mathematics

Article

Nonlinear Identification for Control by Using NARMAX Models

Dan Stefanoiu [1,2], Janetta Culita [1,*], Andreea-Cristina Voinea [1] and Vasilica Voinea [1]

1 Faculty of Automatic Control and Computers, National University of Science and Technology "Politehnica" Bucharest, 313 Splaiul Independentei, 060042 Bucharest, Romania; dan.stefanoiu@upb.ro (D.S.); andreea.voinea@stud.acs.upb.ro (A.-C.V.); vasilica.voinea@upb.ro (V.V.)
2 Academy of Romanian Scientists, 3 Ilfov Street, 050044 Bucharest, Romania
* Correspondence: janetta.culita@upb.ro

Abstract: The identification (and control) of nonlinear systems is one of the most important and actual research directions. Moreover, many systems are multivariable. Different from linear system identification (where only a few classes of models are available), in the case of nonlinear systems, the class set of models is quite diverse. One of the most appealing nonlinear models belongs to the nonlinear ARMAX (NARMAX) class. This article focusses on the identification of such a model, which can be compared with other models (such as nonlinear ARX (NARX) and linear ARMAX) in an application based on the didactical installation ASTANK2. The mathematical foundation of NARMAX models and their identification method are described at length within this article. One of the most interesting parts is concerned with the identification of optimal models not only in terms of numerical parameters but also as structure. A metaheuristic (namely, the Cuckoo Search Algorithm) is employed with the aim of finding the optimal structural indices based on a special cost function, referred to as *fitness*. In the end, the performances of all three models (NARMAX, NARX, and ARMAX) are compared after the identification of the ASTANK2 installation.

Keywords: linear/nonlinear system identification; multivariable systems; identification for control; simulation of identification models; nonconvex optimization using biology inspired metaheuristics

MSC: 93B30; 37M05; 90C26; 92-10

check for updates

Citation: Stefanoiu, D.; Culita, J.; Voinea, A.-C.; Voinea, V. Nonlinear Identification for Control by Using NARMAX Models. *Mathematics* **2024**, *12*, 2252. https://doi.org/10.3390/math12142252

Academic Editor: Simeon Reich

Received: 14 June 2024
Revised: 12 July 2024
Accepted: 15 July 2024
Published: 19 July 2024

1. Introduction

The complexity and diversity of real-life systems approached in many branches of engineering and science require developing accurate mathematical models as a fundamental step in supporting a large panoply of applications, such as automatic control, optimization, prediction, fault diagnosis, analysis, and simulation.

One approach to obtain accurate models of a system with observable outputs is by employing system identification techniques. *System Identification* (SI) [1,2], also nowadays referred to as *data-driven modeling*, is an interdisciplinary domain that aims to build mathematical models of artificial systems and natural phenomena from inputoutput observations [3]. Because most practical systems have an intrinsic nonlinear behavior [4], with different degrees of nonlinearity, and linear models cannot always capture their rich dynamics [5], increased attention has been paid, in the last two decades, to *Nonlinear System Identification* (NSI) [6–8]. Therefore, NSI is a challenging topic in modern research, with applications in many fields, such as chemical engineering [9–13], power systems [14,15], robotics [16–18], mechanics [19–24], fluid flows [25], agriculture [26], aeronautics [3], genetics [27], neuroscience [28,29], meteorology [30], signal processing [7], and so many others.

Assuming that the data acquisition has already been performed, NSI involves three stages [7]: first, one has to select the class (type) of the nonlinear models; second, and the most critical, an adequate structure of the nonlinear model has to be determined, followed by the parameter estimation; finally, the previously identified model will be tested on the validation dataset, and, if valid (unbiased and correct) [25], it is returned as being reliable for further applications. There are some popular classes of nonlinear models that have been addressed in the SI literature over the years, most of them for the discrete-time domain, such as Volterra series [31,32], block-structured systems (like Hammerstein) [4,10,33], Wiener class [32], and WienerHammerstein class [21].

One model that has captured the interest of many scientists is the *Nonlinear Auto-Regressive Moving Average with eXogenous input* (NARMAX) one [6,8,20,34,35], with its simpler versions, NARX and NARMA [3,13,21,25,28,36]. In fact, the NARMAX models are naturally built by using the well-known ARMAX linear models [1].

