consider an I-ELM-NN of the form (1.2) with  $L_g$  neurons of the form (1.3). Then, there exists a scalar function  $\eta : \mathbb{R}^2 \mapsto \mathbb{R}_+$ , satisfying  $\eta(0, \chi) = \eta(\chi, 0) = 0, \forall \chi \in \mathbb{R}_+$ ,  $\eta(\chi, \psi) < \eta(\chi', \psi), \eta(\psi, \chi) \leq \eta(\psi, \chi'), \forall \psi, \chi, \chi' \in \mathbb{R}_+ : \chi < \chi'$ , such that

$$\|\nu\|_a \leq \eta\left(\frac{1}{L_g}, a\right) \quad (1.6)$$

## II. PROBLEM FORMULATION: FINE-TUNING WITH SAFE AND EFFICIENT PERFORMANCE

Let us consider a general LNCS application where the underlying system dynamics are described according to the following nonlinear difference equation

$$z_{t+1} = g(z_t, u_t, d_t), \quad z_0 = z(0) \quad (2.1)$$

where  $z_t, u_t, d_t$  denote the vectors of system states, control inputs, and exogenous signals, respectively,  $t$  denotes the time-index, and  $g(\cdot)$  is a – possibly unknown – sufficiently smooth non-linear vector function, while the control law applied to the system (2.1) is described as follows:

$$u_t = \varpi(\theta, z_t) \quad (2.2)$$

where  $\varpi(\cdot)$  is a known smooth vector function and  $\theta$  is the vector of control parameters. Note that we do not impose any restriction on the form of the controller (2.2).



The performance of the controller (2.2) is evaluated through the following objective function (performance index)

$$\begin{aligned} J(\theta; z_0, D_T) &= \pi_T(z_T) + \sum_{t=0}^{T-1} \pi_t(z_t, u_t) \\ &= \pi_T(z_T) + \sum_{t=0}^{T-1} \pi_t(z_t, \varpi(\theta, z_t)) \end{aligned} \quad (2.3)$$

where  $\pi_t$  are known nonnegative functions,  $T$  denotes the time-horizon over which the control law (2.2) is applied and  $D_T \triangleq [d_0, d_1, \dots, d_{T-1}]$  denotes the time-history of the exogenous signals. By defining  $x = \text{vec}(z_0, D_T)$ , (2.3) may be rewritten as

$$J(\theta, x) = J(\theta; z_0, D_T) \quad (2.4)$$

The problem in hand is to construct an appropriate algorithm which (here  $k$  denotes the current number of fine-tuning experiments, where the duration of each fine-tuning experiment is assumed to be equal to  $T$ ):

- evaluates, at each iteration, the LNCS (2.1)-(2.3) performance for  $\theta = \theta_k$  through the measurement

$$J_k \equiv J(\theta_k, x_k) \quad (2.5)$$

- updates the current controller parameter vector  $\theta_k$  so that it converges as close as possible to one of the local minima  $\theta^*$  of the average value of  $J$  (wrt the random vectors  $x_k$ ), defined according to

$$E \left[ \frac{\partial J}{\partial \theta}(\theta^*, x_k) | \mathcal{G}_k \right] = 0 \quad (2.6)$$

where  $\mathcal{G}_k$  is an appropriately defined  $\sigma$ -field generated – among others – by the past values of the exogenous inputs (see section IV for the formal definition of  $\mathcal{G}_k$ ).

The requirement of convergence of  $\theta_k$  to one of the local minima  $\theta^*$  is not sufficient in most practical situations; additionally to this requirement, the fine-tuning algorithm should be able to provide with *safe and efficient* performance during the fine-tuning process. More precisely, at each iteration of the fine-tuning algorithm the performance index measurement should satisfy

$$J_k \leq J_{k-1} + \epsilon_k \quad (2.7)$$

where  $\epsilon_k$  is an appropriately defined “small” positive term, whose magnitude is proportional to the magnitude and variance of the exogenous inputs. The requirement (2.7) is more than crucial in most practical LNCS fine-tuning applications, since violation of such requirement may cause serious, if not catastrophic, performance, safety, etc, problems. For instance, in the case of fine-tuning of traffic control systems, the violation of requirement (2.7) may lead to serious problems (e.g., complaints, dangerous driving, etc) that may force the traffic operators to cancel the fine-tuning process; similarly, in the case of fine-tuning of LNCS for mechanical structures, the violation of requirement (2.7) may cause the permanent deformation or even the destruction of the structure. It is worth noting, that standard AO methodologies such as the RDKW and SPSA algorithms cannot guarantee that the requirement (2.7) holds during the fine-tuning process mainly due to the use of random perturbations of the controller parameters [11].

### III. THE PROPOSED ALGORITHM

This section presents the details of the proposed algorithm; the mathematical analysis of the convergence and performance properties of the proposed algorithm is presented in the next section. In the sequel, the dimensions of the vectors  $\theta, x_k$  are denoted by  $n_\theta$  and  $n_x$ , respectively.

*Remark 1 (Availability of the estimate  $\bar{x}_k$ ):* The proposed algorithm assumes that an estimate – or prediction –  $\bar{x}_k$  of the vector  $x_k$  is available. In many applications such an assumption is realistic since the entries of  $x_k$  correspond to system states and exogenous inputs which are available for measurement or can be estimated/predicted using appropriate estimation algorithms (see the simulation section for such an example). However, there may be cases where such an assumption is not realistic; in this case it can be readily seen that all the results of the paper are still valid by setting  $\bar{x}_k = 0$ . See also Corollary 1 in section IV.

Let

$$\begin{aligned} \Delta J_k &\equiv \Delta J(\Delta\theta_k, \theta_{k-1}, x_k, x_{k-1}) \\ &= J(\theta_{k-1} + \Delta\theta_k, x_k) - J(\theta_{k-1}, x_{k-1}) \end{aligned} \quad (3.1)$$

where  $\Delta\theta_k = \theta_k - \theta_{k-1}$ . The proposed algorithm makes use of a user-defined collection of  $L_g^k$  I-ELM-NN neurons of the form (note that as already noticed in subsection I.B,  $A_i, b_i$  are randomly generated):

$$\begin{aligned} \phi_i(\theta, x, \bar{\theta}, \bar{x}) &= S(A_i^T \text{vec}(\theta + \bar{\theta}, x) + b_i) \\ &\quad - S(A_i^T \text{vec}(\bar{\theta}, \bar{x}) + b_i) \end{aligned} \quad (3.2)$$

Also, the proposed algorithm makes use of two user-defined positive sequences  $\alpha_k, \beta_k$ ;  $L_g^k, \alpha_k$  and  $\beta_k$  will be specified in the sequel.

Starting with an initial vector  $\theta_0$ , the proposed algorithm initially imposes a full-rank, zero-mean,  $\alpha_1$ -width perturbation  $\Delta\theta_1$  and evaluates the objective function for  $\theta = \theta_0$  and  $\theta = \theta_1$  so as to calculate the first difference  $\Delta J_1 = J(\theta_1 + \theta_0, x_1) - J(\theta_0, x_0)$ . Then, for  $k = 2, 3, \dots$ , the following steps are taking place:

- 1) A collection of  $\phi_i, i \in \{1, \dots, L_g^k\}$  of the form (3.2) is randomly constructed in order to form an I-ELM-NN with  $L_g^k$  neurons, where  $L_g^k$  is calculated as follows:

$$L_g^k = \min \{ \lfloor k/2 \rfloor, \bar{L}_g \} \quad (3.3)$$

with  $\bar{L}_g$  being a user-defined positive integer. The I-ELM-NN associated to the collection of  $\phi_i, i \in \{1, \dots, L_g^k\}$  is used for the estimation of the unknown function  $\Delta J_\ell, \ell \leq k$ :

$$\widehat{\Delta J}(\Delta\theta_\ell, \theta_{\ell-1}, \bar{x}_\ell, \bar{x}_{\ell-1}) = \vartheta^T \phi^{(k)}(\Delta\theta_\ell, \theta_{\ell-1}, \bar{x}_\ell, \bar{x}_{\ell-1}) \quad (3.4)$$

where  $\widehat{\Delta J}(\Delta\theta_\ell, \theta_{\ell-1}, \bar{x}_\ell, \bar{x}_{\ell-1})$  denotes the estimate of  $\Delta J_\ell$  parameterized by the parameter vector  $\vartheta \in \mathbb{R}^{L_g^k}$ ,

$$\phi^{(k)}(\Delta\theta_\ell, \theta_{\ell-1}, \bar{x}_\ell, \bar{x}_{\ell-1}) = \text{vec}(\phi_i(\Delta\theta_\ell, \theta_{\ell-1}, \bar{x}_\ell, \bar{x}_{\ell-1})) \quad (3.5)$$

for  $i \in \{1, \dots, L_g^k\}$  and, as already mentioned,  $\bar{x}_\ell$  denotes an estimate/prediction of  $x_\ell$ .

- 2) **CP-Phase:** If  $k$  is an odd number, then the following are taking place:

- 1)  $n_\theta$  full-rank, zero-mean,  $\alpha_k$ -width vectors  $\Delta\theta_k^{(j)} \in \mathbb{R}^{n_\theta}$ ,  $j \in \{1, \dots, n_\theta\}$  are generated as *candidates* for  $\Delta\theta_k$ .
- 2) For each of the  $\Delta\theta_k^{(j)}$  generated in the previous step as well as for its negative, a *biased least-squares estimate* of  $\Delta J(\chi, \theta_{\ell-1}, x_\ell, x_{\ell-1})$ ,  $\chi \in \mathbb{R}^{n_\theta}$  is generated as follows:

$$\widehat{\Delta J}_{\pm j}(\chi, \theta_{\ell-1}, \bar{x}_\ell, \bar{x}_{\ell-1}) = \vartheta_{\pm j}^\tau \phi^{(k)}(\chi, \theta_{\ell-1}, \bar{x}_\ell, \bar{x}_{\ell-1}) \quad (3.6)$$

where,  $\vartheta_{\pm j}$  is the solution of the following constrained-optimization problem:

$$\begin{aligned} \vartheta_{\pm j} \mapsto \arg \min_{\vartheta} & \frac{1}{2} \sum_{\ell=\ell_k}^{k-1} \left( \Delta J_\ell - \vartheta^\tau \phi_\ell^{(k)} \right)^2 \\ & \text{subject to} \\ & \vartheta^\tau \phi_{\pm j}^{(k)} \leq -\beta_k \end{aligned} \quad (3.7)$$

where  $\phi_\ell^{(k)} = \phi^{(k)}(\Delta\theta_\ell, \theta_{\ell-1}, \bar{x}_\ell, \bar{x}_{\ell-1})$ ,  $\phi_{\pm j}^{(k)} = \phi^{(k)}(\pm \Delta\theta_k^{(j)}, \theta_{k-1}, \bar{x}_k, \bar{x}_{k-1})$  and

$$\ell_k = \max \{k - 2L_g^k - T_h, 0\} \quad (3.8)$$

with  $T_h$  being a user-defined nonnegative integer.

