Hindawi
Complexity
Volume 2018, Article ID 1753262, 17 pages
https://doi.org/10.1155/2018/1753262

Research Article

WH-EA: An Evolutionary Algorithm for Wiener-Hammerstein System Identification

J. Zambrano,[1] **J. Sanchis**,[2] **J. M. Herrero**,[2] **and M. Martínez**[2]

[1] *Universidad Politécnica Salesiana, Cuenca, Ecuador*
[2] *Instituto Universitario de Automática e Informática Industrial, Universitat Politecnica de Valencia, València, Spain*

Correspondence should be addressed to J. Zambrano; jzambranoa@ups.edu.ec

Received 22 November 2017; Accepted 27 December 2017; Published 20 February 2018

Academic Editor: José Manuel Andújar

Current methods to identify Wiener-Hammerstein systems using Best Linear Approximation (BLA) involve at least two steps. First, BLA is divided into obtaining front and back linear dynamics of the Wiener-Hammerstein model. Second, a refitting procedure of all parameters is carried out to reduce modelling errors. In this paper, a novel approach to identify Wiener-Hammerstein systems in a single step is proposed. This approach is based on a customized evolutionary algorithm (WH-EA) able to look for the best BLA split, capturing at the same time the process static nonlinearity with high precision. Furthermore, to correct possible errors in BLA estimation, the locations of poles and zeros are subtly modified within an adequate search space to allow a fine-tuning of the model. The performance of the proposed approach is analysed by using a demonstration example and a nonlinear system identification benchmark.

1. Introduction

Nonlinearities are present to a greater or lesser extent in all real processes. When nonlinearities are weak, linear models can be successfully used to forecast the evolution of variables or to design control schemes. Currently, a lot of methods to build linear models can be found in the literature [1–5]. However, when nonlinearities are hard, linear models just can be used in a specific operation range. If process operating range is large, cause-effect relationship should be represented by a nonlinear model. An alternative to nonlinear system modelling is to describe the process phenomena using rigorous first-principles formulation [6–8]; nevertheless, in most cases, it can be a very challenging task. Another alternative is the use of soft computing methods for process identification. In this framework, nonlinear system identification has attracted considerable interest of researchers over the past few years. Nowadays, nonlinear identification is an open research topic where some benchmark problems have been proposed [9–12] and real measurement data are available for testing and validate different nonlinear identification methods (e.g., DaISy database [13]).

One of the most challenging problems regarding nonlinear system identification is the selection of a good model structure. Currently there are several structures based on neural networks [14], block-oriented models [15, 16], Volterra series [17], NARMAX models [18], and fuzzy models [19], among others. A review of black box methods to nonlinear identification can be found in Suykens and Vandewalle [20].

In this paper, block-oriented models are considered, which are a class of nonlinear representations consisting of linear time-invariant (LTI) systems coupled with nonlinear static functions (NL) [21]. Within this class of models, the most popular ones are Wiener (LTI-NL), Hammerstein (NL-LTI), Wiener-Hammerstein (LTI-NL-LTI), and Hammerstein-Wiener (NL-LTI-NL) models [15]. Nowadays, several methods to identify these models can be found in the literature. An interesting classification of contributions that have been developed until the last decade can be found in Lopes dos Santos et al. [22].

Block-oriented models are attractive for their simplicity and great capabilities to model nonlinear dynamic systems [23–28]. Specifically, Wiener-Hammerstein models have proved to be able describe several systems like a paralyzed

skeletal muscle [29, 30], a limb reflex control system [31], a DC-DC converter [32], a heat exchanger system and a superheater-desuperheater in a boiler system [33], and a thermal process [34], among others [35]. Not only does the study of block-oriented models address parameters estimation, but also these structures are used to implement modern control strategies [36–40].

In the context of block-oriented models, knowledge of process dynamics can be a good starting point for identification [52]. In this regard, the Best Linear Approximation or BLA of a nonlinear system [52–55] can be used. For the specific case of Wiener-Hammerstein models, the BLA does not provide information about the dynamics of each LTI block. Therefore, all BLA-based Wiener-Hammerstein identification methods have concentrated their efforts on the BLA division to generate initial estimates and avoid suboptimal local minima in an optimization procedure. In Sjöberg et al. [45], both LTI subsystems are initialized with all possible BLA partitions and least squares optimization is applied to fit the nonlinearity. Although identification results are very good, the number of possible partitions (combinations) grows with the number of poles and zeros of the BLA and therefore the computational cost required for this method can be very high.

To avoid multiple BLA divisions, in Lauwers [44] and Sjöberg et al. [45] an "advanced" method is proposed where both LTI subsystems are overparameterized with all poles and zeros of the BLA. This method is formulated as a linear-in-the-parameters total-least-squares problem for which the back LTI subsystem is inverted and basis functions are used to represent both linear subsystems. To minimize the effect of overparameterization, an order reduction technique is applied. However, since the formulation is based on neglecting the effect of disturbances, the solution is in general not consistent if there is noise on the output. In addition, the BLA is required to be invertible.

Another approach to initialize Wiener-Hammerstein models is presented by Westwick and Schoukens [42], where the poles and/or zeros of the BLA are classified by using a nonlinear transformation of the input and the output residuals (quadratic/cubic BLA). On the same context of QBLA/CBLA and in line with "brute-force" method, Westwick and Schoukens [46] propose a scanning technique for a rapid evaluation of all possible BLA partitions between both LTI blocks of the Wiener-Hammerstein system. With this evaluation, the vast majority of possible partitions are discarded. Both proposals based on QBLA/CBLA show excellent results and overcome some disadvantages of "brute-force" and "advanced" methods; however, the QBLA/CBLA estimation can be difficult (high variance) since it is estimated from the BLA residuals.

In a more recent work, Schoukens et al. [41] propose a more robust method based on QBLA/CBLA. Unlike the two previous proposals, the BLA is split into a nonparametric framework. This avoids mainly the parameterization of the QBLA that can be tedious given that the number of poles and zeros tends to be high. Once the front and the back dynamics of the Wiener-Hammerstein model have been identified, a parameterization of both LTI blocks is required in an additional step. This step can be complicated because a linear phase shift can be present in the nonparametric estimate.

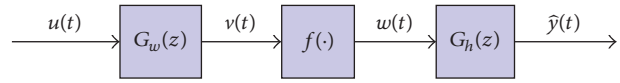

Figure 1: Wiener-Hammerstein model.

To avoid QBLA/CBLA estimation, in Vanbeylen [43] a fractional model parameterization based in multiplicities (powers) of poles and zeros is presented. The problem is formulated in the frequency domain and fractional exponents indicate which poles and zeros belong to each subsystem after an optimization problem is solved. Once the poles and zeros of the BLA have been classified, both LTI blocks must be parameterized in an additional step.

All methods mentioned here, each with its advantages and disadvantages, identify Wiener-Hammerstein models from the BLA. However, all require high user interaction to parameterize the LTI blocks and/or a final optimization to refit all parameters of the Wiener-Hammerstein model. In this work, we show that it is possible to obtain a good Wiener-Hammerstein model from the BLA by solving a single optimization problem, where user interaction is only required at the beginning, just configuring simple parameters of an evolutionary algorithm.

Hereinafter, this paper is organized as follows. In Section 2, Wiener-Hammerstein systems formulation is revisited together with some relevant information about the BLA. The proposed evolutionary algorithm, WH-EA, is presented and described in detail in Section 3, while its application and results on a numerical example and on the benchmark data SYSID'09 are presented in Section 4. Finally, in Section 5, some conclusions are reported.

2. Background

2.1. Wiener-Hammerstein Model. A Wiener-Hammerstein model consists of two LTI subsystems $G_w(z)$ and $G_h(z)$ surrounding a static nonlinear function $f(v(t))$ (Figure 1). Both LTI subsystems can be represented in the discrete-time domain as rational transfer functions in factorized form:

$$\begin{aligned} v(t) &= G_w(z)\,u(t) \\ &= K_w \frac{\prod_{i=1}^{n_b}\left(z - z_{w_i}\right)/\left(1 - z_{w_i}\right)}{\prod_{i=1}^{n_a}\left(z - p_{w_i}\right)/\left(1 - p_{w_i}\right)} u(t), \\ \hat{y}(t) &= G_h(z)\,w(t) \\ &= K_h \frac{\prod_{i=1}^{n_d}\left(z - z_{h_i}\right)/\left(1 - z_{h_i}\right)}{\prod_{i=1}^{n_c}\left(z - p_{h_i}\right)/\left(1 - p_{h_i}\right)} w(t), \end{aligned} \tag{1}$$

where z is the discrete-time operator, K_w, $p_{w_1} \cdots p_{w_{na}}$ and $z_{w_1} \cdots z_{w_{nb}}$ represent the static gain, poles, and zeros of the front LTI block, respectively, and K_h, $p_{h_1} \cdots p_{h_{nc}}$ and $z_{h_1} \cdots z_{h_{nd}}$ represent the static gain, poles, and zeros of the back LTI block, respectively.

The nonlinearity can be represented as a linear combination of a finite set (M) of basis functions:

$$w(t) = f(v(t)) = \sum_{m=1}^{M} \beta_m f_m(v(t)), \quad (2)$$

where $v(t)$ and $w(t)$ are the input and output of the static nonlinearity, β_m are weighting parameters to be estimated, and f_m are basis functions.

From (1) and (2), the output of the Wiener-Hammerstein model is analytically related to the input through the following expression:

$$\widehat{y}(t, \boldsymbol{\theta}) = G_h(z, \boldsymbol{\theta}_{\mathbf{h}}) f(\boldsymbol{\theta}_{\mathbf{NL}}, G_w(z, \boldsymbol{\theta}_{\mathbf{w}}) u(t)), \quad (3)$$

where

$$\begin{aligned} \boldsymbol{\theta}_{\mathbf{w}} &= \left[K_w, z_{w_1}, z_{w_2} \cdots z_{w_{nb}}, p_{w_1}, p_{w_2} \cdots p_{w_{na}}\right], \\ \boldsymbol{\theta}_{\mathbf{NL}} &= \left[\beta_1, \beta_2, \ldots, \beta_m\right], \\ \boldsymbol{\theta}_{\mathbf{h}} &= \left[K_h, z_{h_1}, z_{h_2} \cdots z_{h_{nd}}, p_{h_1}, p_{h_2} \cdots p_{h_{nc}}\right], \\ \boldsymbol{\theta} &= \left[\boldsymbol{\theta}_{\mathbf{w}}, \boldsymbol{\theta}_{\mathbf{NL}}, \boldsymbol{\theta}_{\mathbf{h}}\right]. \end{aligned} \quad (4)$$

The challenge is to find the best $\boldsymbol{\theta}$ so that the predicted output $\widehat{y}(t, \boldsymbol{\theta})$ is as close as possible to the measured output $y(t)$. Without prior knowledge of the system, this identification problem is not easy to solve, because there are some inconveniences that must be overcome:

(i) Parameters n_a, n_b, n_c, and n_d are not known (i.e., the structure of the LTI blocks is unknown).

(ii) The order and basis functions for nonlinearity are not known.

(iii) Without adequate initial values of $\boldsymbol{\theta}$, it is quite possible that the optimization process, trying to find the best $\boldsymbol{\theta}$, gets stuck in a local minimum.

(iv) Internal variables $v(t)$ and $w(t)$ are not measurable.

As a complement to the formulation presented, the following assumptions are made about the system.

Assumption 1. The nonlinear system to be identified can be described by (3).

Assumption 2. The Wiener-Hammerstein system will be identified from an input/output data set $\{u(t), y(t)\}_{t=1}^{N}$. The input signal $u(t)$ is Gaussian or equivalent (see Section 2.2 for more details), while the measured output $y(t)$ may be corrupted by stationary additive noise $n(t)$. It is further assumed that the noise is independent of the input excitation signal:

$$y(t) = y_0(t) + n(t). \quad (5)$$

Assumption 3. There is no cancellation of poles and zeros and all poles of both LTI subsystems must be within the unit circle.