The NARMAX model has proven to be versatile and to perform very well in many applications. It is based on a comprehensive mathematical description accommodating a wide range of nonlinear stochastic and deterministic systems [20,25,34,37]. Although the output is expressed by a nonlinear relationship of the lagged variable signals (inputs, outputs, and noise), the model is often linear in parameters, which makes it identifiable [8]. Although introduced a long time ago by Billings and Leontaritis [6], this model has been extensively documented and developed in the last two decades in different configurations, such as power-form polynomial models [22], rational models [38], neural networks [14,16,19,22], radial basis functions [39], wavelet networks [12,40], adaptive neuro-fuzzy inference systems [24], and models based on fuzzy logic [41]. Among these, polynomial models are one of the most systematically studied [16], having some advantages over non-polynomial ones (like, for example, machine learning models [26]): they are easy to interpret, describe an explicit relationship between the system variables (signals), and their identification methods are efficient and applicable on a relatively reduced dataset. On the other hand, non-polynomial models have the advantage of overlooking the problem of the model structure detection [37], which is the most critical task.

In polynomial representation, the model structure generally refers to the number and degrees of the polynomials associated to the model terms (or regressors) or, more specifically, to the degree of nonlinearity and the lags of the variables in the model terms. If at least some preliminary information about the system structure is available, the parameter estimation can simply be carried out afterwards. Otherwise, different selection criteria should be employed to search for the optimal structure that has to accomplish a trade-off between the accuracy and parsimony of the model. Consequently, the most critical step in NSI is to find the most appropriate model (structure) that fits the underlying system, including the noise dynamics [5]. In practical systems, noise handling is not a trivial task, especially for complex nonlinear systems. Depending on the complexity of the system generating the measured data, one can choose linear or nonlinear noise representations. For example, in [8], a nonlinear model of the noise has been proposed to better identify a satellite behavior by using real-life data.

Various efficient identification methods, mainly based on regressor orthogonalization, have been widely adopted to estimate the parameters of the NARMAX polynomial models, such as *Orthogonal Least Squares* (OLS) [25,38], Forward Regression OLS [9,22,25,26,42] or Orthogonal Forward Regression [20], Multiple Forward Regression OLS [43], *Extended Least Squares* (ELS) [1,15], and Recursive ELS [17,34,38]. These methods need to be performed within an optimization framework to find the optimal model structure of the nonlinear system. Therefore, the selection of the regressor set (or model terms) interleaves with the parameter estimation [44].

To obtain the most contributing model terms, two types of optimization methods have been widely applied: classical and nonconventional. The methods of the first type rely on an exhaustive search of the most significant model terms, usually chosen from a very large dictionary of identification models. The model terms are evaluated, ranked, and selected at each iteration by specific indicators, such as *Error Reduction Ratio* (ERR) [25,42], Sum of ERR applied separately or in combination with the Bayesian information criterion [13,25,40], Akaike information criteria [10,25,42], final prediction error [25], NashSutcliffe Efficiency [10], Mean Square Error [4,13], Root Mean Square Error [10], and R^2 [22].

Finally, the model is generated with the selected model terms whose metric is greater than a predefined threshold [20]. This approach proves to be practical, although it leads to time-consuming procedures as soon as the searching space increases. A detailed description of these methods can be found in [8].

Nonconventional optimization methods have captured considerable attention in modern research by introducing stochastic population-based Evolutionary Algorithms [4]. They are meant to solve complex optimization problems from real-life engineering [36] and are seen as an alternative to classical (exact) optimization methods. These optimization methods, also known as *metaheuristics*, have the potential to return a suitable solution from extended searching domains with reasonable computational effort. Various metaheuristics have been successfully utilized to select the structure of NARX/NARMAX models, such as *Particle Swarm Optimization* (PSO) [23,40], Genetic Algorithms [13,37], Multi-objective Optimization Differential Evolution [36], and Multi-Population Parallel Genetic Programming [45]. In [23], the authors proved that the application of PSO for NARX identification led to comparable results with OLS. In [45], the authors demonstrated that genetic programming can effectively handle complex stochastic NSI problems.