- 3) Finally, the perturbation  $\Delta\theta_k$  is chosen as follows

$$\Delta\theta_k = \arg \min_{\Delta\theta_k^{(\pm j)}, j \in \{1, \dots, n_\theta\}} \frac{1}{2} \sum_{\ell=\ell_k}^{k-1} \left( \Delta J_\ell - \vartheta_{\pm j}^\tau \phi_\ell^{(k)} \right)^2 \quad (3.9)$$

where, with some abuse of notation,

$$\Delta\theta_k^{(\pm j)} = \pm \Delta\theta_k^{(j)} \quad (3.10)$$

3) **GD-Phase:** If  $k$  is an even number, then the following are taking place:  $\Delta\theta_k$  is updated based on an *unbiased least-squares estimate*  $\widehat{\Delta J}_k$  of  $\Delta J_k$  defined as follows:

$$\widehat{\Delta J}_k(\Delta\theta_\ell, \theta_{\ell-1}, \bar{x}_\ell, \bar{x}_{\ell-1}) = \bar{\vartheta}_k^\tau \phi^{(k)}(\Delta\theta_\ell, \theta_{\ell-1}, \bar{x}_\ell, \bar{x}_{\ell-1}) \quad (3.11)$$

where

$$\bar{\vartheta}_k \mapsto \arg \min_{\vartheta} \frac{1}{2} \sum_{\ell=\ell_k}^{k-1} \left( \Delta J_\ell - \vartheta^\tau \phi_\ell^{(k)} \right)^2 \quad (3.12)$$

Then,  $\Delta\theta_k$  is calculated as follows

$$\Delta\theta_{k,i} = -\beta_k \frac{\widehat{\Delta J}_k(\alpha_k e_i, \hat{\theta}_{k-2}, \bar{x}_k, \bar{x}_{k-2})}{\alpha_k} + \hat{\theta}_{k-2,i} - \theta_{k-1,i} \quad (3.13)$$

where  $e_i$  denotes the  $n_\theta$  dimensional vector with entries  $e_{ii} = 1$ ,  $e_{ij} = 0$ ,  $i \neq j$  and  $\hat{\theta}_{k-2}$  is defined as follows:

$$\hat{\theta}_{k-2} = \begin{cases} \theta_{k-2} & \text{if } \left| \hat{\theta}_{k-2} \right| \leq c_\theta \\ \arg \min_{\theta_\ell, \ell \in \{\ell_k, \dots, k-1\}} |\theta_\ell| & \text{otherwise} \end{cases} \quad (3.14)$$

where  $c_\theta$  is a sufficiently large user-defined positive constant.

By applying the well-known Kuhn-Tucker theorem [7], [13] to the constrained optimization problem (3.7) we obtain that

$$\vartheta_{\pm j} = \begin{cases} \bar{\vartheta}_k & \text{if } \vartheta_{\pm j}^\tau \phi_{\pm j}^{(k)} \leq -\beta_k \\ \bar{\vartheta}_k - \lambda_{\pm j} \Phi_k^{-1} \phi_{\pm j}^{(k)} & \text{otherwise} \end{cases} \quad (3.15)$$

where<sup>2</sup>

$$\bar{\vartheta}_k = \Phi_k^{-1} \sum_{\ell=\ell_k}^{k-1} \Delta J_\ell \phi_\ell^{(k)} \quad (3.16)$$

(note that  $\bar{\vartheta}_k$  above coincides with the solution of the unconstrained least-squares optimization problem (3.12)),

$$\begin{aligned} \Phi_k &= \sum_{\ell=\ell_k}^{k-1} \phi_\ell^{(k)} \left( \phi_\ell^{(k)} \right)^\tau \\ \lambda_{\pm j} &= \frac{\beta_k + \sum_{\ell=\ell_k}^{k-1} \Delta J_\ell \left( \phi_\ell^{(k)} \right)^\tau \Phi_k^{-1} \phi_\ell^{(k)}}{\left( \phi_{\pm j}^{(k)} \right)^\tau \Phi_k^{-1} \phi_{\pm j}^{(k)}} \end{aligned} \quad (3.17)$$

*Remark 2:* It is worth noting that, similarly to the proposed algorithm, the algorithm proposed and analyzed in [10] consists of two phases: the GD-phase, which is exactly the same as the one of the proposed algorithm and the P-phase, which, similarly to conventional AO algorithms, introduces random perturbations to the current control parameter vector  $\theta$ . On the other hand, the algorithm proposed and analyzed in [11], involves only one phase which is similar to the CP-phase of the proposed algorithm. The replacement of the P-phase by the CP-phase is crucial for the efficiency of the proposed algorithm: contrary to the P-phase which – due to the use of random perturbations – may introduce poor performance or instability problems, the algorithm used in the CP-phase guarantees safe and efficient performance in the sense that requirement (2.7) is fulfilled. On the other hand, the alternation among the GD- and CP-phases possesses the advantage – over the algorithm of [11] which involves only a CP-phase – of efficiently dealing with unbounded exogenous inputs something that was not possible in the case of the algorithm of [11]; moreover – see the simulation section – the alternation between GD- and CP-phases seems to lead to a better performance over the algorithms that use only a CP-phase.

It is also noting that the presence of the CP-phase is more than crucial for the efficiency of the proposed algorithm; in particular, the CP-phase is responsible for providing the proposed algorithm with the so-called *Persistence of Excitation* – (PE) property, see e.g. [6], which is a sufficient and necessary condition for the neural approximator  $\widehat{\Delta J}$  to be able to efficiently learn the unknown function  $\Delta J$ .

*Remark 3 (Number of Neurons):* Contrary to other applications of neural approximators where the number of neurons  $\bar{L}_g$  should be large enough to guarantee efficient approximation over the whole input set, this is not the case here: in the case of the proposed algorithm it is sufficient that the approximator has enough regressor terms to come up with an approximation of the unknown function  $\Delta J$  over a small neighborhood around the most recent vector  $\theta_k$ . As a matter of fact, in all practical applications of algorithms using neural approximators for optimization purposes, (see [10], [11]) as well as in various applications where we tested the proposed algorithm,

<sup>2</sup>By using equality (A.3) – see Appendix A – it can be seen that  $\Phi_k^{-1}$  exists with probability 1 and, moreover, that the denominator of (3.17) is different than zero with probability 1.

a choice for  $\bar{L}_g, T_h$  according to  $\bar{L}_g \approx 2(n_\theta + n_x), T_h = 50$  was found to produce quite satisfactory results.

#### IV. MAIN RESULTS

In this section, we establish the properties of the proposed algorithm. Our basic analysis is based on two assumptions: The first assumption is described as follows:

- (A1)  $\bar{x}_k$  is bounded for all  $k$  and, moreover,  $E[x_k - \bar{x}_k | \mathcal{G}_k] = 0$  and  $E[|x_k - \bar{x}_k|^2 | \mathcal{G}_k] < \infty$ , where  $\mathcal{G}_k$  denotes the  $\sigma$ -field generated by  $\{x_0, \dots, x_{k-1}, \bar{x}_0, \dots, \bar{x}_{k-1}, \theta_0, \Delta\theta_1, \dots, \Delta\theta_{2\lfloor k/2 \rfloor - 1}\}$ .

In simple words, assumption (A1) requires that the exogenous input estimation error  $x_k - \bar{x}_k$  is zero-mean with finite variance and the estimate  $\bar{x}_k$  is bounded; note that assumption (A1) allows for  $x_k$  to be unbounded. Note also that in the case where there is no available estimate  $\bar{x}_k$ , assumption (A1) reduces to the requirement that the exogenous vector  $x_k$  is zero-mean with finite variance.

In order to describe our second assumption some preliminaries are needed: Let  $c_{\theta,x} = \sqrt{n_\theta(c_\theta + \bar{c})^2 + n_x c_x^2}$  where  $\bar{c}$  is an  $\mathcal{O}(\max\{\alpha_k\}) + \mathcal{O}(\max\{\beta_k\})$  constant such that  $|\theta_k| \leq c_\theta + \bar{c}, \forall k$  and  $c_x$  is a sufficiently large positive constant; let also (see subsection I.B)

$$\begin{aligned} \vartheta_k^* &= \mathcal{W}(L_g^k, \phi^{(k)}, \Delta J) \\ &= \arg \min_{\vartheta} \|\Delta J(\Delta\theta, \theta, x, x') - \vartheta^\tau \phi^{(k)}(\Delta\theta, \theta, x, x')\|_{c_{\theta,x}} \end{aligned}$$

and

$$\nu_k(\Delta\theta, \theta, x, x') = \Delta J(\Delta\theta, \theta, x, x') - \vartheta_k^{*\tau} \phi^{(k)}(\Delta\theta, \theta, x, x')$$

Then, the second assumption is described as follows:

- (A2)  $E[\nu_k(\Delta\theta_\ell, \theta_{\ell-1}, x_\ell, x_{\ell-1}) | \mathcal{G}_k] = \mathcal{O}\left(\eta\left(\frac{1}{L_g}, c_{\theta,x}\right)\right), \forall k \geq \bar{L}_g, \forall \ell \in \{\ell_k, \dots, k\}$ .

Typical examples that satisfy (A2) are presented below:

- 1) The case where the exogenous signal  $x_k$  is bounded with probability 1; in this case, property (P1) implies (A2).
- 2) The case where  $J(\theta, x) = J(\theta) + x$  and  $\bar{x}_k = 0$ ; then assumption (A1) and property (P1) imply (A2) with  $c_{\theta,x} = c_\theta + \bar{c}$ .
- 3) The case where the function  $S$  in (3.2) is bounded and  $E[x_{2,k} | \mathcal{G}_k] = 0$ , where  $x_{2,k}$  is defined as follows

$$x_k = \mathcal{I}_k x_{1,k} + (1 - \mathcal{I}_k) x_{2,k}$$

where  $\mathcal{I}_k = 1$  if  $|x_k| \leq c_x$  and  $\mathcal{I}_k = 0$ , otherwise.

The next theorem summarizes the properties of the proposed algorithm (3.2)-(3.9).

*Theorem 1:* Let assumptions (A1), (A2) hold. Suppose additionally that

$$\begin{aligned} \alpha_{2n+1} &> 0, \beta_{2n+1} \geq 0, \forall n \in \mathcal{Z} \\ \lim_{n \rightarrow \infty} \alpha_{2n+1} &\rightarrow \alpha > 0, \lim_{n \rightarrow \infty} \beta_{2n+1} \rightarrow \beta \geq 0 \end{aligned} \quad (4.1)$$

$$\begin{aligned} \alpha_{2n} &= \alpha_0 n^{-\eta_1}, \eta_1 \in (0, 1/2) \\ \beta_{2n} &= \beta_0 \frac{\ln n}{n} \text{ or } \beta_{2n} = \beta_0 n^{-\eta_1 - 1/2} \end{aligned} \quad (4.2)$$

where  $\alpha_0 > 0, \beta_0 > 0$ . Moreover, let  $\mathcal{C}_k = \left\{ \theta : \theta = \bar{\theta} + \Delta\theta, \Delta\theta \in \{-\alpha_k, \alpha_k\}^{n_\theta} \text{ and } \left| \Delta\theta^\tau \frac{\partial J}{\partial \theta}(\bar{\theta}, x_k) \right| < \beta_k + \gamma_1(x_{k-1})n_\theta\alpha_k^2 + \gamma_2(\theta, x_k - x_{k-1}) \right\}$ . Then,

- (a) At the **CP-phase** the following holds:

If  $\theta_{k-1} \notin \mathcal{C}_{k-1}$ ,

$$\begin{aligned} J_k &\leq J_{k-1} - \beta_k + \mathcal{O}\left(\eta\left(\frac{1}{L_g}, c_{\theta,x}\right)\right) \\ &+ \mathcal{O}\left(\sup_{\ell \in \{\ell_k, \dots, k\}} |x_\ell - \bar{x}_\ell|\right) \end{aligned} \quad (4.3)$$

while, if  $\theta_{k-1} \in \mathcal{C}_{k-1}$

$$J_k < J_{k-1} + \beta_k + \mathcal{O}(x_k) n_\theta \alpha_k^2 + \mathcal{O}(|x_k - x_{k-1}|) \quad (4.4)$$

- (b) At the **GD-phase** the following holds:

$$J_k \leq J_{k-2} - \beta_k \left| \frac{\partial J}{\partial \theta}(\hat{\theta}_{k-2}, x_k) \right|^2 + \tau_{1k} \quad (4.5)$$

where  $\tau_{1k} = \beta_k \mathcal{O}\left(\eta\left(\frac{1}{L_g}, c_{\theta,x}\right)\right) + \mathcal{O}(x_k) n_\theta \alpha_k^2 + \mathcal{O}(x_k - \bar{x}_k) + \mathcal{O}(x_k - x_{k-1})$ .