Assumption 4. Nonlinearity is static and its current output $w(t)$ only depends on the current input $v(t)$ (i.e., the nonlinearity has no memory).

2.2. The Best Linear Approximation (BLA) of a Wiener-Hammerstein System. The BLA of a nonlinear system for a given class of excitation signals is a linear model that minimizes the expected mean square error between the true output of the nonlinear system and the output of the linear model [56]:

$$G_{\mathrm{BLA}}(z) = \arg\min_{G(z)} E\left[\left|y(t) - G(z) u(t)\right|^2\right], \quad (6)$$

where $u(t)$ is the input that excites the nonlinear system, $y(t)$ is the measured output, and E is the expectation operator. An alternative way to obtain the BLA of a nonlinear system is in a nonparametric framework:

$$G_{\mathrm{BLA}}(j\omega_k) = \frac{S_{yu}(j\omega_k)}{S_{uu}(j\omega_k)}, \quad (7)$$

where $S_{yu}(j\omega_k)$ is the cross-power spectrum between the output $y(t)$ and the input $u(t)$ and $S_{uu}(j\omega_k)$ is the auto power spectral density of $u(t)$ [54, 57].

The BLA depends on the excitation power spectrum (bandwidth and amplitude level) and excitation probability density function. Therefore, obtaining the BLA is restricted to the type of input signal that excites the process. Most estimation methods to obtaining the BLA use Gaussian noise signals or equivalent [58].

When a nonlinear system is excited with a Gaussian signal or equivalent, according to Bussgang's theorem [59], the nonlinearity can be replaced by a constant (K_{NL}). Therefore, in the specific case of a Wiener-Hammerstein system, the BLA can be defined by the following expression:

$$G_{\mathrm{BLA}}(z) = K_{\mathrm{NL}} G_{wh}(z), \quad (8)$$

where $G_{wh}(z)$ represents the dynamics of the nonlinear system:

$$G_{wh}(z) = \frac{\prod_{i=1}^{n_b+n_d} (z - z_i)/(1 - z_i)}{\prod_{i=1}^{n_a+n_c} (z - p_i)/(1 - p_i)} u(t). \quad (9)$$

It is evident that $p_1 \cdots p_{n_a+n_c}$ and $z_1 \cdots z_{n_b+n_d}$ are the poles and zeros that must be assigned to $G_w(z)$ and $G_h(z)$. Although the BLA does not provide information to distinguish the dynamics between both LTI subsystems, knowledge of the overall dynamics of a Wiener-Hammerstein system is a good starting point to identify such systems.

3. The Evolutionary Algorithm (WH-EA)

In this paper, the identification of a Wiener-Hammerstein system is addressed as an optimization problem, which is formulated considering the following issues:

(i) The BLA is estimated in the first instance.

(ii) The poles and zeros of the BLA must be classified to find the dynamics of the front and back of the Wiener-Hammerstein model.

(iii) The pole-zero locations of the BLA can change moderately to improve modelling errors.

(iv) Without loss of generality, it is possible to model a Wiener-Hammerstein system considering that both linear blocks have unit gain (gains from $G_w(z)$ and $G_h(z)$ are not part of the optimization problem).

(v) The nonlinear static function is modelled as a piecewise function represented by a set of points in the v, w plane.

To explain in detail how WH-EA works, this section has been divided into three parts. First part explains how Wiener-Hammerstein model is coded for individuals in the population; in addition, the optimization problem statement is presented. Second part explains in detail the customized genetic operators developed. Finally, the third part explains the general procedure of WH-EA.

3.1. Optimization Problem Statement. Changing pole/zero locations of the BLA to improve modelling error implies new estimates around the known values. These locations for both linear subsystems are coded in a single vector as follows:

$$\mathbf{P} = [zc_1, \ldots, zc_{nc}, zr_1, \ldots, zr_{nr}, zi_1, \ldots, zi_{nc}, pc_1, \ldots, pc_{mc}, pr_1, \ldots, pr_{mr}, pi_1, \ldots, pi_{mc}], \quad (10)$$

where $zr_1, \ldots, zr_{nr}$ and $pr_1, \ldots, pr_{mr}$ contain the locations of the real zeros and poles, respectively, and $zc_1, \ldots, zc_{nc}$ and $zi_1, \ldots, zi_{nc}$ contain the real and imaginary parts of complex conjugate zeros, respectively, while $pc_1, \ldots, pc_{mc}$ and $pi_1, \ldots, pi_{mc}$ contain the real and imaginary parts of complex conjugate poles, respectively. The values of nc, nr, mc, and mr depend on the number of zeros and poles (real and/or complex conjugates) of the BLA.

Poles and zeros contained in (10) must be classified to obtain the dynamics of the front and back blocks of a Wiener-Hammerstein model. This classification is performed using a binary vector:

$$\mathbf{C} = [xz_1, \ldots, xz_{nc+nr}, xp_1, \ldots, xp_{mc+mr}]. \quad (11)$$

The first part of the vector, $\mathbf{C}$ $(xz_1, \ldots, xz_{nc+nr})$, is associated with $zc_1, \ldots, zc_{nc}$, $zr_1, \ldots, zr_{nr}$ and indicates the zeros classification, while its second part, $\mathbf{C}$ $(xp_1, \ldots, xp_{mc+mr})$, is associated with $pc_1, \ldots, pc_{mc}$, $pr_1, \ldots, pr_{mr}$ indicating the poles classification. Note that imaginary parts are not considered for classification since they are already associated with their corresponding real parts. It is assumed that if $xz_{i\text{th}} = 1$, the corresponding ith element of $\mathbf{P}$ with $i = 1, \ldots, nc + nr$ (i.e., a real zero or a pair of complex conjugated zeros) will belong to the subsystem $G_w(z)$; otherwise, it will belong to the subsystem $G_h(z)$. In the same way, this correspondence can be applied to classify the poles using $xp_1, \ldots, xp_{mc+mr}$.

For example, if a nonlinear system is approximated by a BLA with four poles, $p_{1,2} = -0.32 \pm 0.77j$, $p_3 = -0.11$, and $p_4 = 0.17$, and three zeros, $z_{1,2} = 1.41 \pm 0.56j$, $z_3 = 1.1$, then $nc = 1$, $nr = 1$, $mc = 1$, and $mr = 2$, $\mathbf{P}$ would be structured as $[1.41, 1.1, 0.56, -0.32, -0.11, 0.17, 0.77]$, and vector $\mathbf{C}$ should contain five elements whose values switch between zero and one as the algorithm evolves. By way of illustration if $\mathbf{C} = [1, 0, 0, 1, 1]$, then $G_w(z)$ would have two zeros and two poles: $z_{1,2} = 1.41 \pm 0.56j$, $p_3 = -0.11$, $p_4 = 0.17$, while $G_h(z)$ would have a zero and two poles: $z_3 = 1.1$, $p_{1,2} = -0.32 \pm 0.77j$.

With respect to nonlinear static function, let us consider that it is represented by a set of n points:

$$\mathbf{B} = [v_1, \ldots, v_n, w_1, \ldots, w_n], \quad (12)$$

where the pairs $(v_1, w_1), \ldots, (v_n, w_n)$ correspond to their coordinates in a two-dimensional v-w plane. The location of these points and the interpolation method used will determine the quality of the captured static nonlinearity.

The proposed evolutionary algorithm is based on stochastic population of candidate solutions (individuals). Each individual contains genetic information related to

(i) the pole/zero locations in the Z-plane of the linear subsystems ($\mathbf{P}$),

(ii) the point coordinates representing the nonlinear static function ($\mathbf{B}$),

(iii) and the pole/zero classification for blocks $G_w(z)$ and $G_h(z)$ ($\mathbf{C}$),

such that any Wiener-Hammerstein model ((1) and (2)) can be easily described from this coded information. Recall that gains from linear blocks are assumed to be 1 and that parameters n_a, n_b, n_c, and n_d will be implicitly optimized and they will depend on the structure of vector $\mathbf{C}$.

To find the best set of parameters, an optimization problem is stated based on a prediction-error method and the typical mean-squared error criterion (although any other criteria can be used in the proposed method, such as the mean absolute or maximum error criteria):

$$\varepsilon(t, \boldsymbol{\theta}) = y(t) - \widehat{y}(t, \boldsymbol{\theta}), \quad (13)$$

$$J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{t=1}^{N} \varepsilon^2(t, \boldsymbol{\theta}), \quad (14)$$

where $\boldsymbol{\theta} = [\mathbf{P}, \mathbf{B}, \mathbf{C}]$ and the solution of the optimization problem is stated as

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \quad J(\boldsymbol{\theta}), \quad (15)$$

where $\widehat{\boldsymbol{\theta}}$ contains the genetic information from the best individual at the end of generations.

3.2. Genetic Operators. Customized mutation and crossover operators will be developed taking in mind the problem at hand: to identify all parameters of the Wiener-Hammerstein model in a single optimization trial. Figure 2 shows the structure of an individual as well as the genetic operators developed on each piece of genetic information. Note that i and g have been introduced into the formulation. Subscript i represents an individual in the population, while the superscript g indicates the current population.

The specific mutation and crossover operators designed are randomly selected to maintain a balance between exploration and exploitation of the search space. Mutation operations are used to maintain genetic diversity, while crossover

Pole-zero locations *Zeros* $P_i^g = \{ \boxed{zc_1, \ldots, zc_{nc}} \boxed{zr_1, \ldots, zr_{nr}} \boxed{zi_1, \ldots, zi_{nc}} \cdots$ *Real values* *Imag. values* *Poles* $\cdots \boxed{pc_1, \ldots, pc_{mc}} \boxed{pr_1, \ldots, pr_{mr}} \boxed{pi_1, \ldots, pi_{mc}} \}$ *Real values* *Imag. values*	Mutation M.1 Crossover C.1
Static nonlinearity $B_i^g = \{ \boxed{v_1, \ldots, v_n,} \boxed{w_1, \ldots, w_n} \}$ *Abscissa* *Ordinate*	Mutation M.2 Mutation M.3 Crossover C.2
Pole-zero classification $C_i^g = \{ \boxed{xz_1, \ldots, xz_{nc+nr},} \boxed{xp_1, \ldots, xp_{mc+mr}} \}$ *Zeros* G_w *or* G_h *Poles* G_w *or* G_h	Mutation M.4

Figure 2: Structure of individual and genetic operations performed on each piece of genetic information.

operations allow genetic information from the best individuals to be combined and disseminated throughout the generations. Further details on how the algorithm works will be given in Section 3.3.

3.2.1. Location in the Z-Plane of Poles and Zeros. Theoretically in a Wiener-Hammerstein model, the pole-zero locations of $G_w(z)$ and $G_h(z)$ subsystems correspond to the pole-zero locations of the BLA; however, it is well known that once the BLA has been divided, a refit can be used to improve the modelling error. In this regard, the proposed algorithm considers that while the BLA is divided and nonlinearity is captured, the pole-zero locations can change subtly.

Both operations used on this portion of genetic information produce offspring vector $\widetilde{\mathbf{P}}^{\mathbf{g}}$, which directly inherits from its parent $\mathbf{P}_{\mathbf{i}}^{\mathbf{g}}$ all the genetic information except in a gene. This gene will be selected using a random integer number $r_{zp} \in [1, \ldots, nr + 2nc + mr + 2mc]$ and modified according to the corresponding genetic operator *mutation M.1* or *crossover C.1*.