This paper focuses on the identification of a real-life laboratory fluidic system named ASTANK2 [46], by using a multivariable NARMAX model. We assumed that despite the nonlinear plant dynamics, the noise corrupting the data can be modeled by a linear ARMA model, without loss of generality. One of the main challenges in multivariable system identification is to obtain a precise model for each output channel, despite their correlation. Few papers tackle multivariable practical applications. One can mention here [11,20,21], although in [20], one operates with subsystems with single inputs and decoupled outputs. In this paper, a NARMAX model with two inputs and two outputs has been identified for the ASTANK2 installation, by using the *Multi-Step Extended Orthogonal Least Squares* (MS-EOLS) method (with GramSchmidt orthogonalization). At each iteration, the regressors and outputs are filtered by an autoregressive filter, to refine the auxiliary model and, consequently, to enhance the global model accuracy. The model structure consists of both the number of polynomials associated to the signal products (the *major* indices) and their corresponding degrees (the *minor* indices). An optimization criterion referred to as *fitness* is defined such that the optimal NARMAX model matches the identification as well as the validation datasets, on both output channels, simultaneously. Due to the lack of information about the structural indices and the fractal nature of the fitness (induced by stochastic disturbances), a metaheuristic has been employed to obtain the optimal structure of the model, namely, the *Cuckoo Search Algorithm* (CSA) [47]. The CSA was used in hierarchical manner, such that for each combination of major indices, the optimal combination of minor indices is found. The optimal model is, thus, determined to have the best major and minor indices all at once.

As already known, the accuracy of the noise model should lead to unbiased estimates of the system parameters and good precision of *one-step prediction* (osp) and/or *multi-step prediction* (msp) of the system output [42]. This proves the reliability of the system model over a long-range prediction horizon [44]. Nevertheless, such a goal is difficult to reach for, because, in msp (where predicted instead of measured output values are employed to forecast the current output), the errors progressively accumulate [22] and can lead to biased output. In [22], the authors preferred to use the osp while assessing the model quality, since msp did not provide good results. In [28], the osp performances surpassed those of the msp with three lags for multiple datasets. In this paper, the proposed optimal NARMAX model was determined by using msp. A simpler model, of NARX type, was considered as well. Such models are more reliable in an automatic control application, when no *input/output* (i/o) data are available in advance for measuring, compared to optimal models that provide accurate osps. To be fair, a comparative analysis between optimal NARMAX, NARX, and ARMAX models was made. The obtained results revealed that the NARMAX model is superior to NARX model, which, in turn, has better performance than ARMAX model, in terms of accuracy.

The novelty of the approach within this article consists of a practical (and quite general) methodology for the identification of multivariable nonlinear systems (represented here by NARMAX model class), in view of automatic control, by employing multi-step prediction. It is well known that for control applications, the measured input/output data are not known a priori. The design of the optimal controller is only based on the previously identified model and, consequently, on the simulated output data. Therefore, the long-term simulation of the identification model by multi-step prediction becomes crucial, especially when the noise component must be simulated (as a part of the model). To the best of our knowledge, most of the results reported in the literature devoted to SI employ one-step prediction as the simulation technique of the identified models. Or, such models are less appropriate to be integrated into a closed loop application than the ones identified through multi-step prediction.

This article focusses on the following topics. Section 2 describes, in the first subsection, the proposed identification model of NARMAX type, with 2 input and 2 output channels. Within the next subsection, the parameter identification method, i.e., the MS-EOLS, is explained. The third and fourth subsections are devoted to the determination of the optimal structural indices that characterize the NARMAX model. Here, after defining the fitness as cost function, the optimization problem is formulated, and the CSA is briefly described. Section 3 gives some implementation details of numerical algorithms designed based on the previous section. Section 4 reveals the simulation results and a comparative study on the performances of the optimal NARMAX, NARX, and ARMAX models. Section 5 concludes the paper and proposes some future research directions.