- (c) Moreover, if  $c_\theta$  is sufficiently large,

$$\lim_{k \rightarrow \infty} |\theta_k - \theta^*| = \mathcal{O}\left(\eta\left(\frac{1}{L_g}, c_{\theta,x}\right)\right), \text{ with probability 1} \quad (4.6)$$

*Proof:* See Appendix B.  $\blacksquare$

In simple words, Theorem 1 states that both of the requirements posed in section II are met by the proposed algorithm. More precisely, according to Theorem 1:

- The proposed algorithm guarantees convergence of the control parameter vector  $\theta_k$  arbitrarily close to one of the local minima  $\theta^*$ . The “distance” between  $\theta^*$  and the limit of  $\theta_k$  depends on the number of neurons used by the neural approximator (3.4); roughly speaking, the larger is the number of neurons in (3.4) the smaller is this distance.
- Moreover, the proposed algorithm guarantees that the requirement (2.7) is met. To see this, note that from (4.3), (4.4) and (4.5) we have that

$$J_k \leq J_{k-i} - B_k + N_k + E_k \quad (4.7)$$

where  $i = 1$  at the CP-phase and  $i = 2$  at the GD-phase, and

- the term<sup>3</sup>  $B_k$  is a term that is strictly positive when  $\theta_k$  is far from  $\theta^*$  and becomes negligible or equal to the design parameter  $\beta_k$  if  $\theta_k$  is close to  $\theta^*$ ;
- the term  $N_k$  is an  $\mathcal{O}\left(\eta\left(\frac{1}{L_g}, c_{\theta,x}\right)\right)$  term. This term can be made arbitrary small – for  $k \geq \bar{L}_g$  – by increasing the number of neurons  $\bar{L}_g$  used in the (3.4);
- the term  $E_k$  is a term whose magnitude depends on the magnitude of  $x_k, x_k - \bar{x}_k$  and  $x_k - \bar{x}_{k-1}$ ; note that the presence of the term  $E_k$  in (4.7) is unavoidable regardless of the particular algorithm used.

<sup>3</sup>The term  $B_k$  is defined – at the CP-phase – according to  $B_k = \beta_k$  if  $\theta_{k-1} \notin \mathcal{C}_{k-1}$  and  $B_k = -\beta_k$  if  $\theta_{k-1} \in \mathcal{C}_{k-1}$ ; and – at the GD-phase – according to  $B_k = \beta_k \left| \frac{\partial J}{\partial \theta}(\hat{\theta}_{k-2}, x_k) \right|^2$ .

*Remark 4:* Using Lemma A.1 (see Appendix A), it can be seen that the SPSSA algorithm satisfies  $J_k < J_{k-1} + \tau_k^{SPSSA}$  where

$$\tau_k^{SPSSA} = \alpha_k \left| \frac{\partial J}{\partial \theta}(\theta_{k-1}, x_k) \right| + \mathcal{O}(|x_{k-1}|) \alpha_k^2 + \mathcal{O}(|x_k - x_{k-1}|)$$

On the other hand, the standard gradient-descent algorithm  $\theta_k = \theta_{k-1} - \alpha_k \frac{\partial J}{\partial \theta}(\theta_{k-1}, \bar{x}_k)$  (which is, though, applicable only in cases where perfect knowledge of  $J$  and its gradient is available) can be seen to satisfy  $J_k < J_{k-1} + \tau_k^{GD}$  where

$$\tau_k^{GD} = \mathcal{O}(|x_k - \bar{x}_k|) \alpha_k + \mathcal{O}(|x_{k-1}|) \alpha_k^2 + \mathcal{O}(|x_k - x_{k-1}|)$$

Note that the term  $\tau_k^{SPSSA}$  may become significantly large especially when  $\theta_k$  is far from a local minimum of  $J$  wrt  $\theta$ , in which case the term  $\left| \frac{\partial J}{\partial \theta}(\theta_{k-1}, x_k) \right|$  is large. Note also that the presence of the terms  $\mathcal{O}(|x_{k-1}|)$ ,  $\mathcal{O}(|x_k - x_{k-1}|)$  is unavoidable no matter what is the particularly algorithm used.

We close this section, by presenting a direct corollary of Theorem 1 for the case  $\bar{x}_k = 0$ , i.e. the case where no estimation/prediction of the exogenous inputs is available.

*Corollary 1:* Consider the case where  $\bar{x}_k = 0$ ,  $\forall k$ . Then assumptions (A1), (A2), (4.1), (4.2) imply parts (a) and (b) of Theorem 1 and, moreover,

$$\lim_{k \rightarrow \infty} \theta_k = \theta^*, \text{ with probability 1}$$

*Proof:* Firstly notice that  $\beta_k \rightarrow 0$  implies that  $\theta_k$  converges to a constant vector  $\bar{\theta}$ . Let us re-define  $\vartheta_k^*$  as follows:

$$\vartheta_k^* = \arg \min_{\vartheta} \|\Delta J(\Delta \theta, \theta) - \vartheta^\tau \phi^{(k)}(\Delta \theta, \theta)\|_{c_k}$$

where

$$\|f(\theta, \theta')\|_{c_k} = \sqrt{\int_{\mathbb{R}^{2n_\theta}} |\bar{w}(\theta, \theta') f(\theta, \theta')|^2 d\theta d\theta'}$$

$$\bar{w}(\theta, \theta') = \begin{cases} 1 & \text{if } |\theta - \bar{\theta}| < c_k \text{ and } |\theta' - \bar{\theta}| < c_k \\ 0 & \text{otherwise} \end{cases}$$

and  $c_k = \max_{\ell \in \{\ell_k, \dots, k-1\}} |\theta_\ell - \bar{\theta}|$ . Then, we have that the terms  $\nu_k, \nu_\ell$ , defined in the proof of Theorem 1, satisfy  $\nu_k, \nu_\ell \rightarrow \eta(1/\bar{L}_g, 0) = 0$ ; the rest of the proof is the same as in Theorem 1. ■

## V. SIMULATION EXPERIMENTS

In this section, we present simulation results on the application of the proposed algorithm to the fine-tuning of an LNCS applied to a motorway ramp metering system. It has to be emphasized that the problem of computing the optimal parameters for this particular control system can be formulated as a nonlinear optimization problem whose solution depends on the traffic demand (vehicles entering the motorway network), see [18]. As a result, even in the case where the system dynamics are exactly known the computation of the optimal control parameters is a NP-hard problem, and, moreover, it requires knowledge of the traffic demand. It is also noted that – after a tedious and time-consuming fine-tuning – the implementation of the control system treated in this section in various motorway networks produced very

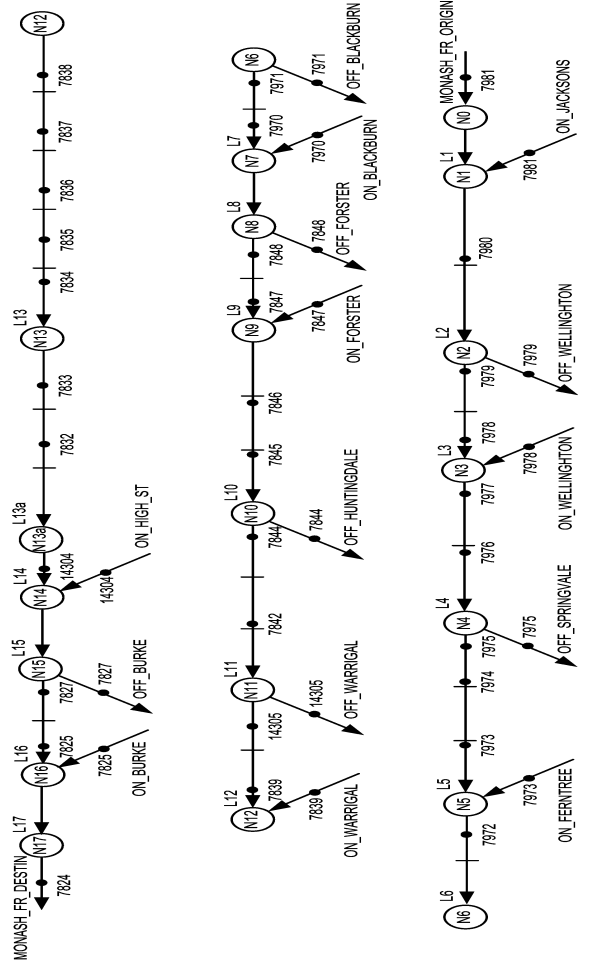


Fig. 1. The motorway stretch used in the simulations.

efficient performance [18]. Next we briefly present some details regarding the particular application.

*Traffic Network:* Figure 1 displays a schematic diagram of the 17km-long motorway stretch assumed in our experiments; this stretch is part of the Monash-CityLink-West Gate Corridor in Melbourne, Australia, operated by VicRoads. The numbered circles in figure 1 represent the network junctions and the numbered dots correspond to the detector (sensor) locations. The motorway stretch of figure 1 contains a total of 8 on-ramps and 7 off-ramps; congestion usually appears right downstream junction N3 and spills back, creating severe shock waves. In our simulations we assumed that controlled ramp metering is imposed at all ramps except ramp ON\_FERNTREE since congestion appears upstream that ramp.

*Control System:* The form of the motorway ramp metering control system assumed in our experiments can be described as follows [18]:

$$r(t) = H(L_1 z(t-1) + L_2) \quad (5.8)$$

where  $t = 0, 1, \dots$ , denotes the discrete time-index with sampling time period equal to 30 secs,  $r$  denotes the control input vector and  $z$  is a vector of traffic measurements. The control input vector  $r$  corresponds to the ramp flows allowed through the implementation of ramp metering and  $z$  corresponds



to the vector of average densities at the detector locations downstream the controlled ramps. The density measurements from detectors 7980, 7977, 7972, 7848, 7846 and 7838 were used to form the vector  $z$ . The nonlinear (saturation) operator  $H$  is used to guarantee that the control decisions satisfy minimum and maximum allowable ramp flow constraints. Finally, the constant matrix  $L_1$  and the constant vector  $L_2$  denote the tunable controller parameters. It is worth noting that the popular local ramp metering strategy ALINEA [18] has exactly the form (5.8) with the matrix  $L_1$  being a diagonal matrix and the vector  $L_2$  depending on the critical densities of the locations downstream of the on-ramps. Also, a variety of multivariable (network-wide) ramp-metering strategies, see [18], have the form (5.8) in which case  $L_1, L_2$  are calculated using nonlinear optimization or optimal control techniques.

*Traffic Network Simulation:* The macroscopic traffic simulation tool METANET [16] was used for the simulation experiments. The traffic model parameters assumed in METANET were obtained by use of nonlinear parameter estimation techniques [19], minimizing the mismatch between the METANET states and actual traffic data provided by VicRoads.

*Traffic Demand Scenarios:* Based on actual traffic data provided by VicRoads, a *basic daily traffic demand scenario*  $D_{i,t}$  (where  $D_{i,t}$  denotes the number of vehicles entering the  $i$ -th motorway origin<sup>4</sup> at the  $t$ -th interval) with duration equal to 8 hours<sup>5</sup> was designed based on actual measurements; at each fine-tuning experiment, a random perturbation of the basic scenario was used. More precisely, at each fine-tuning experiment, the traffic demand was calculated according to  $D_{i,t}^k = \max\{0, D_{i,t} + D_{i,t} w_{i,t}^k\}$ , where  $D_{i,t}^k$  denotes the traffic demand at the  $k$ -th fine-tuning experiment and  $w_{i,t}^k$  is a Gaussian zero-mean term with variance equal to 0.2. It is worth noting that the basic demand scenario corresponded to highly congested traffic conditions. Note also that due to the use of the Gaussian random term  $w_{i,t}^k$ , the exogenous signal  $x_k$  – whose entries correspond to the elements of  $D_{i,t}^k$  – is an unbounded signal in the sense that it is not possible to a priori assess an upper bound for its magnitude.

*Performance Index:* The average mean speed of the whole traffic network (in km/h) over the 8 hours was used as the performance metric to be optimized by the fine-tuning algorithm. It is worth noting that the average mean speed can be calculated based on detector measurements. Note also that since the goal of a traffic control system is to *maximize* mean speed, performance index maximization (by appropriately modifying the proposed algorithm) instead of minimization was implemented.