Mutation M.1. The selected gene is mutated to explore in an individualized way new pole-zero locations of the BLA. A new location $\widetilde{P}_j^g$ is determined by a random number N_{zp} with Gaussian distribution:

$$\widetilde{P}_j^g = \begin{cases} P_{i,j}^g + N_{zp}\left(0, \sigma^2(g)\right) & \text{if } \left(j = r_{zp}\right) \\ P_{i,j}^g & \text{otherwise,} \end{cases} \tag{16}$$

where $j = 1 \cdots nr + 2nc + mr + 2mc$. $P_{i,j}^g$ and $\widetilde{P}_j^g$ represent the jth elements of vectors $\mathbf{P}_{\mathbf{i}}^{\mathbf{g}}$ and $\widetilde{\mathbf{P}}^{\mathbf{g}}$, respectively.

The new locations for poles and zeros are explored within a search space defined by $\mathbf{P}^{\mathbf{min}}$ and $\mathbf{P}^{\mathbf{max}}$; therefore, $P_j^{\min} \leq \widetilde{P}_j^g \leq P_j^{\max}$, where $P_j^{\min}$ and $P_j^{\max}$ are the jth elements of vectors $\mathbf{P}^{\mathbf{min}}$ and $\mathbf{P}^{\mathbf{max}}$, respectively (see Section 3.3 for more details on search space for poles and zeros).

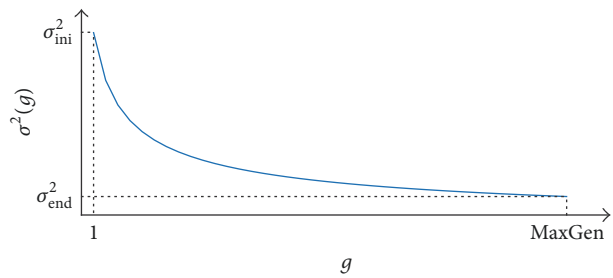

Figure 3: Variation of standard deviation over generations to control the aggressiveness of mutations.

Aggressiveness of mutations can be controlled through the standard deviation:

$$\sigma^2(g) = \frac{\Delta_s}{100}\left(\frac{\sigma^2_{\text{ini}}}{\sqrt{1 + g * \sigma^2_{\text{ratio}}}}\right), \qquad \sigma^2_{\text{ratio}} = \frac{\left(\sigma^2_{\text{ini}}/\sigma^2_{\text{end}}\right)^2 - 1}{\text{MaxGen} - 1}, \tag{17}$$

where MaxGen is the predefined number of algorithm generations; σ^2_{ratio} is the rate at which the standard deviation will decrease from σ^2_{ini} to σ^2_{end} as the generations pass (see Figure 3); the parameter Δ_s bounds the limits of the interval in which the selected gene can be moved. For this mutation, $\Delta_s = P_{r_{zp}}^{\max} - P_{r_{zp}}^{\min}$. Variation of $\sigma^2(g)$ will allow mutations to be more subtle in the last generations to achieve a fine-tuning of the corresponding parameters.

Crossover C.1. The selected gene is formed using genetic information from the parent, $P_{i,j}^g$, combined with the corresponding genetic information from the best individual, $P_{\text{best},j}^g$, in the current population:

$$\widetilde{P}_j^g = \begin{cases} \dfrac{P_{i,j}^g + P_{\text{best},j}^g}{2} & \text{if } \left(j = r_{zp}\right) \\ P_{i,j}^g & \text{otherwise,} \end{cases} \tag{18}$$

where $j = 1 \cdots nr + 2nc + mr + 2mc$.

3.2.2. Nonlinear Static Function. As the algorithm evolves, points for nonlinear static function must be located adequately in the v-w plane. Here any type of interpolation can be used to capture the static nonlinearity. To achieve a good fit, mutations M.2 and M.3 plus a crossover operation are used. Both mutations used on this portion of genetic information produce offspring vector $\widetilde{\mathbf{B}}^{\mathbf{g}}$, which directly inherits from its parent $\mathbf{B}_{\mathbf{i}}^{\mathbf{g}}$ all the genetic information except in two genes. This pair of genes represents the coordinates of the point that will be modified. Unlike mutation operations, the crossover operation generates offspring with a single modified gene which corresponds to the ordinate of a point. Given the correspondence between the abscissa and the ordinate of a point, for the three operations a single integer random

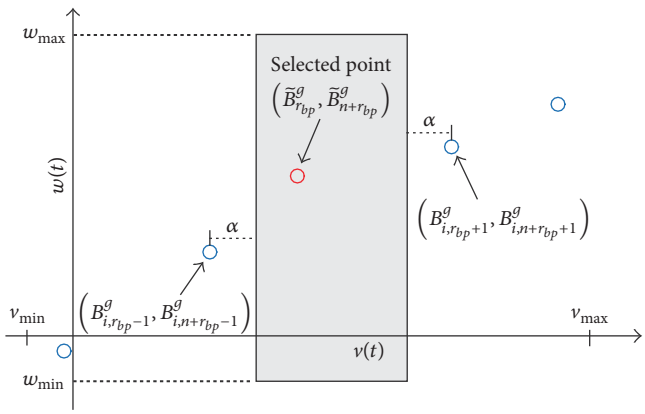

FIGURE 4: Bounds for mutation M.2. Grey area indicates $\widetilde{B}^g_{r_{bp}}$ and $\widetilde{B}^g_{n+r_{bp}}$ feasible space.

number $r_{bp} \in [1, n]$ will allow us to select the gene(s) to be modified.

Mutation M.2. This genetic operation allows us to explore in the v-w plane new positions for the points. The mutation in both genes is handled by random numbers (N_v, N_w) with Gaussian distribution:

$$\widetilde{B}^g_j = \begin{cases} B^g_{i,j} + N_v\left(0, \sigma^2(g)\right) & \text{if } \left(j = r_{bp}\right) \\ B^g_{i,j} + N_w\left(0, \sigma^2(g)\right) & \text{if } \left(j = n + r_{bp}\right) \\ B^g_{i,j} & \text{otherwise} \end{cases} \quad (19)$$

with $j = 1 \cdots 2n$. $B^g_{i,j}$ and $\widetilde{B}^g_j$ represent the jth elements of vectors $\mathbf{B^g_i}$ and $\widetilde{\mathbf{B}}^{\mathbf{g}}$, respectively. To avoid overlapping points, bounds for mutations on the abscissa axis are set depending on the selected point to mutate $(\widetilde{B}^g_{r_{bp}})$ and the location of its neighbors according to

(i) $B^g_{i,r_{bp}} + \alpha < \widetilde{B}^g_{r_{bp}} < B^g_{i,r_{bp}+1} - \alpha$; if $r_{bp} = 1$,

(ii) $B^g_{i,r_{bp}-1} + \alpha < \widetilde{B}^g_{r_{bp}} < B^g_{i,r_{bp}+1} - \alpha$; if $r_{bp} = 2 \cdots n-1$,

(iii) $B^g_{i,r_{bp}-1} + \alpha < \widetilde{B}^g_{r_{bp}} < B^g_{i,r_{bp}} - \alpha$; if $r_{bp} = n$,

where α is a user-defined parameter that indicates how close the points can be located. To achieve a good fit of the nonlinearity α must be small, relative to the search space on the abscissa axis defined by v_{min} and v_{max}. Note that the horizontal boundaries for the endpoints are delimited by their position and their left or right neighbor, respectively. Bounds for mutations on ordinate axis are fixed and equal for all points. This allows each point to move freely throughout the search space on the ordinate axis defined by w_{min} and w_{max}. The vertical and horizontal bounds for a selected point are illustrated in Figure 4. When mutation M.2 is required, the selected point can be changed to a new position within the grey rectangle. Details on how to determine v_{min}, v_{max} w_{min}, and w_{max} will be given in Section 3.3.

To achieve a fine-tuning of all nonlinearity points, mutations' aggressiveness can be controlled through standard deviation (17) as in mutation M.1. Note that $\Delta_s = w_{max} - w_{min}$ is constant for all mutations over ordinate axis, while for abscissa axis mutations, Δ_s can be calculated as

$$\Delta_s = \begin{cases} B^g_{i,r_{bp}+1} - \alpha - v_{min}; & \text{if } \left(r_{bp} = 1\right) \\ B^g_{i,r_{bp}+1} - B^g_{i,r_{bp}-1} - 2\alpha & \text{if } \left(r_{bp} = 2 \cdots n-1\right) \\ v_{max} - B^g_{i,r_{bp}-1} - \alpha & \text{if } \left(r_{bp} = n\right). \end{cases} \quad (20)$$

Mutation M.2 has great potential to explore the searching space. This genetic operation will locate points where there are slope changes. During first generations, it is useful to shape nonlinearity, while in last ones, it allows a refinement. However, when a point is located where there is a slope change, it could be kept in this location until the end of generations, especially when there are abrupt changes in the slope. Because jumps between points are not allowed with mutation M.2, when a point is kept in a place where there is a significant change of slope, one or more points would remain trapped to the left or right of it. This would lead to having redundant points in a segment that would not require so many or worse, to having a segment (curvature) that would not contain enough points. To avoid this drawback, the exploration in the search space is complemented with mutation M.3.

Mutation M.3. This genetic operation is designed to concentrate as many points as possible on the curvatures that nonlinearity can have. Therefore, it will be required that each point can be displaced on the abscissa axis by jumping one or more positions of the other points. Let us define a segment as the horizontal space between two consecutive points (so for n points there will be $n-1$ segments); then a random integer number $r_s \in [1, n-1]$ will indicate to which segment the selected point will move. The first half of the offspring vector is found using the following expression:

$$\widetilde{B}^g_j = \begin{cases} \dfrac{B^g_{i,r_s} + B^g_{i,r_s+1}}{2} & \text{if } \left(j = r_{bp}\right) \\ B^g_{i,j} & \text{otherwise} \end{cases} \quad (21)$$

with $j = 1 \cdots n$. Note that if $r_{bp} = r_s$ or $r_{bp} = r_s - 1$, the corresponding point will not make a jump but it will be located at the midpoint between its current position and the position of the point on the right or left, respectively. As in the mutation M.2 to prevent points from getting too close together, the α parameter is also used in this mutation; therefore, a jump is conditioned to the space available in the selected segment to accommodate a new point. Minimum space should be 2α. If this condition is not met, r_s must be regenerated to randomly search for another segment.

To provide a smooth transition between adjacent segments, gene mutation corresponding to the position on the ordinates axis is performed using a quadratic interpolation. To do that, three neighboring points are required. The second half of the offspring vector is found using the following expression:

$$\widetilde{B}_j^g = \begin{cases} \left[\left(\widetilde{B}_{j-n}^g\right)^2, \widetilde{B}_{j-n}^g, 1\right] * \begin{bmatrix} k_2 \\ k_1 \\ k_0 \end{bmatrix} & \text{if } \left(j = n + r_{bp}\right) \\ B_{i,j}^g & \text{otherwise} \end{cases} \quad (22)$$

with $j = n + 1 \cdots 2n$. $\widetilde{B}_{j-n}^g$ is the v-coordinate of the selected point to mute which can be found with (21); k_0, k_1, and k_2 are the coefficients of the quadratic polynomial Ψ defined by three adjacent points selected once the new v-coordinate of the point that is mutating is known. The three adjacent points can be selected directly when a point has mutated to the first or last segment,

$$\Psi = \begin{cases} f\left(\left(B_{i,1}^g, B_{i,n+1}^g\right); \left(B_{i,2}^g, B_{i,n+2}^g\right); \left(B_{i,3}^g, B_{i,n+3}^g\right)\right): & \text{if } \left(r_s = 1\right) \\ f\left(\left(B_{i,n}^g, B_{i,2n}^g\right); \left(B_{i,n-1}^g, B_{i,2n-1}^g\right); \left(B_{i,n-2}^g, B_{i,2n-2}^g\right)\right): & \text{if } \left(r_s = n - 1\right) \end{cases}, \quad (23)$$

while if the point has mutated to a nonextreme segment, the three adjacent points can be selected using the two points that define that segment plus one on its right or left. For more effective exploration, a random number $r_{3p} \in (0, 1]$ is used for selection:

$$\Psi = \begin{cases} f\left(\left(B_{i,r_s-1}^g, B_{i,n+r_s-1}^g\right); \left(B_{i,r_s}^g, B_{i,n+r_s}^g\right); \left(B_{i,r_s+1}^g, B_{i,n+r_s+1}^g\right)\right): & \text{if } \left(r_{3p} \leq 0.5\right) \\ f\left(\left(B_{i,r_s}^g, B_{i,n+r_s}^g\right); \left(B_{i,r_s+1}^g, B_{i,n+r_s+1}^g\right); \left(B_{i,r_s+2}^g, B_{i,n+r_s+2}^g\right)\right): & \text{otherwise} \end{cases}. \quad (24)$$

Figure 5 illustrates how a jump occurs with mutation M.3. Notice that the new ordinate is calculated according to the polynomial formed by the two points of the segment plus a point to the right: that is, $r_{3p} > 0.5$. After a jump has occurred, an ascending reordering of the points with respect to the abscissa values is necessary.