2. Theoretical Background

2.1. A NARMAX Model with 2 Inputs and 2 Outputs

Before describing the NARMAX models, recall the equations of linear ARMAX model (well known to the SI community) in case of a *Single-Input-Single-Output* (SISO) system:

$$\begin{cases} \mathrm{A}(\mathrm{q}^{-1})y[n] = \mathrm{B}(\mathrm{q}^{-1})u[n] + \mathrm{C}(\mathrm{q}^{-1})e[n] \\ \mathrm{E}\{e[n]e[m]\} = \lambda^2\delta_0[n-m],\ \forall\, n,m \in \mathbb{N}, \end{cases} \tag{1}$$

where u and y are the i/o signals, e is the stochastic (exogenous) white noise of variance $\lambda^2 > 0$, E is the statistical mean (the mathematical expectation) operator, q^{-1} is the one-step delay operator, δ_0 is the origin-centered unit-impulse (the Kronecker symbol), while A, B, and C are the following polynomials:

$$\begin{cases} \mathrm{A}(\mathrm{q}^{-1}) = 1 + a_1\mathrm{q}^{-1} + a_2\mathrm{q}^{-2} + \cdots + a_{na}\mathrm{q}^{-na} \\ \mathrm{B}(\mathrm{q}^{-1}) = b_1\mathrm{q}^{-1} + b_2\mathrm{q}^{-2} + \cdots + b_{nb}\mathrm{q}^{-nb} \\ \mathrm{C}(\mathrm{q}^{-1}) = 1 + c_1\mathrm{q}^{-1} + c_2\mathrm{q}^{-2} + \cdots + c_{nc}\mathrm{q}^{-nc} \end{cases}. \tag{2}$$

The polynomial coefficients in Definition (2), their degrees (na, nb, and nc), as well as the variance λ^2 are unknown and can be identified from i/o data.

Interestingly, the first equation in (1) can equivalently be written in filtering form:

$$y[n] = \frac{\mathrm{B}(\mathrm{q}^{-1})}{\mathrm{A}(\mathrm{q}^{-1})}u[n] + \frac{\mathrm{C}(\mathrm{q}^{-1})}{\mathrm{A}(\mathrm{q}^{-1})}e[n] = y_u[n] + v[n],\ \forall\, n \in \mathbb{N}, \tag{3}$$

which outlines that the output y is obtained by adding the colored noise v (resulted after filtering the white noise) to the useful (util) output y_u of the system filter. Both filters are rational and have the same poles, given by the roots of polynomial A. Equation (3) reveals that the ARMAX model is composed of two simpler models: ARX for the useful part and ARMA for the noisy part.

The NARMAX model can be defined starting from the ARMAX model, as follows (for now, in the case of SISO systems):

$$\begin{cases} \mathrm{A}(\mathrm{q}^{-1})y[n] + \sum\limits_{p=0}^{P-1} \mathrm{A}_p(\mathrm{q}^{-1})y[n]y[n-p] = \\ \qquad = \mathrm{B}(\mathrm{q}^{-1})u[n] + \sum\limits_{m=0}^{M-1} \mathrm{B}_m(\mathrm{q}^{-1})u[n]u[n-m] + \sum\limits_{k=0}^{K-1} \mathrm{D}_k(\mathrm{q}^{-1})y[n]u[n-k] + \mathrm{C}(\mathrm{q}^{-1})e[n] \\ \mathrm{E}\{e[n]e[m]\} = \lambda^2\delta[n-m]\,, \quad \forall\, n, m \in \mathbb{N}\,. \end{cases} \tag{4}$$

One can easily notice that the ARMAX model is included in Definition (4). The supplementary terms describe the nonlinearity of product type. Thus, products between i/o signals are inserted and processed by polynomials:

$$\begin{cases} \mathrm{A}_p(\mathrm{q}^{-1}) = a_{p,1}\mathrm{q}^{-1} + a_{p,2}\mathrm{q}^{-2} + \cdots + a_{p,na_p}\mathrm{q}^{-na_p}\,, \quad \forall\, p \in \overline{0, P-1} \\ \mathrm{B}_m(\mathrm{q}^{-1}) = b_{m,1}\mathrm{q}^{-1} + b_{m,2}\mathrm{q}^{-2} + \cdots + b_{m,nb_m}\mathrm{q}^{-nb_m}\,, \quad \forall\, m \in \overline{0, M-1} \\ \mathrm{D}_k(\mathrm{q}^{-1}) = d_{k,1}\mathrm{q}^{-1} + d_{k,2}\mathrm{q}^{-2} + \cdots + d_{k,nd_k}\mathrm{q}^{-nd_k}\,, \quad \forall\, k \in \overline{0, K-1} \end{cases}. \tag{5}$$