*Tunable Parameters:* All entries of  $L_1$  and  $L_2$  were fine-tuned, corresponding to a vector of tunable parameters with dimension equal to  $7 \times 7 + 7 = 56$ .

*Estimation of Exogenous Signals:* The exogenous vector  $x_k$  in this particular application corresponds to the traffic demand  $D_{i,t}^k$ . As it was shown in [10], a low-dimension, noisy

estimate<sup>6</sup>  $\bar{x}_k$  of the traffic demand can be constructed based on traffic measurements at the networks origins; see [10] for more details. In the particular application treated here, the methodology of [10] produced a vector  $\bar{x}_k$  with dimension  $n_x = 60$ .

*Algorithm and I-ELM-NN Design:* As explained in the paragraph “Simulations Runs” below, the proposed algorithm was applied for 149 iterations<sup>7</sup> in all simulation experiments; as a result, the maximum allowable number of I-ELM-NN nodes – denoted by  $\bar{L}_g$  in section III – is equal to  $\bar{L}_g = 75$ ; see also equation (3.3). The I-ELM-NN activation function in (1.3) was chosen according to  $S(x) = \tanh(x)$ , while the entries of  $A_i$  and  $b_i$  in (1.3) were chosen – at each algorithm iteration – to be Gaussian zero-mean random terms with variance equal to 0.1. The design constant  $T_h$  in (3.8) was chosen according to  $T_h = 50$ ; see Remark 3. The design sequences  $\alpha_k, \beta_k$  were chosen according to  $\alpha_{2n+1} = \alpha$ ,  $\beta_{2n+1} = \beta$  and  $\alpha_{2n} = \alpha_0 n^{-1/3}$ ,  $\beta_{2n} = \beta_0 \frac{\ln n}{n}$ , where  $\alpha, \beta, \alpha_0, \beta_0$  are positive constants. The choice of the constants  $\alpha, \alpha_0, \beta$  was quite straightforward. More precisely, by checking the values of  $L_1, L_2$  produced by the control strategy ALINEA in similar applications, we found that a modification of the elements of  $L_1, L_2$  according to the choice  $\alpha = \alpha_0 = 0.001$  was sufficient to produce a non-negligible change in the performance index without introducing stability problems; on the other hand, it can be seen that  $\beta$  corresponds to a “doable” increase of the performance index (mean speed) at each iteration of the algorithm; as a result a choice of  $\beta = 1$  is sufficient since it corresponds to a mean speed increase of 1 km/h. While the choice for  $\alpha, \alpha_0, \beta$  was quite straightforward, this was not the case for the constant  $\beta_0$ ; as a result, different values for  $\beta_0$  were tested throughout the simulation experiments.

*Initial Controller Parameters:* The initial matrices  $L_1$  and  $L_2$  were set equal to zero and, therefore, the starting point of the fine-tuning algorithms was a controller incorporating no knowledge about the overall system dynamics.

*Comparison with Existing AO Algorithms:* In order to evaluate the efficiency of the proposed approach, its performance was compared with the following existing AO algorithms: the SPSA algorithm [20] as well as the algorithms P-GD [10] and CP [11]. Since, the design of the three aforementioned algorithms involves quite few design parameters (similar to the design parameters  $\alpha_0, \beta_0, \bar{L}_g, T_h$ , etc of the proposed algorithm), the performance of each of these three algorithms was optimized by experimenting with different sets of their design parameters. In all cases, the proposed algorithm performance was compared to the optimized set of design parameters for the aforementioned three algorithms. Due to space limitations, more details on the choice of the design parameters of these algorithms are omitted.

*Simulation Runs:* For each of the compared algorithms, 10 different Runs using the same algorithm’s design parameters but different randomly generated traffic demand scenarios (cal-

<sup>6</sup>It is worth noting that the methodology of [10] for calculating  $\bar{x}_k$  results in an estimation error  $x_k - \bar{x}_k$  that is also an unbounded signal.

<sup>7</sup>The number of days (iterations) the fine-tuning was active was chosen to be equal to 149 in order to provide all algorithms considered in the simulations with sufficient time to converge to their “best” value for  $\theta$ .

<sup>4</sup>The motorway origins are defined as the mainstream origin N12 in Figure 1 as well as the 8 on-ramps.

<sup>5</sup>Only the “peak hours period”, i.e. the period of high traffic demand within the day, was considered in the simulations.

culated as described in paragraph ‘‘Traffic Demand Scenarios’’ of this section) were executed. In this way, a significant number of samples was created. It has to be noted that in all Runs considered, the fine-tuning process was active for 149 days (iterations); after day 149, the fine-tuning process was stopped and the best ramp metering controller (corresponding to  $\theta_k$  that produced the maximum mean speed over the 149 daily experiments) obtained throughout the fine-tuning process was tested for the next 20 daily experiments.

*Evaluation Criteria:* In order to compare the algorithms’ performance three different evaluation criteria were used:

- *TotDaysBellowMSthres*=average number of days (over all 10 different Runs) with mean speed below an appropriately defined threshold, i.e.

$$TotDaysBellowMSthres = \frac{1}{10} \sum_{R=1}^{10} \sum_{d=1}^{149} \mathcal{I}(J_d^R \leq MSthres)$$

where  $R$  denotes the Run index,  $d$  denotes the day (iteration) index,  $J_d^R$  denotes the daily mean speed at the  $d$ -th algorithm iteration for the  $R$ -th Run,  $MSthres$  denotes the aforementioned threshold and  $\mathcal{I}(J_d^R \leq MSthres) = 1$  if  $J_d^R \leq MSthres$  km/h and  $\mathcal{I}(J_d^R \leq MSthres) = 0$ , otherwise. The  $MSthres$  was chosen so that it reflects a daily mean speed bound beyond which the overall system operation is considered *unsafe*. It is worth noting that in practical ramp metering applications the traffic operators impose such thresholds; if these thresholds are repeatedly violated then the fine-tuning process, or, even the overall ramp metering system operation may be canceled. The particular value for  $MSthres$  was chosen according to  $MSthres=50$ km/h; this particular choice is 2-3 km/h below the average mean speed obtained using  $L_1 = 0, L_2 = 0$ . Apparently, the criterion *TotDaysBellowMSthres* is used for evaluating the safety attributes of the compared algorithms, i.e., their ability to keep the constant  $\epsilon_k$  in (2.7) as small as possible.

- *AverConvDay, MaxConvDay*=the average and maximum day number (over all 10 different Runs) the algorithm performance reaches a 10% distance from the best algorithm performance over the days 1-149, i.e.

$$AverConvDay = \frac{1}{10} \sum_{R=1}^{10} \arg \min_d \mathcal{J}(J_d^R \geq 0.9J_*^R)$$

$$MaxConvDay = \max_R \arg \min_d \mathcal{J}(J_d^R \geq 0.9J_*^R)$$

where  $\mathcal{J}(J_d^R \geq 0.9J_*^R) = d$  if  $J_d^R \geq 0.9J_*^R$  and  $\mathcal{J}(J_d^R \geq 0.9J_*^R) = 149$ , otherwise, with  $J_*^R$  denoting the best algorithm performance over days 1-149, i.e.  $J_*^R = \max_{d \in \{1, \dots, 149\}} J_d^R$ . This criterion was introduced in order to evaluate the *convergence rate* of the algorithms being evaluated. Note that due to the highly stochastic nature of the fine-tuning problem considered in this section, the iteration (day) number the algorithms converge to – or close to – their optimal value can vary significantly and that is the reason we incorporate the worst case performance, identified by *MaxConvDay*, in the convergence rate evaluation.

- *AverMSAAfterFT*=average daily mean speed (over all 10 different Runs and all days 150-170) after fine-tuning was stopped, i.e.

$$AverMSAAfterFT = \frac{1}{10 \times 21} \sum_{R=1}^{10} \sum_{d=150}^{170} J_d^R$$

This criterion provides with an estimate of the steady-state convergence characteristics of the algorithm being evaluated.

Figure 2 shows some instances of the application of SPSA, P-GD and CP algorithms, respectively, for different simulation Runs. As already mentioned, the performance of the above three algorithms for the best set of their design parameters is exhibited. Figure 3 shows some instances the performance of the proposed algorithm (referred as the CP-PD algorithm) for different values of the parameter  $\beta_0$ . Table 1 summarizes the performance evaluation based on the three evaluation criteria defined previously.

Close inspection of Table 1 and figures 2, 3 reveals the following:

- SPSA and P-GD algorithms practically fail to produce any improvement on the overall system performance. In almost all Runs – for both algorithms – the best performance achieved was about the same as the one achieved using the initial ramp meter controller.
- The CP algorithm guarantees a safe performance while in most cases it achieves a quickly converging performance. However, there may be cases where the CP algorithm fails to produce a significant performance improvement. Such a case is exhibited in case of Simulation Run 3 (figure 2, lower plot). In other words, while the CP algorithm guarantees safe performance, there is always the risk the CP algorithm to fail to improve the overall system performance.
- The proposed algorithm always achieves to improve considerably the overall control system’s performance. Note, however, that the improvement in the overall system performance is made possible by sacrificing safety, since the proposed algorithm’s safety attributes (identified by the criterion *TotDaysBellowMSthres*) can be slightly (case  $\beta_0 = 0.01$ ) or significantly worse (case  $\beta_0 = 0.1$ , or  $\beta_0 = 1$ ) than those of the CP algorithm. In all cases, though, the steady state improvement produced by the proposed algorithm is significantly larger than that of the rest three algorithms (see last column in Table 1). Moreover, having in mind that for all choices of  $\beta_0$  the proposed algorithm produces a significant improvement over the rest algorithms and, moreover, in the case where  $\beta_0 = 0.01$  the *TotDaysBellowMSthres* is quite small, it is expected that a real-life implementation of the proposed algorithm can be extremely successful.

## VI. CONCLUSIONS

Fine-tuning of Large-Scale Nonlinear Control Systems (LNCS) is often a tedious, complicated and risky task that is usually performed by human experts without the use of a systematic approach. In this paper, a new adaptive/neural

**Table 1:** Performance of AO algorithms.

	<i>TotDaysBellowMSthres</i>	<i>AverBestDay</i>	<i>MaxConvDay</i>	<i>AverMSAAfterFT</i>
SPSA [20]	24.7	81.3	122	54.7
P-GD [10]	15.4	50.5	113	55.1
CP [11]	2.0	26.0	67	63.2
CP-GD ( $\beta_0 = 0.01$ )	5.3	17.2	25	68.1
CP-GD ( $\beta_0 = 0.1$ )	28.2	54.3	125	72.3
CP-GD ( $\beta_0 = 1$ )	36.7	63.7	132	71.4

algorithm has been proposed that can be used towards the development of a systematic, automated procedure that will make possible the efficient and safe fine-tuning of LNCS through appropriate learning mechanisms. The proposed approach combines appropriately existing algorithms proposed by the authors in the past and the Incremental-Extreme Learning Machine Neural Network (I-ELM-NN). Among the nice properties of the proposed algorithm is its ability to deal with unbounded exogenous signals as well as its significantly improved convergence over the existing algorithms.

#### APPENDIX A TECHNICAL PROOFS

The following lemmas are needed for the establishment of the proof of Theorem 1 presented in Appendix B:

*Lemma 1:* The following holds:

$$\begin{aligned} \Delta J(\pm\Delta\theta_k, \theta_{k-1}, x_k, x_{k-1}) &= \pm\Delta\theta_k^T \frac{\partial J}{\partial \theta}(\theta_{k-1}, x_{k-1}) \\ &+ \gamma_{11}(x_{k-1}, \Delta\theta_k) + \gamma_2(\theta_{k-1} \pm \Delta\theta_k, x_k - x_{k-1}) \\ \gamma_{11}(x_{k-1}, \Delta\theta_k) &\leq \gamma_1(x_{k-1}) |\Delta\theta_k|^2 \end{aligned} \quad (\text{A.1})$$

where  $\gamma_1(x)$  is<sup>8</sup> a positive function that is bounded for bounded  $x$  and  $\gamma_2(\theta_{k-1} \pm \Delta\theta_k, x_k - x_{k-1})$  is a function satisfying  $\gamma_2(\theta, x_k - x_{k-1}) = \mathcal{O}(|x_k - x_{k-1}|)$  for any bounded  $\theta$ .