Crossover C.2. This genetic operation works just like crossover C.1 and is applied only to vary the position of a point on the ordinate axis:

$$\widetilde{B}_j^g = \begin{cases} \dfrac{B_{i,j}^g + B_{\text{best},j}^g}{2} & \text{if } \left(j = n + r_{bp}\right) \\ B_{i,j}^g & \text{otherwise} \end{cases} \quad (25)$$

with $j = n + 1 \cdots 2n$. $B_{\text{best},j}^g$ is the jth element of vector $\mathbf{B}_{\mathbf{best}}^{\mathbf{g}}$, which corresponds to the individual in the current population g with the best fitness value.

3.2.3. Pole-Zero Classification. Due to the stochastic nature of evolutionary algorithms, the binary values of (11) will change as the algorithm evolves, generating different structures of $G_w(z)$ and $G_h(z)$. The evolution of this piece of genetic information is handled by a simple mutation operator.

Mutation M.4. Unlike the previous ones, this operator generates a new vector $\widetilde{\mathbf{C}}^{\mathbf{g}}$ that depends entirely on the effects of mutation, meaning that for this piece of genetic information there is no information exchange between generations. This allows free testing of different structures for $G_w(z)$ and $G_h(z)$ to avoid premature convergence. When this operation is required, a random process will generate the mutation vector:

$$\widetilde{C}_j^g = \begin{cases} 1 & \text{if } N_c \leq 0.5 \\ 0 & \text{otherwise} \end{cases} \quad (26)$$

with $j = 1 \cdots nc + nr + mc + mr$. $\widetilde{C}_j^g$ is the jth element of vector $\widetilde{\mathbf{C}}^{\mathbf{g}}$. N_c is a random number with standard uniform distribution on the open interval $(0, 1)$. Note that the structure of $\widetilde{\mathbf{C}}^{\mathbf{g}}$ is built under two considerations: the LTI subsystems cannot be improper and the sum of zeros and the sum of the poles between both subsystems must be equal to the number of zeros and poles of the BLA, respectively.

3.3. WH-EA Description. WH-EA is an elitist evolutionary algorithm that evolves a population of NP individuals. Each individual contains three portions of genetic information related to the parameters of a Wiener-Hammerstein model ($\boldsymbol{\theta}_{\mathbf{i}}^{\mathbf{g}} = [\mathbf{P}_{\mathbf{i}}^{\mathbf{g}}, \mathbf{B}_{\mathbf{i}}^{\mathbf{g}}, \mathbf{C}_{\mathbf{i}}^{\mathbf{g}}]$). Like any other evolutionary algorithm, WH-EA is inspired by biological evolution over generations. Starting from an initial population, new generations are created using information of the current generation g and performing crossover and/or mutation operations and selection based on the fitness of the new individuals. Algorithm 1 shows a pseudocode of main steps performed in WH-EA, whereas details of all the parameters that the algorithm uses are shown in WH-EA Parameters.

Initialize the Population. The initial population $[\mathbf{P}^{\mathbf{0}} \ \mathbf{B}^{\mathbf{0}} \ \mathbf{C}^{\mathbf{0}}]$ contains NP individuals generated within a search space delimited by lower and upper bounds. For each piece of genetic information, the lower and upper bounds can be determined from the information provided by the BLA (location of poles and zeros and static gain).

Real and imaginary values of poles and zeros from the BLA without modifications are introduced as part of the first

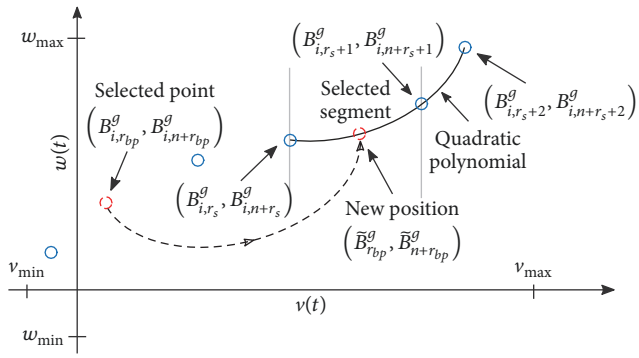

FIGURE 5: Mutation M.3 with $r_{bp} = 2$, $r_s = 4$, and $r_{3p} > 0.5$. Jump to the selected segment (dashed line). Quadratic polynomial (solid line).

(1) Initialise the population;
(2) Evaluate fitness of all population;
(3) **for** $g = 1$ *to* MaxGen **do**
(4) Find $\boldsymbol{\theta}^{\mathbf{g}}_{\mathbf{best}}$
(5) Random selection of a individual (r_1);
(6) Compute $\gamma(g)$;
(7) **if** $r_{pznl} \leq \gamma(g)$ **then**
(8) Compute $\widetilde{\mathbf{B}}^{\mathbf{g}}$ using Algorithm 2;
(9) **else**
(10) Compute $\widetilde{\mathbf{P}}^{\mathbf{g}}$ using Algorithm 3;
(11) **end if**
(12) **if** $r_c \leq \xi$ **then**
(13) Compute $\widetilde{\mathbf{C}}^{\mathbf{g}}$ using Mutation M.4;
(14) **end if**
(15) Update population;
(16) **end for**
(17) Print $\boldsymbol{\theta}^{\mathbf{MaxGen}}_{\mathbf{best}}$

ALGORITHM 1: Pseudocode of WH-EA.

individual $\mathbf{P}^{\mathbf{0}}_{\mathbf{1}}$ of the initial population. For the rest, mutation M.1 (16) is used but fixing the parent vector as $\mathbf{P}^{\mathbf{0}}_{\mathbf{1}}$, that is, the BLA.

Therefore, mutation M.1 must be executed $NP - 1$ times under the above conditions to generate mutated versions of the BLA. Initialization and mutation M.1 are performed considering a search space delimited by lower bound $\mathbf{P}^{\mathbf{min}}$ and upper bound $\mathbf{P}^{\mathbf{max}}$ which are set around the values of the BLA:

$$P^{\min}_j = P^0_{1,j} - \Upsilon^{\min}_j,$$
$$P^{\max}_j = P^0_{1,j} + \Upsilon^{\max}_j, \tag{27}$$

where $\Upsilon^{\min}_j$ and $\Upsilon^{\max}_j$ represent the jth elements of the user-defined vectors $\boldsymbol{\Upsilon}^{\mathbf{min}}$ and $\boldsymbol{\Upsilon}^{\mathbf{max}}$, respectively, indicating how much the BLA's pole/zero position can change as the algorithm evolves.

On the other hand, the initial population corresponding to two-dimensional points must be generated within a search space defined horizontally by the minimum ($v_{\min}$) and maximum ($v_{\max}$) amplitude of $v(t)$ (the output of $G_w(z)$) and vertically by the minimum ($w_{\min}$) and maximum ($w_{\max}$) amplitude of $w(t)$ (the input to $G_h(z)$). Although $v(t)$ and $w(t)$ are not known, $v_{\min}$ and $v_{\max}$ depend on the input $u(t)$ and $G_w(z)$. Since $G_w(z)$ is a linear system, the following expressions can be used:

$$v_{\min} = \Omega * u_{\min},$$
$$v_{\max} = \Omega * u_{\max}, \tag{28}$$

where $u_{\min}$ and $u_{\max}$ are the minimum and maximum values of the signal $u(t)$, respectively, and Ω is a scaling factor which depends on the pole/zero locations and the static gain of $G_w(z)$. Without loss of generality, it is possible to model a Wiener-Hammerstein system considering that both linear blocks have unit gain.

Regarding $w_{\min}$ and $w_{\max}$, if the input $u(t)$ with mean u_{mean} enters $G_w(z)$ regardless of its structure, the mean of the output signal v_{mean} will be equal to u_{mean}. This is not the case when the signal $v(t)$ enters the nonlinear block since the gain K_{NL} might provide an offset to $w(t)$. However, since $G_h(z)$ is a linear block, the mean of the output y_{mean} will be equal to the mean of the incoming signal w_{mean}. With this information, the straight line of Figure 6 can be drawn, and $w_{\min}$ and $w_{\max}$ can be found using the following equations:

$$w_{\min} = y_{\text{mean}} + K_{\text{NL}}\left(\Omega u_{\min} - u_{\text{mean}}\right),$$
$$w_{\max} = y_{\text{mean}} + K_{\text{NL}}\left(\Omega u_{\max} - u_{\text{mean}}\right). \tag{29}$$

As can be seen from (28) to (29), the search space for nonlinearity depends on the input signal, output signal, static gain of the BLA, and Ω which is a user-defined parameter. Since both linear subsystems will be estimated with unit gain, neither of these will amplify their input signals, therefore $v_{\min} > u_{\min}$ and $v_{\max} < u_{\max}$. For these two conditions to be met, Ω must be less than one. In the same way, it must be observed that $w_{\min} < y_{\min}$ and $w_{\max} > y_{\max}$. If Ω is less than one, the first pair of conditions will always be met; however, there is no guarantee that the second pair of conditions will be met. Since it is possible to perform this check prior to the execution of the algorithm, if the second pair of conditions are not met, Ω must be increased, but considering that it must be less than one. It should also be taken into account that if Ω is too large, the search space will be larger than necessary, so the algorithm will cost more to estimate static nonlinearity.

To initialize this portion of genetic information, the n points are uniformly distributed between $\Omega u_{\min}$ and $\Omega u_{\max}$ and located on the straight line shown in Figure 6. These points are introduced as part of the first individual in the population $\mathbf{B}^{\mathbf{0}}_{\mathbf{1}}$. The corresponding genetic information for the rest of individuals is generated using mutation M.2 (19), but considering that the parent vector is always $\mathbf{B}^{\mathbf{0}}_{\mathbf{1}}$.

Finally, genetic information corresponding to pole-zero classification is initialized directly using mutation M.4 (26) NP times.

Evaluate Fitness. Performance of each individual in the population is defined by a fitness criterion which can be calculated

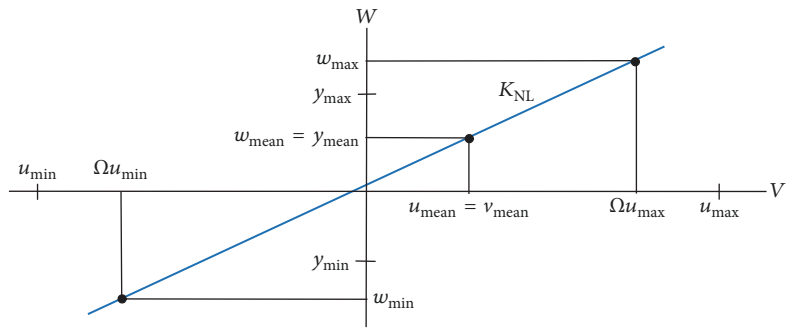

Figure 6: Information to define the search space for the nonlinear function.

using (14), where $\boldsymbol{\theta}$ is obtained from the encoded information in $[\mathbf{P}_{\mathbf{i}}^{\mathbf{g}}, \mathbf{B}_{\mathbf{i}}^{\mathbf{g}}, \mathbf{C}_{\mathbf{i}}^{\mathbf{g}}]$.