Depending on the delays employed in signal products, several polynomials are needed. Thus, the model has two types of structural indices: *major*, which determine the number of polynomials for the nonlinear part (namely, P, M, and K) and *minor*, which determine the degrees of all the polynomials (namely, na, nb, nc, na_0, …, na_{P-1}, nb_0, …, nb_{M-1}, nd_0, …, nd_{K-1}). Note that the polynomials in Definition (5) have no free term, just like the B polynomial in Definition (2). This restriction is set by considering that all the useful filters of the model need at least one step delay to process their inputs. On the contrary, the noisy part corrupts the useful part without any delay, which enforces a free term in the C polynomial. The free term of polynomial A is necessary to separate the current output ($y[n]$) from all the other samples of signals (but $e[n]$), which are delayed by at least one step. In this way, Equations (2), (4) and (5) can be implemented for simulation without algebraic loops. Obviously, all the structural indices, as well as all the polynomial coefficients and the variance λ^2, are unknown.

The NARMAX model of Equations (2), (4) and (5) is not the most general one. One can increase the generality and accentuate the nonlinear behavior by considering products of more than two signals and/or involving the white noise in the products with itself and/or with the other i/o signals (see, for example, [8]). Such models become closer and closer to Volterra models [7], which are among the most general nonlinear identification models. However, in real life, one can hardly find systems that need general models like these ones.

Although the signals of NARMAX model are nonlinear, the output continues to linearly depend on polynomial coefficients, i.e., on parameters. This allows for expressing the model by grouping all the parameters in a (large) vector as follows:

$$\boldsymbol{\theta} = \begin{bmatrix} \mathbf{a}^T & \mathbf{a}_0^T & \cdots & \mathbf{a}_{P-1}^T & \mathbf{b}^T & \mathbf{b}_0^T & \cdots & \mathbf{b}_{M-1}^T & \mathbf{d}_0^T & \cdots & \mathbf{d}_{K-1}^T & \mathbf{c}^T \end{bmatrix}^T, \tag{6}$$

with natural notations. More specifically, $\mathbf{a}$ is the column vector of polynomial A coefficients, $\mathbf{b}$ encompasses all the coefficients of polynomial B, and so on. Consequently, the first equation of Definition (4) becomes:

$$y[n] = \boldsymbol{\varphi}^T[n]\boldsymbol{\theta} + e[n],\ \forall\, n \in \mathbb{N}, \tag{7}$$

where

$$\begin{aligned} \boldsymbol{\varphi}[n] = \Big[\ & \boldsymbol{\varphi}_y^T[n] \quad \boldsymbol{\varphi}_{y,0}^T[n] \quad \cdots \quad \boldsymbol{\varphi}_{y,P-1}^T[n] \quad | \\ & \boldsymbol{\varphi}_u^T[n] \quad \boldsymbol{\varphi}_{u,0}^T[n] \quad \cdots \quad \boldsymbol{\varphi}_{u,M-1}^T[n] \quad | \\ & \boldsymbol{\varphi}_{yu,0}^T[n] \quad \cdots \quad \boldsymbol{\varphi}_{yu,K-1}^T[n] \quad | \quad \boldsymbol{\varphi}_e^T[n]\ \Big]^T,\ \ \forall\, n \in \mathbb{N}\,, \end{aligned} \tag{8}$$

with

$$\begin{cases} \boldsymbol{\varphi}_y^T[n] = \begin{bmatrix} -y[n-1] & \cdots & -y[n-na] \end{bmatrix} \\ \boldsymbol{\varphi}_{y,p}^T[n] = \begin{bmatrix} -y[n-1]y[n-p-1] & \cdots & -y[n-na_p]y[n-p-na_p] \end{bmatrix}, \quad \forall\, p \in \overline{0, P-1} \\ \boldsymbol{\varphi}_u^T[n] = \begin{bmatrix} u[n-1] & \cdots & u[n-nb] \end{bmatrix} \end{cases}$$

$$\begin{cases} \boldsymbol{\varphi}_{u,m}^T[n] = \begin{bmatrix} u[n-1]u[n-m-1] & \cdots & u[n-nb_m]u[n-m-nb_m] \end{bmatrix}, \quad \forall\, m \in \overline{0, M-1} \\ \boldsymbol{\varphi}_{yu,k}^T[n] = \begin{bmatrix} y[n-1]u[n-k-1] & \cdots & y[n-nd_k]u[n-k-nd_k] \end{bmatrix}, \quad \forall\, k \in \overline{0, K-1} \\ \boldsymbol{\varphi}_e^T[n] = \begin{bmatrix} e[n-1] & \cdots & e[n-nc] \end{bmatrix}, \end{cases} \tag{9}$$

for any normalized instant $n \in \mathbb{N}$. Conventionally, any signal is null for non-positive normalized instants ($n < 1$).