*Proof:* Following the approach adopted in [1], we fix two vectors  $\theta, \bar{\theta}$  and define  $\xi$  as the scalar parameter satisfying  $g(\xi, x) = J(\theta + \xi\bar{\theta}, x)$ . Using the chain rule we have that  $\frac{dg}{d\xi}(\xi, x) = \bar{\theta}^T \frac{\partial J}{\partial \theta}(\theta + \xi\bar{\theta}, x)$ . Therefore,

$$\begin{aligned} J(\theta + \bar{\theta}, x) - J(\theta, x) &= g(1, x) - g(0, x) \\ &= \int_0^1 \frac{dg}{d\xi}(\xi, x) d\xi = \int_0^1 \bar{\theta}^T \frac{\partial J}{\partial \theta}(\theta + \xi\bar{\theta}, x) d\xi \\ &= \int_0^1 \bar{\theta}^T \frac{\partial J}{\partial \theta}(\theta, x) d\xi + \int_0^1 (\bar{\theta}^T \frac{\partial J}{\partial \theta}(\theta + \xi\bar{\theta}, x) - \bar{\theta}^T \frac{\partial J}{\partial \theta}(\theta, x)) d\xi \\ &= \bar{\theta}^T \frac{\partial J}{\partial \theta}(\theta, x) + \int_0^1 (\bar{\theta}^T \frac{\partial J}{\partial \theta}(\theta + \xi\bar{\theta}, x) - \bar{\theta}^T \frac{\partial J}{\partial \theta}(\theta, x)) d\xi \end{aligned} \quad (\text{A.2})$$

Since  $J$  is at least  $C^2$ , we have that there exists a positive function  $L(x)$  (which is bounded for bounded  $x$ ) such that for all  $\theta, \bar{\theta}$

$$\left| \bar{\theta}^T \frac{\partial J}{\partial \theta}(\theta, x) - \bar{\theta}^T \frac{\partial J}{\partial \theta}(\bar{\theta}, x) \right| \leq L(x) |\theta - \bar{\theta}|$$

and therefore the second term in the RHS of (A.2) satisfies  $\left| \int_0^1 (\bar{\theta}^T \frac{\partial J}{\partial \theta}(\theta + \xi\bar{\theta}, x) - \bar{\theta}^T \frac{\partial J}{\partial \theta}(\theta, x)) d\xi \right| \leq \int_0^1 |\bar{\theta}| \left| \frac{\partial J}{\partial \theta}(\theta + \xi\bar{\theta}, x) - \frac{\partial J}{\partial \theta}(\theta, x) \right| d\xi \leq |\bar{\theta}| \int_0^1 L(x) \xi |\bar{\theta}| d\xi =$

<sup>8</sup>Note that  $\gamma_1(x)$  is a function that is bounded for bounded  $x$  and not a bounded function; in other words,  $\gamma_1(x)$  may be unbounded when  $x$  becomes unbounded.

$\frac{L(x)}{2} |\bar{\theta}|^2$ . By setting  $\theta = \theta_{k-1}, x = x_{k-1}, \bar{\theta} = \pm\Delta\theta_k$  in (A.2) and using the above inequality we obtain  $\Delta J(\pm\Delta\theta_k, \theta_{k-1}, x_{k-1}, x_{k-1}) = \pm\Delta\theta_k^T \frac{\partial J}{\partial \theta}(\theta_{k-1}, x_{k-1}) + \gamma_{11}(x_{k-1}, \Delta\theta_k)$  where  $\gamma_{11}(x_{k-1}, \Delta\theta_k)$  is a term satisfying  $|\gamma_{11}(x_{k-1}, \Delta\theta_k)| \leq \frac{L(x)}{2} |\Delta\theta_k|^2$ . The proof is established by defining  $\gamma_1(x) = 2L(x)$  and  $\gamma_2(\theta, x_k - x_{k-1}) = J(\theta, x_k) - J(\theta, x_{k-1})$ . ■

*Lemma 2:* For all  $k$  odd, the following holds, provided that  $\alpha_{2n+1} > 0, \forall n \in \mathcal{Z}$ :

$$\text{rank} \left[ \phi_{\ell_k}^{(k)}, \dots, \phi_{k-1}^{(k)}, \phi_{\pm j}^{(k)} \right] = L_g^k, \text{ with probability 1} \quad (\text{A.3})$$

*Proof:* We only provide with a sketch of the proof: it can be seen that – since  $S$  is invertible – if (A.3) does not hold then there exists a nonzero vector  $\chi$  such that  $\chi^T (\text{Avec}(\Delta\theta_\ell, \bar{x}_\ell - \bar{x}_{\ell-1}) + b) = 0, \ell = \ell_k, \dots, k-1$ , and  $\chi^T (\text{Avec}(\Delta\theta_k^{(\pm j)}, \bar{x}_k - \bar{x}_{k-1}) + b) = 0$  where  $A$  denotes the matrix whose rows are the vectors  $A_i^T$  and  $b = \text{vec}(b_i)$ . Since  $A_i, b_i$  and  $\Delta\theta_k^{(\pm j)}$  are randomly chosen and moreover  $\Delta\theta_\ell \neq 0$  (due to the requirement that  $\alpha_{2n+1} > 0$ ), it is quite straightforward to show that the probability a nonzero vector  $\chi$  to satisfy the above system of equations is zero. ■

*Lemma 3:* Consider the assumptions imposed in Theorem 1. Then  $\Delta J(\Delta\theta_k^{(\pm j)}, \theta_{k-1}, x_k, x_{k-1}) \leq -\beta_k$  implies (B.6).

*Proof:* Using (B.1) we directly obtain that  $\Delta J(\Delta\theta_k^{(\pm j)}, \theta_{k-1}, x_k, x_{k-1}) \leq -\beta_k \implies \vartheta_k^{*\tau} \phi_{\pm j}^{(k)} + \tilde{\nu}_{\pm j} \leq -\beta_k$  where  $\tilde{\nu}_{\pm j} = \nu_k(\Delta\theta_k^{(\pm j)}, \theta_{k-1}, x_k, x_{k-1}) + \vartheta_k^{*\tau} (\phi^{(k)}(\Delta\theta_k^{(\pm j)}, \theta_{k-1}, x_k, x_{k-1}) - \phi^{(k)}(\Delta\theta_k^{(\pm j)}, \theta_{k-1}, \bar{x}_k, \bar{x}_{k-1}))$ . Using the above relationship it is straightforward to see that  $\vartheta_k^* \in \mathcal{S}_{\pm j}$  where

$$\mathcal{S}_{\pm j} = \left\{ \vartheta \in \mathfrak{R}^{L_g^k} : \vartheta^\tau \phi_{\pm j}^{(k)} \leq -\beta_k + |\tilde{\nu}_{\pm j}| \right\}$$

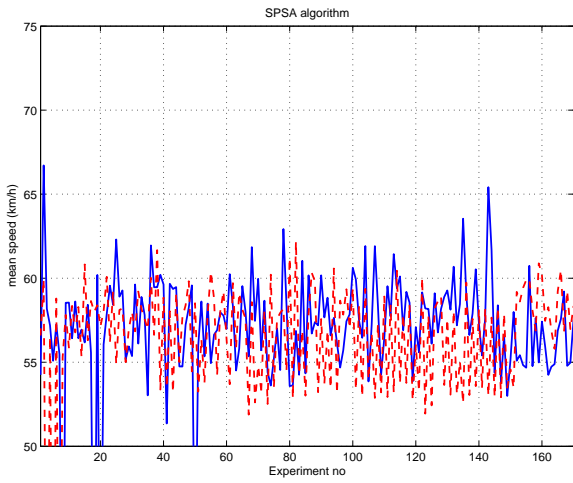
Note now that relation (B.6) is directly obtained from (B.5) and (B.4) in the case where  $\Lambda_{\pm j} = 0$ ; therefore it suffices to establish (B.6) in the case where  $\Lambda_{\pm j} \neq 0$ : since  $\Lambda_{\pm j} \neq 0$ , it is easily seen that  $\vartheta_{\pm j} \in \Sigma_{\pm j}^k$ , where

$$\Sigma_{\pm j} = \left\{ \vartheta \in \mathfrak{R}^{L_g^k} : \vartheta^\tau \phi_{\pm j}^{(k)} = -\beta_k \right\} \subseteq \mathcal{S}_{\pm j}$$

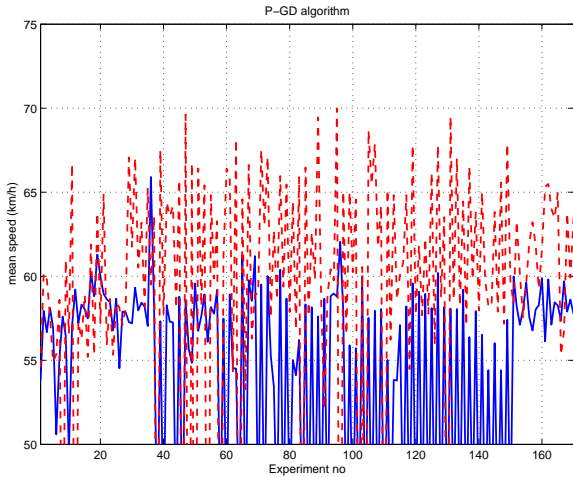
Moreover, from (B.4) and  $\Lambda_{\pm j} \neq 0$  we have that  $\bar{\vartheta}_k \in \Gamma_{\pm j}$ , where

$$\Gamma_{\pm j} = \left\{ \vartheta \in \mathfrak{R}^{L_g^k} : |\vartheta - \vartheta_k^*| \leq |\bar{\nu}_k| \text{ and } \vartheta^\tau \phi_{\pm j}^{(k)} \geq -\beta_k \right\}$$

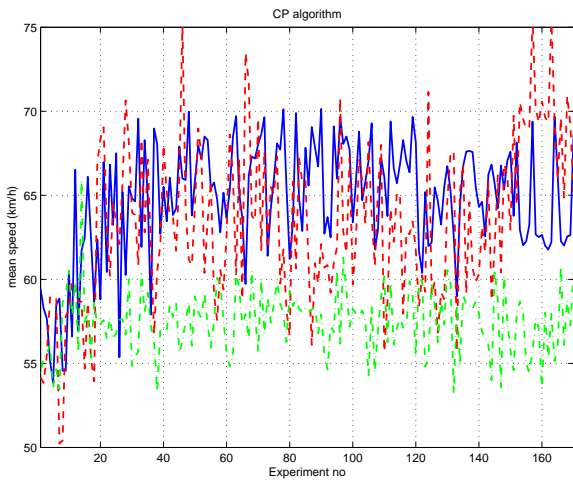
It is not difficult for someone to see that  $\vartheta_{\pm j}$  is the projection of a vector  $\bar{\vartheta}_k \in \Gamma_{\pm j}$  into  $\Sigma_{\pm j}$ . Note also that the subsets