The Offspring. Once population has been initialized, for each generation a random integer number $r_1 \in [1, NP]$ will be used to select the parent from which an offspring $\widetilde{\mathbf{P}}^{\mathbf{g}}$ will be generated. As can be seen in Algorithm 1, not all genetic operators are applied at the same time to generate offspring; this can help to expand diversity and avoid premature convergence.

One or two pieces of the offspring genetic information will be randomly selected for modification according to their respective genetic operators. A random number $r_{pznl} \in (0, 1]$ chooses between modifying the portion related to static nonlinearity using Algorithm 2 or the portion of genetic information related to pole/zero locations using Algorithm 3. The probability for this selection is handled by the control parameter $\gamma(g)$ defined as

$$\gamma(g) = \frac{\gamma_{\text{ini}}}{\sqrt{1 + g * \gamma_{\text{rat}}}}, \qquad \gamma_{\text{rat}} = \frac{\left(\gamma_{\text{ini}}/\gamma_{\text{end}}\right)^2 - 1}{\text{MaxGen} - 1}, \tag{30}$$

where γ_{rat} is the rate at which the probability $\gamma(g)$ will decrease from initial probability γ_{ini} to final probability γ_{end} as generations pass; therefore, $0 < \gamma_{\text{end}} < \gamma_{\text{ini}} \leq 1$. If $\gamma_{\text{ini}} = 1$, the probability of modifying the genetic information of nonlinearity in the first generations will be high, while the probability of modifying the location of poles and zeros will be low. On the other hand, if and $\gamma_{\text{end}} = 0.5$, in the final generations, the algorithm will modify with equal probability both portions of genetic information. The selection of these values is justified by the fact that pole/zero locations are known and they will only be fine-tuned within a suitable search space to amend possible errors in the BLA estimation, whereas nonlinearity is completely unknown, so the algorithm should focus more on this portion of genetic information during first generations.

Variation of genetic information corresponding to the classification of poles and zeros for both LTI subsystems is handled by a comparison between a random number $r_c \in (0, 1]$ and the probability $\xi \in (0, 1]$. The value of probability ξ is defined by the user and will be constant throughout the evolution of the algorithm. Figure 7 shows the behaviour of the control parameters (probabilities) used to select the portions of genetic information that will be modified in each generation.

```
(1)  if r_nmc ≤ δ_nl then
(2)      Compute η(g)
(3)      if r_mm ≤ η_min + η(g) then
(4)          Mutation M.2;
(5)      else
(6)          Mutation M.3;
(7)      end if
(8)  else
(9)      Crossover C.2;
(10) end if
```

Algorithm 2: Modify two-dimensional points for nonlinear function.

```
(1) if r_lmc ≤ δ_zp then
(2)     Mutation M.1;
(3) else
(4)     Crossover C.1;
(5) end if
```

Algorithm 3: Modify polo/zero locations.

Algorithm 2 is used to modify the genetic information related to nonlinear static function. The control parameter $\delta_{nl} \in (0, 1]$ indicates the probability with which the mutation (either M.2 or M.3) or crossover C.2 will be used. Probability of selecting M.2 or M.3 is variable with respect to the generations. During first generations, mutation M.3 is not necessary, since the nonlinearity can be captured thanks to the two-dimensional points movements due to mutation M.2 and crossover C.2 operations. Since it is very likely that nonlinearity includes one or more curvatures, as the algorithm evolves mutation M.3 will be required to concentrate as many points as possible on these curvatures. The variable

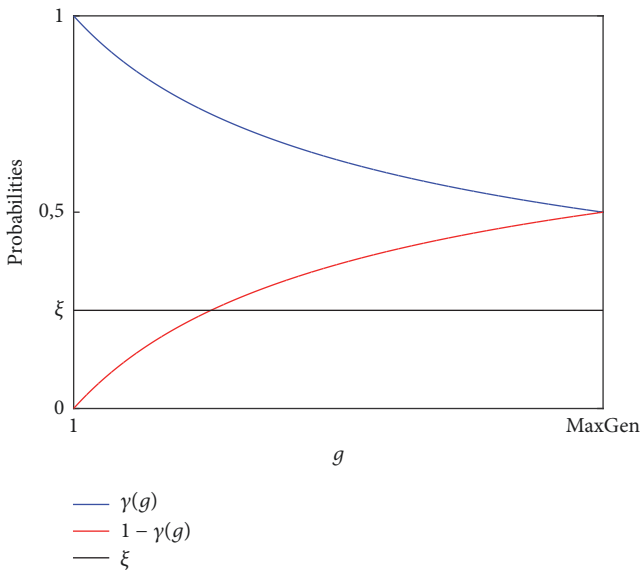

Figure 7: Control parameters of WH-EA for selection of the genetic information to be modified in generation g.

probability for selection between both mutations is defined by

$$\eta(g) = (1 - \eta_{\min}) - (1 - \eta_{\min}) \frac{g}{\text{MaxGen}}, \tag{31}$$

where $\eta_{\min} \in (0, 0.5]$ is a user-defined parameter indicating the minimum probability with which the mutation M.2 can be selected. Note that according to (31) and Algorithm 2, the maximum probability is 1 and occurs in the first generation. As the algorithm evolves, this probability will decrease linearly until it reaches $\eta_{\min}$ in the last generation. When Algorithm 2 is required a random number $r_{nmc} \in (0, 1]$ will allow us to select either a mutation or crossover C.2. If a mutation is selected, a new random number $r_{mm} \in (0, 1]$ will allow us to select between mutation M.2 or mutation M.3.

On the other hand, Algorithm 3 is used to modify the genetic information related to pole/zero locations using mutation M.1 or crossover C.1. The control parameter $\delta_{zp} \in (0, 1]$ indicates the probability with which each genetic operation will be used. Since crossover C.1 causes offspring to inherit genetic information from the best individual, a small value of δ_{zp} may lead to premature convergence, whereas a value closer to 1 will cause the algorithm to converge very slowly. When Algorithm 3 is required, a random number $r_{lmc} \in (0, 1]$ will determine the genetic operation to be used.

Update. It is based on a competition between the generated offspring and the individuals of the population. The contestant with the best fitness will be the one who wins the competition. From a randomly selected individual, the offspring starts to compete until defeating an individual; when this happens the descendant will take his place in the population and the algorithm continues with the next generation. If the offspring comes to compete with all individuals and could not win, this will be discarded and the algorithm will pass to the next generation.

4. Application of WH-EA and Results

WH-EA was tested on a numerical example and on the benchmark for nonlinear system identification in (SYSID'09) [9], where a Wiener-Hammerstein system is selected as test object. The benchmark is not intended as a competition, but as a tool to compare the possibilities of different methods to deal with this specific nonlinear structure.

For both cases, the BLA was estimated with the MATLAB System Identification Toolbox [60] using a Box-Jenkins (BJ) structure. Besides, trends and means were only removed for the BLA identification. The following parameters of the algorithm were set in common for both estimates: $\xi = 0.25$; $\delta_{zp} = 0.75$; $\delta_{nl} = 0.75$; $\eta_{\min} = 0.35$; in addition, initial and final standard deviations for mutations were set to 20 and 1, respectively.

4.1. Numerical Example. A Wiener-Hammerstein system with the following structure was designed (where tansig is the hyperbolic tangent sigmoid transfer function):

$$\begin{aligned} G'_w(z) &= \frac{0.1190}{(z - 0.9048)}, \\ w'(t) &= 0.45 \,\text{tansig}\left(2.80 v'(t)\right), \\ G'_h(z) &= -0.01426 \\ &\quad \cdot \frac{(z - 1.0510)(z + 1)}{(z - 0.9746 + 0.03656j)(z - 0.9746 - 0.03656j)}. \end{aligned} \tag{32}$$

A Gaussian excitation signal of 6 dB was filtered with a cut-off frequency of 6 Hz and used as input signal. The system was simulated and 120000 input/output samples were recorded and separated in two parts: the estimation data set $t_n \in [1001, 70000]$ for identification purposes and the test data set $t_n \in [71001, 120000]$ for validation purposes (in both data sets first 1000 samples were ignored to avoid transient effects). Furthermore, additive white Gaussian noise with a Signal-to-Noise Ratio (SNR) of 45.32 dB was added to the output.

The identification of the BLA was carried out and the model obtained was expressed in factored form:

$$\begin{aligned} G_{\text{BLA}} &= -1e^{-3} \\ &\quad \cdot \frac{(z - 1.0508)(z + 0.9631)}{(z - 0.9749 + 0.0366j)(z - 0.9749 - 0.0366j)(z - 0.9045)}. \end{aligned} \tag{33}$$

The root mean square of the error (eRMS) obtained with this linear model on test data was of 0.0414.

According to the BLA structure, vector $\mathbf{P}_1^0$ was coded with $nc = 0$, $nr = 2$, $mc = 1$, and $mr = 1$ as follows:

$$\mathbf{P}_1^0 = [1.0508, -0.9631, 0.9749, 0.9045, 0.0366]. \tag{34}$$

The search space for nonlinear function was defined with $\Omega = 0.28$, while elements of $\boldsymbol{\Upsilon}^{\min}$ and $\boldsymbol{\Upsilon}^{\max}$ were set to 0.01 (except bounds for the zero $z = -0.9631$ that were set to 0.1, since it influences slightly the dynamics and should have freedom of movement during tuning).

Table 1: Performance of the numerical example estimation using different numbers of points n to represent static nonlinearity.

n	NRMSE (%)	eRMS
8	99.179	$1.183e^{-3}$
10	99.396	$1.109e^{-3}$
12	99.564	$1.044e^{-3}$

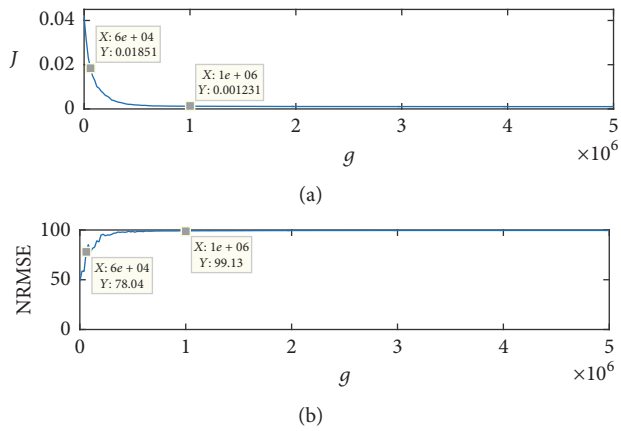

Figure 8: Convergence graph (a) and NRMSE of the captured nonlinearity (b) for Wiener-Hammerstein estimation with $n = 12$.

WH-EA was executed 3 times with MaxGen $= 5 \cdot 10^6$ and different number of points was chosen for the nonlinearity. For all trials, the algorithm was initialized with 60000 individuals and the minimum distance between two points was set to $\alpha = (v_{\max} - v_{\min})/6n$. For each estimated Wiener-Hammerstein model, the eRMS on the test data was computed; in addition, the normalized root mean square error (NRMSE) criterion was used to quantify the goodness of fit between real and captured nonlinearity. The results are reported in Table 1; for all cases piecewise linear interpolation was used to connect the n points.

The poles and zeros of the BLA were correctly classified in the three tests carried out. As can be seen in Table 1, the eRMS of the Wiener-Hammerstein models decreased as the quality of the captured nonlinearity increases. A reasonable model was scored with 12 points considering that the RMS noise was $9.98e^{-4}$. A convergence graph for this model is shown in Figure 8, during WH-EA execution, at 6e4th generation the poles and zeros of the BLA were correctly classified, and from there the best individual of each generation conserved the genetic information for this classification. Since the noise RMS is known, at $g = 1e6$ the performance of the model was good enough, so the algorithm could have been stopped. Anyway, 5e6 generations have been allowed in order to demonstrate the great precision that the algorithm can achieve.