SPSA: Simulation Run 1: Solid Curve, Simulation Run 2: Dashed Curve

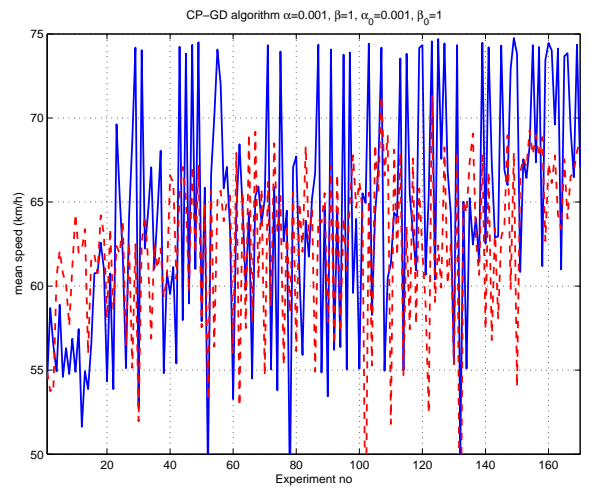
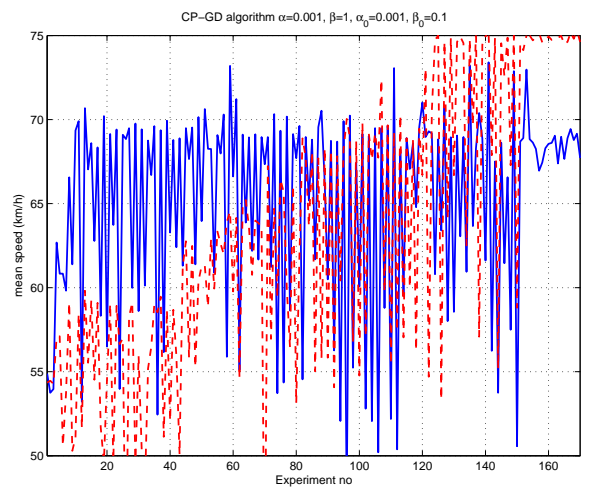
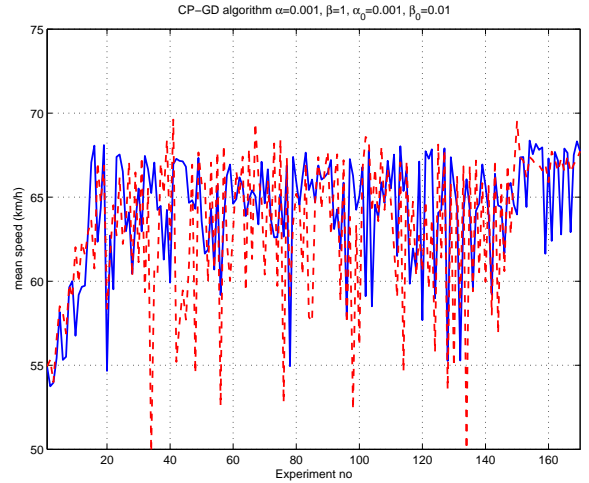


P-GD: Simulation Run 1: Solid Curve, Simulation Run 2: Dashed Curve



CP: Simulation Run 1: Solid Curve, Simulation Run 2: Dashed Curve,  
Simulation Run 3: Dash-dotted Curve

Fig. 2. Fine-tuning results using SPSA [20] (upper plot) P-GD [10] (middle plot) and CP [11] (lower plot) algorithms: in all three cases the algorithm's performance for optimized set of their design parameters is exhibited.



Simulation Run 1: Solid Curves, Simulation Run 2: Dashed Curves

Fig. 3. Fine-tuning results using the proposed algorithm for different values of the parameter  $\beta_0$ .



$\mathcal{S}_{\pm j}, \Sigma_{\pm j}, \Gamma_{\pm j}$  are convex. We have three cases: (a)  $\Sigma_{\pm j} \cap \Gamma_{\pm j} \neq \emptyset$ . In this case,

$$\vartheta_{\pm j} \in \Sigma_{\pm j} \cap \Gamma_{\pm j} \quad (\text{A.4})$$

and thus it is easily seen – since  $\Sigma_{\pm j}, \Gamma_{\pm j}$  are convex – that the distance between  $\vartheta_k^*$  and  $\vartheta_{\pm j}$  is bounded by  $|\bar{\nu}_k|$ , which establishes (B.6) for this case; (b)  $\Sigma_{\pm j} \cap \Gamma_{\pm j} = \emptyset$  and  $\vartheta^{\tau} \phi_{\pm j}^{(k)} > -\beta_k, \forall \vartheta \in \Gamma_{\pm j}$ . In this case, it can be easily seen that the distance between  $\vartheta_k^*$  and  $\vartheta_{\pm j}$  is bounded by  $|\tilde{\nu}_{\pm j}|$  (see the definition of the subset  $\mathcal{S}_{\pm j}$ ) which establishes (B.6) for this case; (c)  $\Sigma_{\pm j} \cap \Gamma_{\pm j} = \emptyset$  and  $\vartheta^{\tau} \phi_{\pm j}^{(k)} < -\beta_k, \forall \vartheta \in \Gamma_{\pm j}$ , or, equivalently,  $\Lambda_{\pm j} = 0$ , which is a contradiction. Thus, (B.6) has been established. ■

*Lemma 4:* Consider the assumptions imposed in Theorem 1. Then

$$\Delta J(\Delta\theta^{(\pm j)}, \theta_{k-1}, x_k, x_{k-1}) > -\beta_k, \forall j \in \{1, \dots, n_{\theta}\}$$

implies that  $\theta_k \in \mathcal{C}_{k-1}$  and that (B.17) holds.

*Proof:* Let  $\nabla J_{k-1} = \frac{\partial J}{\partial \theta}(\theta_{k-1}, x_{k-1})$ . From (A.1) – see Lemma 1 – we have that  $-\beta_k < \Delta J(\Delta\theta_k^{(\pm j)}, \theta_{k-1}, x_k, x_{k-1}) \leq \left(\Delta\theta_k^{(\pm j)}\right)^{\tau} \nabla J_{k-1} + \gamma_1(x_{k-1}) \left|\Delta\theta_k^{(\pm j)}\right|^2 + \gamma_2(\theta_{k-1} + \Delta\theta_k^{(\pm j)}, x_k - x_{k-1})$  which implies (since  $\Delta\theta_k^{(\pm j)} = \pm\Delta\theta_k^{(j)}$ ) that

$$\pm \bar{\Delta}^{\tau} \nabla J_{k-1} < \text{vec} \left( \beta_k + \gamma_1(x_{k-1}) \left|\Delta\theta_k^{(\pm j)}\right|^2 + \gamma_2(\theta_{k-1} + \Delta\theta_k^{(\pm j)}, x_k - x_{k-1}) \right) \quad (\text{A.5})$$

where  $\bar{\Delta} = \left[\Delta\theta_k^{(1)}, \dots, \Delta\theta_k^{(n_{\theta})}\right]$ . By using (1.1) it is straightforward to establish that  $\theta_{k-1} \in \mathcal{C}_{k-1}$ . Using the definition of the subset  $\mathcal{C}_k$ , the fact that  $\theta_{k-1} \in \mathcal{C}_{k-1}$  and Lemma 1, we have – since  $\Delta\theta_k = \Delta\theta_k^{(\pm j)} \in \{-\alpha_k, \alpha_k\}^{n_{\theta}}$  for some  $\pm j \in \{n_{\theta}, \dots, -1, 1, \dots, n_{\theta}\}$  – that  $\Delta J(\Delta\theta_k, \theta_{k-1}, x_k, x_{k-1}) \leq \left(\Delta\theta_k\right)^{\tau} \nabla J_{k-1} + \gamma_1(x_{k-1}) \left|\Delta\theta_k\right|^2 + \gamma_2(\theta_k, x_k - x_{k-1}) < \beta_k + 2\gamma_1(x_{k-1}) \left|\Delta\theta_k\right|^2 + 2\gamma_2(\theta_k, x_k - x_{k-1})$  which establishes (B.17). ■

## APPENDIX B PROOF OF THEOREM 1

Since the proposed algorithm is applied after iteration  $k = 2$ , the proof concentrates in the case where  $k \geq 2$ . Note also that from (3.14) and the fact that  $\bar{x}_k$  is bounded, we have that  $\theta_k$  is bounded for all  $k$ .

Using the definitions of  $\vartheta_k^*, \nu_k$  (see section IV), we have that

$$\Delta J = \vartheta_k^{*\tau} \phi^{(k)}(\Delta\theta, \theta, \bar{x}, \bar{x}') + \nu_k(\Delta\theta, \theta, x, x') + \vartheta_k^{*\tau} \left( \phi^{(k)}(\Delta\theta, \theta, x, x') - \phi^{(k)}(\Delta\theta, \theta, \bar{x}, \bar{x}') \right) \quad (\text{B.1})$$

where  $\bar{x}, \bar{x}'$  denote the estimates of  $x, x'$ , respectively; therefore, if  $\widehat{\Delta J}(\Delta\theta, \theta, \bar{x}, \bar{x}') = \vartheta^{\tau} \phi^{(k)}(\Delta\theta, \theta, \bar{x}, \bar{x}')$ , then

$$\Delta J(\Delta\theta, \theta, x, x') - \widehat{\Delta J}(\Delta\theta, \theta, \bar{x}, \bar{x}') = \tilde{\vartheta}_k^{\tau} \phi^{(k)}(\Delta\theta, \theta, \bar{x}, \bar{x}') + \bar{\nu}_k(\Delta\theta, \theta, x, x', \bar{x}, \bar{x}') \quad (\text{B.2})$$

where  $\tilde{\vartheta}_k = \vartheta_k^* - \vartheta$  denotes the parameter estimation error and

$$\bar{\nu}_k(\Delta\theta, \theta, x, x', \bar{x}, \bar{x}') = \nu_k(\Delta\theta, \theta, x, x') + \vartheta_k^{*\tau} \left( \phi^{(k)}(\Delta\theta, \theta, x, x') - \phi^{(k)}(\Delta\theta, \theta, \bar{x}, \bar{x}') \right) \quad (\text{B.3})$$

Using (3.16), (B.1), (B.3) we obtain

$$\begin{aligned} \bar{\vartheta}_k &= \left( \sum_{\ell=\ell_k}^{k-1} \phi_{\ell}^{(k)} \left( \phi_{\ell}^{(k)} \right)^{\tau} \right)^{-1} \sum_{\ell=\ell_k}^{k-1} \left( \vartheta_k^{*\tau} \phi_{\ell}^{(k)} + \bar{\nu}_{\ell} \right) \phi_{\ell}^{(k)} \\ &= \vartheta_k^* + \Phi_k^{-1} \sum_{\ell=\ell_k}^{k-1} \bar{\nu}_{\ell} \phi_{\ell}^{(k)} = \vartheta_k^* + \bar{\nu}_k \end{aligned} \quad (\text{B.4})$$

where – with some abuse of notation –  $\bar{\nu}_{\ell} = \bar{\nu}_k(\Delta\theta_{\ell}, \theta_{\ell-1}, x_{\ell}, x_{\ell-1}, \bar{x}_{\ell}, \bar{x}_{\ell-1})$ . Let also

$$\Lambda_{\pm j} = \begin{cases} 0 & \text{if } \bar{\vartheta}_k^{\tau} \phi_{\pm j}^{(k)} \leq -\beta_k \\ -\lambda_{\pm j} \Phi_k^{-1} \phi_{\pm j}^{(k)} & \text{otherwise} \end{cases}$$

Using (3.15) we have that

$$\vartheta_{\pm j} = \bar{\vartheta}_k + \Lambda_{\pm j} \quad (\text{B.5})$$

For each  $\pm j : j \in \{1, \dots, n_{\theta}\}$  we have the following two different cases:

- 1)  $\Delta J(\Delta\theta_k^{(\pm j)}, \theta_{k-1}, x_k, x_{k-1}) \leq -\beta_k$ . In Lemma 3 we establish that in this case the following holds:

$$|\vartheta_{\pm j} - \vartheta_k^*| \leq \max \{ |\bar{\nu}_k|, |\tilde{\nu}_{\pm j}| \} \quad (\text{B.6})$$

where  $\tilde{\nu}_{\pm j} = \bar{\nu}_k(\Delta\theta_k^{(\pm j)}, \theta_{k-1}, x_k, x_{k-1}, \bar{x}_k, \bar{x}_{k-1})$ .