In Figure 9, pole/zero locations of the BLA, the obtained Wiener-Hammerstein model, and real system are compared. Notice how WH-EA has moved initial locations trying to get to the true values improving modelling error.

On the other hand, a graphical comparison between real and captured nonlinearity is shown in Figure 10. Linear subsystems of the estimated Wiener-Hammerstein model are represented by (35), while the ordered pairs for the nonlinear static function are shown in Table 2.

$$G_w(z) = \frac{0.0259}{z - 0.9048},$$
$$G_h(z) = -0.01960 \cdot \frac{(z - 1.0512)(z + 0.9713)}{(z - 0.9746 + 0.0365j)(z - 0.9746 - 0.0365j)}. \quad (35)$$

4.2. Nonlinear System Identification Benchmark. The system to be modelled is an electronic nonlinear circuit with a Wiener-Hammerstein structure (see Figure 11). This system was built by Vandersteen [61] and presented as a benchmark problem for system identification by Schoukens et al. [9].

The first linear dynamic system $G_1(s)$ is designed as a third-order Chebyshev filter (pass-band ripple of 0.5 dB and cut-off frequency of 4.4 kHz). The second linear dynamic system $G_2(s)$ is a third-order inverse Chebyshev filter (stop-band attenuation of 40 dB starting at 5 kHz). This system has a transmission zero in the frequency band of interest. This can complicate the identification significantly, because the inversion of such a characteristic is difficult. The system was excited with a filtered Gaussian signal (cut-off frequency 10 kHz). Data used for estimation corresponds to interval $t_n \in [1, 100000]$, whereas test data corresponds to the remaining part $t_n \in [101001, 188000]$. In order to analyse the performance of estimation methods, the mean value of the simulation error (μ), the standard deviation of the error (std), and the root mean square value of the error (eRMS) must be calculated on test and estimation data [9].

Since first 5000 data samples just contain quantization noise, a set of 95000 input/output data $t_n = 5001, \ldots, 100000$ was used to estimate the BLA. Multiple simulations were performed considering different combinations of poles and zeros for the input/output model and for the noise model. For each BJ model, the eRMS on test data set $t_n = 101001, \ldots, 188000$ was computed. The BLA was obtained with 6 poles, 5 zeros, and one sample delay for the input/output model and 3 zeros and 3 poles for the noise model. The BLA is fully described with $K_{\mathrm{NL}} = 0.7840$ and the pole-zero pattern shown in Figure 12. The eRMS of this linear model was of 56.159 mV on test data and 43.143 mV after removing trends and means. According to the BLA structure, vector $\mathbf{P}_1^0$ was coded with $nc = 1$, $nr = 4$, $mc = 2$, and $mr = 2$ as follows:

$$\mathbf{P}_1^0 = [0.7605, -0.2733, 0, -3.4122, -30.2553, 0.6501, 0.7314, \ldots, 0.8912, 0.8289, 0.7004, 0.4358, 0.1692]. \quad (36)$$

During BLA estimation stage, different noise models were tested and it was observed that all poles, real zeros within the unitary circle, and complex zeros where they are located correspond to the dominant dynamics of the system, while real zeros outside the unitary circle were more likely to vary

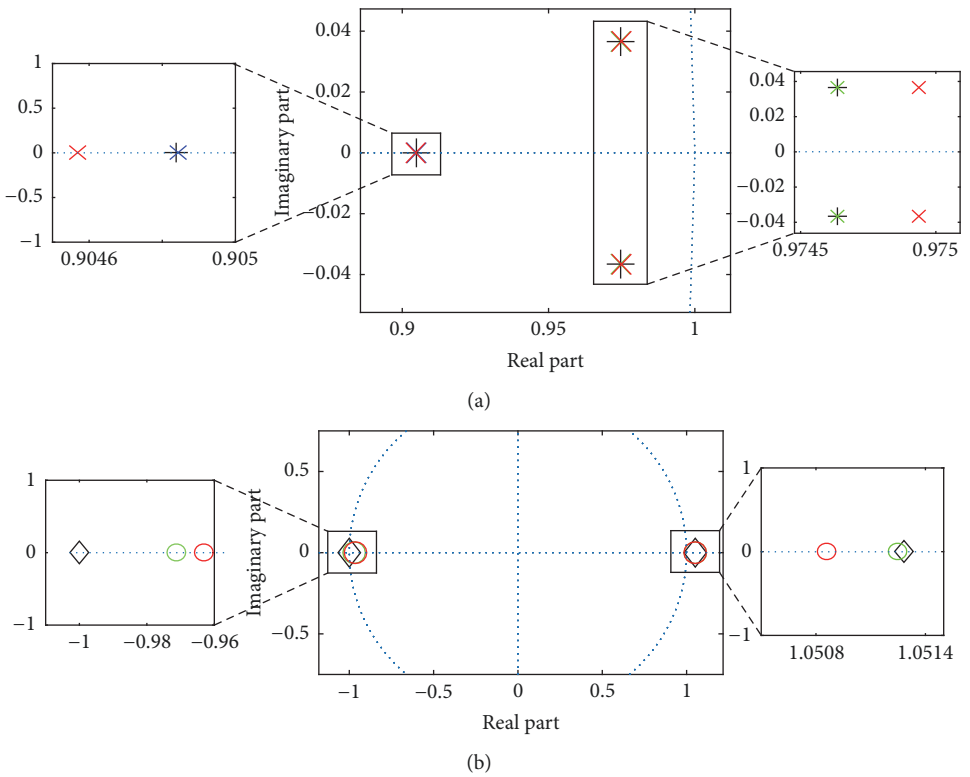

Figure 9: (a) Poles of the real system (black +), the BLA (red ×), and the estimated G_w (blue ×) and G_h (green ×) models. (b) Zeros of the real system (black ◇), the BLA (red o), and the estimated G_h model (green o).

Table 2: Coordinates of the estimated nonlinearity with $n = 12$.

i	1	2	3	4	5	6	7	8	9	10	11	12
v_i	−0.9694	−0.5794	−0.3716	−0.2479	−0.1251	0.1038	0.1996	0.2893	0.3363	0.4739	0.7648	1.0808
w_i	−0.3303	−0.3250	−0.2894	−0.2354	−0.1415	0.1178	0.2031	0.2558	0.2754	0.3126	0.3309	0.3317

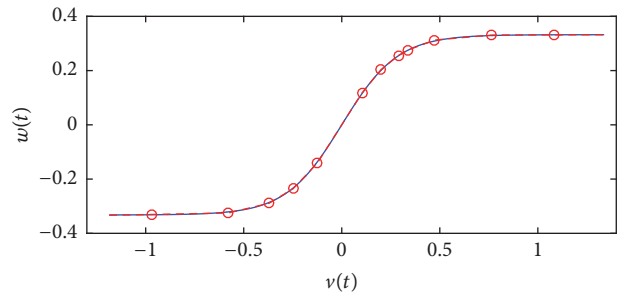

Figure 10: Comparison between true (solid-blue) and estimated nonlinearity defined as a piecewise linear function (dashed-red) using pairs $[v_i, w_i]$ (red circles).

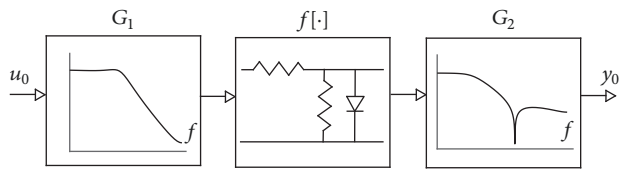

Figure 11: Wiener-Hammerstein benchmark.

their location. This information was used to define the search space for refining the location of poles and zeros:

$$\begin{aligned}\boldsymbol{\Upsilon}^{\min} &= [0.025, 0.025, 0.025, 1, 10, 0.025, 0.025, \ldots, \\ &\quad 0.025, 0.025, 0.025, 0.025, 0.025], \\ \boldsymbol{\Upsilon}^{\max} &= [0.025, 0.025, 0.025, 1, 10, 0.025, 0.025, \ldots, \\ &\quad 0.025, 0.025, 0.025, 0.025, 0.025].\end{aligned} \tag{37}$$

Bounds (37) limit search space in the system dominant dynamics within ±0.025, while for $z = -30.255$ and $z = -3.412$, limits are between ±10 and ±1, respectively.

Static nonlinearity is represented by piecewise linear functions with $n = 8$ points. Its search space was defined with $\Omega = 0.51$ and the minimum distance between two points was calculated with $\alpha = (v_{\max} - v_{\min})/10n$. The algorithm was initialized with 5000 individuals and $3e7$ generations were executed. The performance of the estimated Wiener-Hammerstein model is shown in Table 3. In Figure 13, it is depicted how the algorithm has distributed pole/zero locations for both linear subsystems. Notice how some of them were displaced to improve the modelling error. The

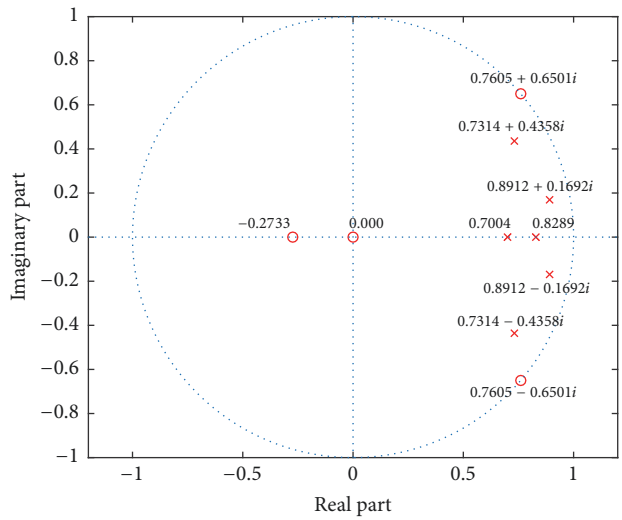

Figure 12: Poles (×) and zeros (o) of the BLA for the benchmark data (SYSID'09). Two real zeros fall outside the plot in −30.2553 and −3.4122.

outputs of the estimated Wiener-Hammerstein model, the linear model error, and the nonlinear model error on test data are shown in Figure 14, while the DFT spectra of this signals are shown in Figure 15. Captured nonlinearity is plotted in Figure 16.

Final estimated linear blocks $G_w(z)$ and $G_h(z)$ are shown in (38) and (39) respectively, while coordinates for nonlinear function are shown in Table 4. The estimated Wiener-Hammerstein model contains 26 parameters of which 14 are used to represent the static nonlinearity (without end points since they can be located anywhere on their respective end segments; nevertheless, these segments slopes are taken into account). Figure 16 shows how mutation M.2 and mutation M.3 located the two-dimensional points to capture the nonlinearity. As expected, due to the effect of mutation M.3, most of them were concentrated on the curvature.

$$G_w(z) = 6.5e^{-4} \cdot \frac{(z+0.0138)(z+2.9034)(z+26.76)}{(z-0.7243)(z-0.7324+0.4361j)(z-0.7324-0.4361j)}, \quad (38)$$

$$G_h(z) = 0.0120 \cdot \frac{(z+0.2635)(z-0.7575+0.6513j)(z-0.7575-0.6513j)}{(z-0.8191)(z-0.8899+0.1688j)(z-0.8899-0.1688j)}. \quad (39)$$

4.3. Discussion. In contrast to other methods which generate good initial estimates by splitting the poles and zeros of the BLA, WH-EA allows us to identify Wiener-Hammerstein models avoiding high user interaction which is an advantage compared to methods using QBLA, where at least two intermediate procedures are required before fine-tuning all parameters of the Wiener-Hammerstein model.

The *e*RMS of 0.306 mV achieved with WH-EA on test data is quite acceptable considering that the RMS of the quantization noise is 0.189 mV. With respect to the initial model (BLA), the error was reduced by a factor of 183.52 thanks to the captured nonlinearity and the updated pole/zero locations. Table 5 shows other proposals that have been tested on the benchmark. It can be appreciated that the *e*RMS of this paper is slightly higher than others; however, not all estimated models have the same complexity. Some of them use complex models with a greater number of parameters processing raw data before identification, while in this work, WH-EA is fed raw input/output data without preprocessing operations.

Comparing WH-EA with the proposals of Westwick and Schoukens [42] and Vanbeylen [43] whose models have the same complexity as the model estimated in this paper, the results are quite similar; however, to obtain a good final model with these two proposals, it is required that the BLA be estimated with high precision. In Vanbeylen [43] at the BLA division phase, a false position of a pole or zero could cause the values of the fractional powers to be close to 1/2 which would cause the user to make a bad decision and the BLA is badly divided. This problem is much more critical in Westwick and Schoukens [42] since the method is based on a graphical comparison between the poles and zeros of the BLA and the poles and zeros of the QBLA. With WH-EA, this problem does not occur, since the evolutionary algorithm contemplates possible errors that can be made in the estimation of the BLA. During the algorithm evolution, the binary code used for the classification of the poles and zeros of the BLA can be changed without user interaction as the false positions of the poles and zeros are corrected. This is an important advantage of WH-EA, since it is very likely that the BLA estimate is subject to errors due to noise and nonlinearity effects; this has been experimentally demonstrated; for this reason, many proposals carry out a final readjustment of the parameters of the Wiener-Hammerstein model.

About the computational complexity, an iteration in WH-EA involves the random selection of an individual from the population, the offspring generation according to the genetic operations indicated in Section 3, the calculation of the objective function of the offspring, and the population update. This algorithm was implemented on MATLAB® software and was run on a personal computer with core i7 processor of 2,6 GHz and 16 Gb of RAM. The objective function was easily implemented using the filter command to simulate both linear subsystems, while the nonlinearity was interpolated using the interp1 command. For the benchmark system, the average time consumed per 50 iterations was 1.16 s.

5. Conclusions

In this paper, a new method (WH-EA) to identify Wiener-Hammerstein systems in a single step is proposed. The proposal estimates all parameters of the Wiener-Hammerstein model based on a customized evolutionary algorithm (WH-EA). Unlike conventional procedures, WH-EA is able to look for the best BLA split capturing at the same time the process static nonlinearity with high precision, solving a single optimization problem. The algorithm is fed with the estimated BLA and its pole/zero locations are subtly modified within an adequate search space to allow its fine-tuning, while piecewise linear function is used for the nonlinear block. The performance of this approach has been evaluated through

Table 3: Performance indicators of the estimated Wiener-Hammerstein model. All values are shown in mV.

	BLA		Wiener-Hammerstein	
	Estimation	Test	Estimation	Test
μ	−35.825	−35.951	$5.6e^{-4}$	$1.1e^{-5}$
std	42.108	43.143	0.322	0.306
eRMS	55.286	56.159	0.322	0.306

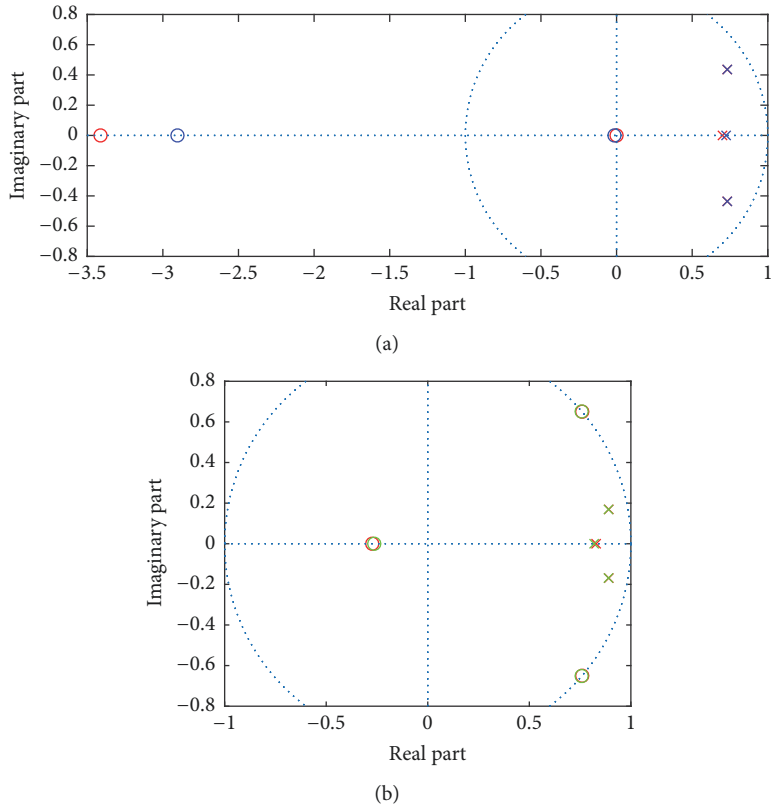

Figure 13: (a) Poles (×) and zeros (o) of the linear subsystem G_w (blue) and BLA (red). Zero in $z = -30.2553$ belonging to BLA and its adjusted final value in $z = -26.7609$ fall outside the plot. (b) Poles (×) and zeros (o) of the linear subsystem G_h (green) and BLA (red).

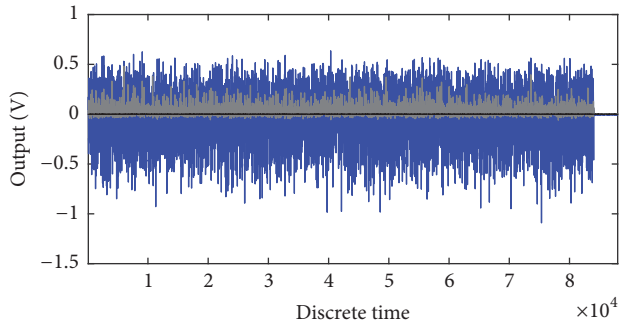

Figure 14: Model output (blue), simulation error of the BLA (grey), and simulation error of the estimated Wiener-Hammerstein model (black).

a numerical example with a complex static nonlinearity and through the well-known benchmark data (SYSID'09). The results show that it is possible, using WH-EA, to identify a Wiener-Hammerstein system with a good precision in a parametric framework avoiding high user interaction and drawbacks involved in using the QBLA. Further research will be related to WH-EA extension for nonlinear multivariable systems by using a multiobjective optimization approach.

WH-EA Parameters

nc: Number of pairs of complex conjugate zeros of the BLA
nr: Number of real zeros of the BLA
mc: Number of pairs of complex conjugate poles of the BLA
mr: Number of real poles of the BLA
n: Number of points to represent nonlinearity
K_{NL}: Static gain of the BLA

Table 4: Nonlinearity coordinates ($n = 8$, 14 parameters) estimated by WH-EA from benchmark data.

i	1	2	3	4	5	6	7	8
v_i	−0.2168	0.1596	0.3819	0.4943	0.6047	0.7596	1.0811	1.3953
w_i	−0.1979	0.1440	0.3443	0.4276	0.4822	0.5248	0.5698	0.5901

Table 5: Performance measurements on benchmark data (SYSID'09). All the values are shown in mV. θ indicates the number of parameters used for the model.

Method/technique	eRMS (mV)	θ
Nonparametric BLA, QBLA [41]	0.278	44
Classification of poles and zeros using QBLA [42]	0.286	26
Fractional model parameterization [43]	0.295	26
Advanced method [44, 45]	0.30	64
WH-EA (this paper)	*0.306*	*26*
Brute force method [45]	0.31	30
Scanning technique [46]	0.370	-
Polynomial nonlinear state space [47]	0.42	797
Generalized Hammerstein-Wiener [48]	0.481	47
Incremental nonlinear optimization [49]	0.679	25
LS-SVMs [50]	4.070	-
Biosocial culture [51]	8.546	34

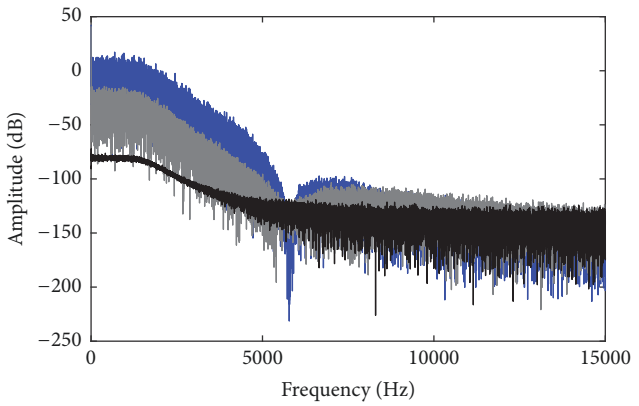

Figure 15: DFT spectra of the modelled output signal (blue), linear model error (grey), and nonlinear model error (black).

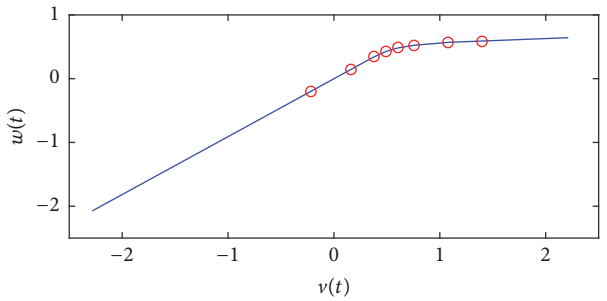

Figure 16: Captured nonlinearity as a piecewise linear function with $n = 8$ by WH-EA from the benchmark data. Notice that the nonlinear block characterization only needs 14 parameters since the first and last straight segments can be defined just with their angles.

$\sigma^2_{\text{ini}}, \sigma^2_{\text{end}}$: Initial and final standard deviations for control of the aggressiveness of mutations M.1 and M.2 ($\sigma^2_{\text{ini}} > \sigma^2_{\text{end}}$)

σ^2_{ratio}: Rate with which the standard deviation $\sigma(g)^2$ decreases from σ^2_{ini} to σ^2_{end}

Δ_s: Space over which a gene can move

α: Minimum distance between two points on the abscissa axis for mutations M.2 and M.3

Ω: Scale factor to define the search space for static nonlinearity

MaxGen: Generations number

NP: Population size

δ_{zp}: Control parameter for selection between mutation M.1 or crossover C.1

δ_{nl}: Control parameter for selection between mutation (either M.2 or M.3) or crossover C.2

$\eta(g)$: Variable probability for selection between mutation M.2 or mutation M.3

$\eta_{\min}$: Minimum probability for selection of mutation M.2. The maximum probability for this selection is 1

$\gamma(g)$: Variable probability to choose between modifying static nonlinearity or location of poles and zeros

γ_{rat}: Rate at which $\gamma(g)$ will decrease from initial probability $\gamma_{\text{ini}} = 1$ to final probability $\gamma_{\text{end}} = 0.5$

ξ: Probability to modify the genetic information related to the classification of poles and zeros.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was partially supported by the Spanish Ministry of Economy and Competitiveness (Project DPI2015-71443-R) and Salesian Polytechnic University of Ecuador through a Ph.D. scholarship granted to the first author.

References

[1] J. Schoukens and R. Pintelon, *Identification of Linear Systems: A Practical Guideline to Accurate Modeling*, Elsevier, 2014.

[2] L. A. Mora and J. E. Amaya, "A new identification method based on open loop step response of overdamped system," *RIAI - Revista Iberoamericana de Automatica e Informatica Industrial*, vol. 14, no. 1, pp. 31–43, 2017.

[3] T. Liu, Q.-G. Wang, and H.-P. Huang, "A tutorial review on process identification from step or relay feedback test," *Journal of Process Control*, vol. 23, no. 10, pp. 1597–1623, 2013.

[4] P. van Overschee and B. de Moor, *Subspace Identification for Linear Systems: Theory Implementation Applications*, Springer Science & Business Media, Germany, 2012.