- 2)  $\Delta J(\Delta\theta_k^{(\pm j)}, \theta_{k-1}, x_k, x_{k-1}) \equiv -\bar{\beta}_k > -\beta_k$ . Using similar arguments as those of the proof of Lemma 3, we can see that in this case

$$\vartheta_k^{*\tau} \phi_{\pm j}^{(k)} = -\bar{\beta}_k - \tilde{\nu}_{\pm j} \quad (\text{B.7})$$

Moreover, from (3.7) we have that

$$\vartheta_{\pm j}^{\tau} \phi_{\pm j}^{(k)} \leq -\beta_k \quad (\text{B.8})$$

Subtracting (B.7) from (B.8), we obtain that

$$(\vartheta_{\pm j} - \vartheta_k^*)^{\tau} \phi_{\pm j}^{(k)} \leq -(\beta_k - \bar{\beta}_k) + \tilde{\nu}_{\pm j}$$

which implies that

$$|\vartheta_{\pm j} - \vartheta_k^*| \geq c_3 \max \{ \beta_k - \bar{\beta}_k - \tilde{\nu}_{\pm j}, 0 \} \quad (\text{B.9})$$

for some positive constant  $c_3$ .

Consider now the process of selection of  $\Delta\theta_k = \Delta\theta_k^{(\pm j)}$  according to (3.9). We have the following two cases:

- (α)  $\exists \bar{h} \in \{-n_{\theta}, \dots, -1, 1, \dots, n_{\theta}\}$  such that  $\Delta J(\Delta\theta_k^{(\bar{h})}, \theta_{k-1}, x_k, x_{k-1}) \leq -\beta_k$ .

If  $\Delta\theta_k$  defined in (3.9) satisfies  $\Delta\theta_k = \Delta\theta_k^{(\bar{h})}$ , then we have from (B.6) that

$$\left| \vartheta_k^{(\bar{h})} - \vartheta_k^* \right| \leq \max \{ |\bar{\nu}_k|, |\tilde{\nu}_{\bar{h}}| \} \quad (\text{B.10})$$

On the other hand, if  $\Delta\theta_k = \Delta\theta_k^{(j^*)}$  for some  $j^* \in \{-n_{\theta}, \dots, -1, 1, \dots, n_{\theta}\}$  with  $j^* \neq \bar{h}$ , we have that

$\Delta J(\Delta\theta_k, \theta_{k-1}, x_k, x_{k-1}) = -\bar{\beta}_k > -\beta_k$  with  $\bar{\beta}_k < \beta_k$ . From (3.9) we have that since  $\Delta\theta_k = \Delta\theta_k^{(j^*)}$  and  $j^* \neq \bar{h}$ ,

$$\sum_{\ell=\ell_k}^{k-1} \left( \Delta J_\ell - \vartheta_{k^+}^{(\bar{h})\tau} \phi_\ell^{(k)} \right)^2 > \sum_{\ell=\ell_k}^{k-1} \left( \Delta J_\ell - \vartheta_k^{(j^*)\tau} \phi_\ell^{(k)} \right)^2 \quad (\text{B.11})$$

Using the above inequality, (B.10) and the definition of  $\vartheta_k^*$  in (B.4) it can be easily established that

$$\left| \vartheta_k^{(j^*)} - \vartheta_k^* \right| \leq \max \{ |\bar{\nu}_k|, |\tilde{\nu}_{j^*}| \} \quad (\text{B.12})$$

Moreover, we have that in this case (B.9) holds, i.e.

$$\left| \vartheta_k^{(j^*)} - \vartheta_k^* \right| \geq c_3 \max \{ \beta_k - \bar{\beta}_k - \tilde{\nu}_{j^*}, 0 \}$$

which implies – by taking into account (B.12) – that

$$|\beta_k - \bar{\beta}_k| = |\tilde{\beta}_k| \leq \max \{ |\bar{\nu}_k|, |\tilde{\nu}_{j^*}| \} \quad (\text{B.13})$$

Using the definitions of  $\bar{\nu}_k, \tilde{\nu}_k, \tilde{\nu}_{\pm j}$ , the fact that from Lemma 2  $\Phi_k^{-1}$  is bounded with probability 1 and property (P1) it can be seen, after some manipulations, that

$$\left| \tilde{\beta}_k \right| \leq \max \{ |\bar{\nu}_k|, |\tilde{\nu}_{j^*}| \} = \hat{\nu}_{\{\ell_k, \dots, k\}} + \hat{\rho}(\theta_{\{\ell_k, \dots, k\}}, x_{\{\ell_k, \dots, k\}} - \bar{x}_{\{\ell_k, \dots, k\}}) \quad (\text{B.14})$$

where,  $\hat{\rho}(\theta_{\{\ell_k, \dots, k\}}, x_{\{\ell_k, \dots, k\}} - \bar{x}_{\{\ell_k, \dots, k\}}) = \mathcal{O}(|x_{\{\ell_k, \dots, k\}} - \bar{x}_{\{\ell_k, \dots, k\}}|)$ ; (here the notation  $x_{\{\ell_k, \dots, k\}}$  is used to denote a matrix whose columns are the vectors  $x_{\ell_k}, \dots, x_k$ ; notice also that  $\hat{\rho}(\cdot)$  is a zero-mean term) and

$$\|\hat{\nu}_{\{\ell_k, \dots, k\}}\|_{c_{\theta, x}} = \mathcal{O}\left(\eta\left(\frac{1}{L_g^k}, c_{\theta, x}\right)\right) \quad (\text{B.15})$$

By combining (B.10) and (B.13), we finally conclude that

$$J_k \leq J_{k-1} - \beta_k + \tilde{\beta}_k \quad (\text{B.16})$$

where  $\tilde{\beta}_k$  satisfies (B.14).

$$(\beta) \quad \Delta J(\Delta\theta^{(\pm j)}, \theta_{k-1}, x_k, x_{k-1}) > -\beta_k \quad \forall j \in \{1, \dots, n_\theta\}.$$

In Lemma 4 we show that in this case  $\theta_{k-1} \in \mathcal{C}_{k-1}$  and, moreover, that

$$J_k < J_{k-1} + \beta_k + 2\gamma_1(x_{k-1})n_\theta\alpha_k^2 + 2\gamma_2(\theta_k, x_k - x_{k-1}) \quad (\text{B.17})$$

Therefore, we have that condition ( $\alpha$ ) holds for  $\theta_{k-1} \notin \mathcal{C}_{k-1}$ ; this together with (B.16), (B.14), (B.15) and (B.17) establish part (a).

We will now establish parts (b) and (c): in the establishment of the proof of parts (b) and (c) we consider only the gradient-descent phase; therefore from now on the subscript  $k$  corresponds to an even number.

Consider any bounded  $n_\theta$ -dimensional vector  $\bar{\Delta}\theta$ ; by using similar arguments as in (B.1), (B.2) we obtain  $\Delta J_k(\bar{\Delta}\theta, \hat{\theta}_{k-2}, \bar{x}_k, \bar{x}_{k-2}) = \bar{\nu}_k^\tau \phi^{(k)}(\bar{\Delta}\theta, \hat{\theta}_{k-2}, \bar{x}_k, \bar{x}_{k-2}) + \bar{\vartheta}_k^\tau \phi^{(k)}(\bar{\Delta}\theta, \hat{\theta}_{k-2}, \bar{x}_k, \bar{x}_{k-2}) + \bar{\nu}_k(\bar{\Delta}\theta, \hat{\theta}_{k-2}, x_k, x_{k-2}, \bar{x}_k, \bar{x}_{k-2}) = \bar{\Delta}J_k(\bar{\Delta}\theta, \hat{\theta}_{k-2}, x_k, x_{k-2}, \bar{x}_k, \bar{x}_{k-2}) + \bar{\nu}_k(\bar{\Delta}\theta, \hat{\theta}_{k-2}, x_k, x_{k-2}, \bar{x}_k, \bar{x}_{k-2}) + (\phi^{(k)}(\bar{\Delta}\theta, \hat{\theta}_{k-2}, \bar{x}_k, \bar{x}_{k-2}))^\tau \Phi_k^{-1} \sum_{\ell=\ell_k}^{k-1} \bar{\nu}_\ell \phi_\ell^{(k)}$  where  $\bar{\vartheta}_k = \vartheta_k^* - \bar{\vartheta}_k$  and the second equality was obtained by using (B.4). The terms  $\bar{\nu}_k, \bar{\nu}_\ell$  in the above relation

can be decomposed as  $\bar{\nu}_k = \nu_k(\bar{\Delta}\theta, \hat{\theta}_{k-2}, x_k, x_{k-2}) + \vartheta_k^{*\tau} \left( \phi^{(k)}(\bar{\Delta}\theta, \hat{\theta}_{k-2}, x_k, x_{k-2}) - \phi^{(k)}(\bar{\Delta}\theta, \hat{\theta}_{k-2}, \bar{x}_k, \bar{x}_{k-2}) \right)$ ,  $\bar{\nu}_\ell = \nu_k(\Delta\theta_\ell, \theta_{\ell-1}, x_\ell, x_{\ell-1}) + \vartheta_k^{*\tau} \left( \phi^{(k)}(\Delta\theta_\ell, \theta_{\ell-1}, x_\ell, x_{\ell-1}) - \phi^{(k)}(\Delta\theta_\ell, \theta_{\ell-1}, \bar{x}_\ell, \bar{x}_{\ell-1}) \right)$  and, thus we have that  $\|\Delta J_k(\bar{\Delta}\theta, \hat{\theta}_{k-2}, \bar{x}_k, \bar{x}_{k-2}) - \bar{\Delta}J_k(\bar{\Delta}\theta, \hat{\theta}_{k-2}, x_k, x_{k-2}, \bar{x}_k, \bar{x}_{k-2})\|_{c_{\theta, x}} = \mathcal{O}\left(\eta\left(\frac{1}{L_g^k}, c_{\theta, x}\right)\right) + \mathcal{O}\left(\sup_{\ell \in \{\ell_k, \dots, k\}} |x_\ell - \bar{x}_\ell|\right)$ . Using the above equality and (3.13) it is quite straightforward for someone to see that

$$\theta_k = \hat{\theta}_{k-2} - \beta_k \frac{\partial J}{\partial \theta}(\hat{\theta}_{k-2}, x_k) + \beta_k h_k$$

where

$$h_k = \mathcal{O}\left(\eta\left(\frac{1}{L_g^k}, c_{\theta, x}\right)\right) + \mathcal{O}\left(\sup_{\ell \in \{\ell_k, \dots, k\}} |x_\ell - \bar{x}_\ell|\right)$$

Combining the above two equalities with Lemma 1 we readily establish part (b).