[5] L. Ljung, *System Identification: Theory for the User, ptr prentice Hall Information and System Sciences Series*, Prentice-Hall, NJ, USA, 1999.

[6] R. S. Esfandiari and B. Lu, *Modeling and Analysis of Dynamic Systems*, CRC Press, 2014.

[7] D. C. Karnopp, D. L. Margolis, and R. C. Rosenberg, *System Dynamics: Modeling, Simulation, and Control of Mechatronic Systems*, John Wiley & Sons, 5th edition, 2012.

[8] J. Bonilla, L. Roca, A. De La Calle, and S. Dormido, "Dynamic model of a molten salt-gas heat recovery system for a hybrid renewable solar thermal power plant," *RIAI - Revista Iberoamericana de Automatica e Informatica Industrial*, vol. 14, no. 1, pp. 70–81, 2017.

[9] J. Schoukens, J. Suykens, and L. Ljung, "Wiener-hammerstein benchmark," in *In 15th IFAC Symposium on System Identification, Saint-Malo*, France, July, 2009.

[10] T. Wigren and J. Schoukens, "Three free data sets for development and benchmarking in nonlinear system identification," in *Proceedings of the 12th European Control Conference, ECC*, pp. 2933–2938, July 2013.

[11] M. Schoukens, P. Mattson, T. Wigren, and J. Noël, Cascaded tanks benchmark combining soft and hard nonlinearities, workshop on nonlinear system identification benchmarks, Brussels, Belgium, 2016.

[12] J. P. Noël and M. Schoukens, Hysteretic benchmark with a dynamic nonlinearity, workshop on nonlinear system identification benchmarks, pp. 7-14, Brussels, Belgium, 2016.

[13] B. De Moor, P. De Gersem, B. De Schutter, and W. Favoreel, "Daisy: a database for identification of systems," Journal A 3, 1997.

[14] G. P. Liu, *Nonlinear Identification And Control: A Neural Network Approach*, Springer Science Business Media, 2012.

[15] F. Giri and E. W. Bai, *Block-Oriented Nonlinear System Identification*, Springer, Berlin, Germany, 2010.

[16] A. Janczak, *Identification of Nonlinear Systems Using Neural Networks and Polynomial Models: A Block-Oriented Approach*, vol. 310, Springer Science & Business Media, 2004.

[17] F. J. Doyle, R. K. Pearson, and B. A. Ogunnaike, *Identification and Control Using Volterra Models*, Springer, New York, NY, USA, 2012.

[18] S. A. Billings, *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*, John Wiley & Sons, Chichester, UK, 2013.

[19] J. J. E. Oviedo, J. P. Vandewalle, and V. Wertz, *Fuzzy logic, Identification and Predictive Control*, Springer Science & Business Media, 2006.

[20] J. A. Suykens and J. P. Vandewalle, *Nonlinear Modeling: Advanced Black-Box Techniques*, Springer Science Business Media, New York, NY, USA, 2012.

[21] S. A. Billings and S. Y. Fakhouri, "Identification of systems containing linear dynamic and static nonlinear elements," *Automatica*, vol. 18, no. 1, pp. 15–26, 1982.

[22] P. Lopes dos Santos, J. A. Ramos, and J. L. Martins de Carvalho, "Identification of a benchmark wiener-hammerstein: a bilinear and hammerstein-bilinear model approach," *Control Engineering Practice*, vol. 20, no. 11, pp. 1156–1164, 2012.

[23] A. Kalafatis, N. Arifin, L. Wang, and W. R. Cluett, "A new approach to the identification of pH processes based on the wiener model," *Chemical Engineering Science*, vol. 50, no. 23, pp. 3693–3701, 1995.

[24] F. Jurado, "A method for the identification of solid oxide fuel cells using a hammerstein model," *Journal of Power Sources*, vol. 154, no. 1, pp. 145–152, 2006.

[25] S. Boubaker, "Identification of nonlinear hammerstein system using mixed integer-real coded particle swarm optimization: application to the electric daily peak-load forecasting," *Nonlinear Dynamics*, vol. 90, no. 2, pp. 797–814, 2017.

[26] M. S. Gaya, M. U. Zango, L. A. Yusuf et al., "Estimation of turbidity in water treatment plant using hammerstein-wiener and neural network technique," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 5, no. 3, pp. 666–672, 2017.

[27] A. Buscarino, C. Corradino, L. Fortuna, M. L. Apicella, G. Mazzitelli, and M. G. Xibilia, "Temperature model identification on FTU liquid lithium limiter," in *Proceedings of the 42nd Conference of the Industrial Electronics Society, IECON 2016*, pp. 6380–6384, October 2016.

[28] M. R. S. Mohammadi, R. Matos, and C. Rebelo, "Bobtail bolt preload loss in wind turbine tower prototype: Hammerstein-wiener identification model," *ce/papers*, vol. 10, pp. 175–184, 2017.

[29] E.-W. Bai, Z. Cai, S. Dudley-Javoroski, and R. K. Shields, "Application of wiener-hammerstein system identification in electrically stimulated paralyzed skeletal muscle modeling," in *Proceedings of the 47th IEEE Conference on Decision and Control, CDC 2008*, pp. 3305–3310, Mexico, December 2008.

[30] E.-W. Bai, Z. Cai, S. Dudley-Javorosk, and R. K. Shields, "Identification of a modified wiener-hammerstein system and its application in electrically stimulated paralyzed skeletal muscle modeling," *Automatica*, vol. 45, no. 3, pp. 736–743, 2009.

[31] O. P. Dewhirst, D. M. Simpson, N. Angarita, R. Allen, and P. L. Newland, "Wiener-Hammerstein parameter estimation using differential evolution," in *International Conference on Bio-inspired Systems and Signal Processing*, pp. 271–276, January, 2010.

[32] J. A. Oliver, R. Prieto, J. A. Cobos, O. García, and P. Alou, "Hybrid wiener-hammerstein structure for grey-box modeling of dc-dc converters," in *Proceedings of the 24th Annual IEEE Applied Power Electronics Conference and Exposition, APEC*, pp. 280–285, USA, February 2009.

[33] A. Haryanto and K.-S. Hong, "Maximum likelihood identification of wiener-hammerstein models," *Mechanical Systems and Signal Processing*, vol. 41, no. 1-2, pp. 54–70, 2013.

[34] I. Benyó, J. Kovács, J. Mononen, and U. Kortela, "Modelling of steam temperature dynamics of a superheater," *International Journal of Simulation: Systems, Science and Technology*, vol. 6, no. 6, pp. 3–9, 2005.

[35] M. J. Korenberg and I. W. Hunter, "The identification of nonlinear biological systems: Lnl cascade models," *Biological Cybernetics*, vol. 55, no. 2-3, pp. 125–134, 1986.

[36] J. C. Gómez, A. Jutan, and E. Baeyens, "Wiener model identification and predictive control of a pH neutralisation process," *IEE Proceedings Control Theory and Applications*, vol. 151, no. 3, pp. 329–338, 2004.

[37] S. Li and Y. Li, "Model predictive control of an intensified continuous reactor using a neural network Wiener model," *Neurocomputing*, vol. 185, pp. 93–104, 2016.

[38] M. Ławryńczuk, "Nonlinear predictive control for Hammerstein–Wiener systems," *ISA transactions*, vol. 55, pp. 49–62, 2015.

[39] Q. Zhang, Q. Wang, and G. Li, "Nonlinear modeling and predictive functional control of hammerstein system with application to the turntable servo system," *Mechanical Systems and Signal Processing*, vol. 72-73, pp. 383–394, 2016.

[40] M. Ławryńczuk, "Nonlinear predictive control of dynamic systems represented by wiener–hammerstein models," *Nonlinear Dynamics*, vol. 86, no. 2, pp. 1193–1214, 2016.

[41] M. Schoukens, R. Pintelon, and Y. Rolain, "Identification of wiener-hammerstein systems by a nonparametric separation of the best linear approximation," *Automatica*, vol. 50, no. 2, pp. 628–634, 2014.

[42] D. T. Westwick and J. Schoukens, "Classification of the poles and zeros of the best linear approximations of wiener-hammerstein systems," in *Proceedings of the Universite Libre de Bruxelles*, pp. 470–475, Belgium, July 2012.

[43] L. Vanbeylen, "A fractional approach to identify wiener-hammerstein systems," *Automatica*, vol. 50, no. 3, pp. 903–909, 2014.

[44] L. Lauwers, *Some practical applications of the best linear approximation in nonlinear block-oriented modelling*, Vrije Universiteit Brussel, 2011.

[45] J. Sjöberg, L. Lauwers, and J. Schoukens, "Identification of Wiener-Hammerstein models: two algorithms based on the best split of a linear model applied to the SYSID'09 benchmark problem," *Control Engineering Practice*, vol. 20, no. 11, pp. 1119–1125, 2012.

[46] D. T. Westwick and J. Schoukens, "Initial estimates of the linear subsystems of wiener-hammerstein models," *Automatica*, vol. 48, no. 11, pp. 2931–2936, 2012.

[47] J. Paduart, L. Lauwers, R. Pintelon, and J. Schoukens, "Identification of a Wiener-Hammerstein system using the polynomial nonlinear state space approach," *Control Engineering Practice*, vol. 20, no. 11, pp. 1133–1139, 2012.

[48] A. Wills and B. Ninness, "Generalised hammerstein-wiener system estimation and a benchmark application," *Control Engineering Practice*, vol. 20, no. 11, pp. 1097–1108, 2012.

[49] A. H. Tan, H. K. Wong, and K. Godfrey, "Identification of a Wiener-Hammerstein system using an incremental nonlinear optimisation technique," *Control Engineering Practice*, vol. 20, no. 11, pp. 1140–1148, 2012.

[50] T. Falck, K. Pelckmans, J. A. K. Suykens, and B. De Moor, "Identification of wiener-hammerstein systems using LS-SVMs," *IFAC Proceedings Volumes*, vol. 420, no. 10, pp. 820–825, 2009.

[51] A. Naitali and F. Giri, "Wiener-hammerstein system identification-an evolutionary approach," *International Journal of Systems Science*, vol. 47, no. 1, pp. 45–61, 2016.

[52] M. Schoukens and K. Tiels, "Identification of block-oriented nonlinear systems starting from linear approximations: a survey," *Automatica*, vol. 85, pp. 272–292, 2016.

[53] L. Ljung, "Estimating Linear Time-Invariant models of nonlinear time-varying systems," *European Journal of Control*, vol. 7, no. 2-3, pp. 203–219, 2001.

[54] M. Enqvist and L. Ljung, "Linear approximations of nonlinear FIR systems for separable input processes," *Automatica*, vol. 41, no. 3, pp. 459–473, 2005.

[55] J. Schoukens, R. Pintelon, T. Dobrowiecki, and Y. Rolain, "Identification of linear systems with nonlinear distortions," *Automatica*, vol. 41, no. 3, pp. 491–504, 2005.

[56] J. Schoukens, R. Pintelon, and Y. Rolain, *Mastering system identification in 100 exercises*, John Wiley & Sons, 2012.

[57] R. Pintelon and J. Schoukens, *System Identification: A Frequency Domain Approach, Second Edition*, John Wiley & Sons, 2012.

[58] J. Schoukens, J. Lataire, R. Pintelon, G. Vandersteen, and T. Dobrowiecki, "Robustness issues of the best linear approximation of a nonlinear system," *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 5, pp. 1737–1745, 2009.

[59] J. Bussgang, *Crosscorrelation Functions of Amplitude-Distorted Gaussian Signals. Research Lab. Electron*, 1952.

[60] L. Ljung, *System Identification Toolbox for Use with MATLAB*, 2007.

[61] G. Vandersteen, *Identification of Linear and Nonlinear Systems in An Errors-In-Variables Least Squares and Total Least Squares Framework*, Vrije Universiteit Brussel, Belgium, 1997.