Coming to part (c), firstly notice that from (A1) we have that the second term in the above equation is zero-mean and has finite variance; moreover, from (A1), (A2) and (3.14) we have that  $E\left[\nu_k(\bar{\Delta}\theta, \hat{\theta}_{k-2}, x_k, x_{k-2}) | \mathcal{G}_k\right] = \mathcal{O}\left(\eta\left(\frac{1}{L_g^k}, c_{\theta, x}\right)\right)$ ,  $E\left[\left|\nu_k(\bar{\Delta}\theta, \hat{\theta}_{k-2}, x_k, x_{k-2})\right|^2 | \mathcal{G}_k\right] < \infty$ ,  $E\left[\nu_k(\Delta\theta_\ell, \theta_{\ell-1}, x_\ell, x_{\ell-1}) | \mathcal{G}_k\right] = \mathcal{O}\left(\eta\left(\frac{1}{L_g^k}, c_{\theta, x}\right)\right)$ ,  $E\left[\left|\nu_k(\Delta\theta_\ell, \theta_{\ell-1}, x_\ell, x_{\ell-1})\right|^2 | \mathcal{G}_k\right] < \infty$ . Using the analysis above, we obtain that

$$\Delta\theta_{k,i} = +\hat{\theta}_{k-2,i} - \theta_{k-1,i} - \beta_k \left( \frac{\Delta J_k(\alpha_i e_i, \hat{\theta}_{k-2}, \bar{x}_k, \bar{x}_{k-2})}{\alpha_k} - \frac{\rho_{k,i}}{\alpha_k} - \frac{\varrho_{k,i}}{\alpha_k} \right) \quad (\text{B.18})$$

where  $\rho_{k,i} = \mathcal{O}\left(\eta\left(\frac{1}{L_g^k}, c_{\theta, x}\right)\right)$  and  $\varrho_{k,i}$  is a zero-mean sequence with finite variance. Moreover, by defining

$$\bar{\nabla} J_k(\theta) = E\left[\frac{\partial J}{\partial \theta}(\theta, x_k) | \mathcal{G}_k\right]$$

we can rewrite (B.18) as

$$\theta_k = \hat{\theta}_{k-2} - \beta_k \left( \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2}) - \beta_k H_{k-2} - \frac{\rho_k}{\alpha_k} - \frac{\varrho_k}{\alpha_k} \right) \quad (\text{B.19})$$

where  $H_{k-2} = \text{vec}\left(\frac{\Delta J_k(\alpha_i e_i, \hat{\theta}_{k-2}, \bar{x}_k, \bar{x}_{k-2})}{\alpha_k}\right) - \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2})$  and  $\rho_k = \text{vec}(\rho_{k,i})$ ,  $\varrho_k = \text{vec}(\varrho_{k,i})$ . If the term  $\rho_k$  was not present, then the above difference equation would be in the standard KW form, in which case convergence of  $\theta_k$  could be established by using standard arguments, see e.g. [14], [3]; in the analysis that follows, we will show that the term  $\rho_k$  cannot have a destabilizing effect. Let  $\bar{J}_k(\theta) = \int \bar{\nabla} J_k(\theta) d\theta \equiv E[J(\theta, x_k) | \mathcal{G}_k]$  and  $\bar{\varrho}_k = H_{k-2} - \frac{\rho_k}{\alpha_k} - \frac{\varrho_k}{\alpha_k}$ . Standard arguments – see e.g. the proof of Proposition 3.1 of [3] – can be used to establish that  $H_{k-2} = c_1 \alpha_k + \bar{\varrho}_k$ , where  $c_1 > 0$  and  $\bar{\varrho}_k$  is a zero-mean term with finite variance. Therefore, we have that (B.19) implies  $\bar{J}_{k-1}(\theta_k) = \bar{J}_{k-1}(\hat{\theta}_{k-2} - \beta_k \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2}) - \beta_k \bar{\varrho}_k) \leq$

$\bar{J}_{k-1}(\hat{\theta}_{k-2}) - \beta_k \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2})^\tau \left( \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2}) + \bar{\varrho}_k \right) + \beta_k^2 \bar{K} \left( \left( \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2}) + \bar{\varrho}_k \right)^2 \right) \leq \bar{J}_{k-1}(\hat{\theta}_{k-2}) - \frac{\beta_k}{2} \left| \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2}) \right|^2 - \beta_k \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2})^\tau \bar{\varrho}_k + \beta_k^2 \bar{K} (\bar{\varrho}_k)^2 + \beta_k^2 c_2$

where  $c_2$  is an  $\mathcal{O}(c_\theta)$  finite constant and  $\bar{K}$  is the Lipschitz constant. Taking conditional expectations and using the definition of  $\bar{\rho}_k$ , we obtain

$$\begin{aligned}
E[\bar{J}(\theta_k) | \mathcal{G}_k] &\leq \bar{J}_{k-1}(\hat{\theta}_{k-2}) - \frac{\beta_k}{2} \left| \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2}) \right|^2 \\
&\quad - \beta_k \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2})^\tau \left( \frac{\rho_k}{\alpha_k} + c_1 \alpha_k \right) \\
&\quad + \beta_k^2 \bar{K} E \left[ (\bar{\varrho}_k)^2 | \mathcal{G}_k \right] + \beta_k^2 c_2 \\
&\leq \bar{J}_{k-1}(\hat{\theta}_{k-2}) - \frac{\beta_k}{4} \left| \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2}) \right|^2 \\
&\quad + 2\beta_k \left( \frac{\rho_k}{\alpha_k} + c_1 \alpha_k \right)^2 + Z_k \quad (\text{B.20})
\end{aligned}$$

where  $Z_k = \beta_k^2 \bar{K} E \left[ (\bar{\varrho}_k)^2 | \mathcal{G}_k \right] + \beta_k^2 c_2$ . It is not difficult for someone to see that  $\beta_k / \alpha_k \rightarrow 0$ ,  $\sum_k \left( \frac{\beta_k}{\alpha_k} \right)^2 < \infty$ ,  $\sum_k (\beta_k)^2 < \infty$ ,  $\sum_k (\beta_k)^2 = \infty$ ,  $\sum_k \beta_k \alpha_k^2 < \infty$  which imply that  $E \left[ (\bar{\varrho}_k)^2 | \mathcal{G}_k \right] < \infty$  and  $\sum_k Z_k < \infty$ ; let us define the variable (note that  $\beta_k / \alpha_k^2 \rightarrow 0$ )

$$X_k = \begin{cases} \frac{1}{4} \beta_k \left| \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2}) \right|^2 - 2\beta_k \left( \frac{\rho_k}{\alpha_k} \right)^2 \\ \text{if } \beta_k \left| \bar{\nabla} J_{k-1}(\hat{\theta}_{k-2}) \right|^2 - 2\beta_k \left( \frac{\rho_k}{\alpha_k} \right)^2 \geq 0 \\ 0, \quad \text{otherwise} \end{cases}$$

Then, inequality (B.20) last inequality implies

$$E[\bar{J}(\theta_k) | \mathcal{G}_k] \leq \bar{J}(\hat{\theta}_{k-2}) - X_k + 2c_1 \beta_k \alpha_k^2 + Z_k$$

Using the Robbins and Siegmund theorem [22] on nonnegative almost-supermartingales, we have – since  $\sum_k \beta_k \alpha_k^2$  and  $\sum_k Z_k$  converge – that  $\sum_k X_k$  converges with probability 1; standard arguments can be now applied – see e.g. proof of part (b) of Proposition 3.1 of [3] – to show that the convergence of  $\sum_k X_k$  together with the facts that  $\sum_k \beta_k = \infty$  and  $\bar{\nabla} J$  is at least  $\mathcal{C}^1$  imply (4.6).

## REFERENCES

- [1] D. P. Bertsekas and J. N. Tsitsiklis, “Gradient convergence in gradient methods with errors,” *SIAM Journal in Optimization*, vol. 10, pp. 627-642, 2000.
- [2] J. R. Blum, “Multidimensional stochastic approximation methods,” *Ann. Math. Statist.*, vol. 25, pp. 737-744, 1954.
- [3] J. Dippon and J. Renz, “Weighted means in stochastic approximation of minima,” *SIAM Journal of Control and Optimization*, vol. 35, pp. 1811-1827, 1997.
- [4] G.-B. Huang, L. Chen and C.K. Sew, “Universal approximation using incremental constructive feedforward networks with random hidden nodes,” *IEEE Transaction on Neural Networks*, vol. 17, pp. 879-892, 2006.
- [5] G.-B. Huang and L. Chen, “Convex incremental extreme learning machine,” *Neurocomputing*, vol. 70, pp. 3056-3062, 2007.
- [6] P. A. Ioannou and J. Sun, *Stable and Robust Adaptive Control*, Englewood Cliffs, NJ: Prentice Hall, 1995.
- [7] W. Karush, “Minima of functions of several variables with inequalities as side constraints,” M.Sc. Dissertation. Dept. of Mathematics, Univ. of Chicago, Chicago, Illinois, 1939.

- [8] J. Kiefer and J. Wolfowitz, “Stochastic estimation of the maximum of a regression function,” *Ann. Math. Statist.*, pp. 462-466, 1952.
- [9] E. B. Kosmatopoulos and P.A. Ioannou, “Robust switching adaptive control of multi-input non-linear systems,” *IEEE Transactions on Automatic Control*, vol. 47, no. 4, pp. 610-624, April 2002.
- [10] E. B. Kosmatopoulos, M. Papageorgiou, A. Vakouli and A. Kouvelas, “Adaptive fine-tuning of non-linear control systems with application to the urban traffic control strategy TUC,” *IEEE Transactions on Control Systems Technology*, vol. 15, no. 6, pp. 991-1002, November 2007.
- [11] E. B. Kosmatopoulos and M. Papageorgiou, “An efficient adaptive optimization scheme,” *17th IFAC World Congress’08*, pp. 5071-5076, Seoul, Korea, July 6-11 2008.
- [12] M. Krstic, I. Kanellakopoulos and P. V. Kokotovic, *Non-linear and Adaptive Control Design*, New York: Wiley, 1995.
- [13] H.W. Kuhn and A.W. Tucker, *Nonlinear programming*, Proceedings of 2nd Berkeley Symposium, Berkeley: University of California Press, pp. 481-492, 1951.
- [14] H. K. Kushner and D. S. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, New York: Springer, 1978.
- [15] V. Maiorov and R.S. Meir, “Approximation bounds for smooth functions in  $C(\mathbb{R}^d)$  by neural networks and mixture networks,” *IEEE Transactions on Neural Networks*, vol. 9, no.5, pp. 969-978, Sept. 1998.
- [16] A. Messmer and M. Papageorgiou, “METANET: A macroscopic simulation program for motorway networks,” *Traffic Engineering & Control*, vol. 31, pp. 466-470 and 549, 1990.
- [17] F. Middelham and H. Taale, “SCOOT compared,” *3rd ITS World Congress*, Orlando, FL, U.S.A., 1996, (available only electronically); see also, F. Middelham, *et al.*, “Assessment of the SCOOT system in Nijmegen,” *IEE Conference Publication*, vol. 422, pp. 66-70, 1996.
- [18] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang, “Review of road traffic control strategies,” *Proceedings of the IEEE*, vol. 91, pp. 2043-2067, 2003.
- [19] M. Papageorgiou, I. Papamichail, M. Papageorgiou and E. Kosmatopoulos, “Modelling, configuration and simulation testing for the VicRoads ramp metering project – Final report,” DSSL, Technical University of Crete, August 2007.
- [20] J. C. Spall, “Multivariate stochastic approximation using a simultaneous perturbation gradient approximation,” *IEEE Transactions on Automatic Control*, vol. 37 pp. 332-341, 1992.
- [21] J. C. Spall, “Adaptive stochastic approximation by the simultaneous perturbation method,” *IEEE Transactions on Automatic Control*, vol. 45, pp. 1839-1853, 2000.
- [22] H. Robbins and D. Siegmund, “A convergence theorem for nonnegative almost supermartingales and some applications,” *Optimizing Methods in Statistics*, J. S. Rustari, ed., Academic Press, New York, pp. 233-257, 1971.
- [23] S. R. Weller and G. C. Goodwin, “Hysteresis switching adaptive control of linear multivariable systems,” *IEEE Transactions on Automatic Control*, vol. 39, pp. 1360-1357, 1994.

PLACE  
PHOTO  
HERE

**Elias B. Kosmatopoulos** received the Diploma, M.Sc. and Ph.D. degrees from the Technical University of Crete, Greece, in 1990, 1992, and 1995, respectively. He is currently an Assistant Professor with the Department of Production Engineering and Management, Technical University of Crete (TUC), Greece and Deputy Director of the Dynamic Systems and Simulation Laboratory at TUC. Prior to joining TUC, he was Research Assistant Professor with the Department of Electrical Engineering-Systems, University of Southern California (USC) and a Postdoctoral Fellow with the Department of Electrical & Computer Engineering, University of Victoria, B.C., Canada. Dr. Kosmatopoulos’ research interests are in the areas of neural networks, adaptive optimization and control and intelligent transportation systems. He is the author of over 35 journal papers. Currently he is being involved in various projects concerning intelligent traffic control and control of swarms of flying robots.



PLACE  
PHOTO  
HERE

**Anastasios Kouvelas** received his Diploma Degree and M.Sc. from the Department of Production & Management Engineering, Technical University of Crete, Greece. He is currently an PhD. student at the same department. His main interests are in control of urban traffic networks and intelligent transport systems.