The regression form (7) yields employing an identification method from the Least Squares class to obtain the optimal models.

The NARMAX model can also be expressed by means of filters. Thus:

$$y[n] = y_u[n] + v[n], \ \forall\, n \in \mathbb{N}, \tag{10}$$

where the useful signal is obtained by a bank of filters having the same poles as follows:

$$\begin{aligned} y_u[n] = -\sum_{p=0}^{P-1} \frac{\mathrm{A}_p(\mathrm{q}^{-1})}{\mathrm{A}(\mathrm{q}^{-1})} y[n]y[n-p] + \frac{\mathrm{B}(\mathrm{q}^{-1})}{\mathrm{A}(\mathrm{q}^{-1})} u[n] + \sum_{m=0}^{M-1} \frac{\mathrm{B}_m(\mathrm{q}^{-1})}{\mathrm{A}(\mathrm{q}^{-1})} u[n]u[n-m] + \\ + \sum_{k=0}^{K-1} \frac{\mathrm{D}_k(\mathrm{q}^{-1})}{\mathrm{A}(\mathrm{q}^{-1})} y[n]u[n-k], \quad \forall\, n \in \mathbb{N}, \end{aligned} \tag{11}$$

while the colored noise is obtained by means of an ARMA model as follows:

$$v[n] = \frac{\mathrm{C}(\mathrm{q}^{-1})}{\mathrm{A}(\mathrm{q}^{-1})} e[n], \ \forall\, n \in \mathbb{N}. \tag{12}$$

Expressions (10)–(12) are extremely useful when simulating the model after identification by means of filtering blocks. The simulation scheme is illustrated in Figure 1. Two blocks are emphasized in the scheme: NARMAX-IN to the left and NARMAX-OUT to the right.

Another method to simulate the NARMAX model is to split the regression form (7) into two terms: one exclusively dealing with the white noise and the other one working with i/o signals only. More specifically, Equation (7) becomes:

$$y[n] = \boldsymbol{\varphi}_{u,y}^T[n]\ \boldsymbol{\theta}_{u,y} + \boldsymbol{\varphi}_e^T[n]\ \boldsymbol{\theta}_e + e[n], \ \forall\, n \in \mathbb{N}, \tag{13}$$

where

$$\boldsymbol{\theta}_{u,y} = \begin{bmatrix} \mathbf{a}^T & \mathbf{a}_0^T & \cdots & \mathbf{a}_{P-1}^T & \mathbf{b}^T & \mathbf{b}_0^T & \cdots & \mathbf{b}_{M-1}^T & \mathbf{d}_0^T & \cdots & \mathbf{d}_{K-1}^T \end{bmatrix}^T, \ \boldsymbol{\theta}_e = \mathbf{c} \tag{14}$$

and $\boldsymbol{\varphi}_{u,y}$ is the remaining vector of $\boldsymbol{\varphi}$ after removing the $\boldsymbol{\varphi}_e$ component (see Equations (8) and (9) again). This time, the NARMAX model can be obtained by means of two simpler models: NARX and MA.

So, practically, the colored noise is obtained by simply filtering the white noise with the help of C polynomial. In this case, Equation (13) works with the following couple of useful and noisy signals:

$$\begin{cases} y_u[n] = \boldsymbol{\varphi}_{u,y}^T[n]\ \boldsymbol{\theta}_{u,y} \\ v[n] = \boldsymbol{\varphi}_e^T[n]\ \boldsymbol{\theta}_e = \left[\mathrm{C}(\mathrm{q}^{-1}) - 1\right] e[n] \end{cases}, \ \forall\, n \in \mathbb{N}. \tag{15}$$

Alternatively, the simulation of NARMAX model can be performed by using Equations (13) and (15). The blocks NARMAX-IN and NARMAX-OUT exist as well. Different from the filtering scheme in Figure 1, here, NARMAX-IN returns the useful signal y_u of Equation (15), while NARMAX-OUT takes y_u and e as inputs and returns the overall output y.

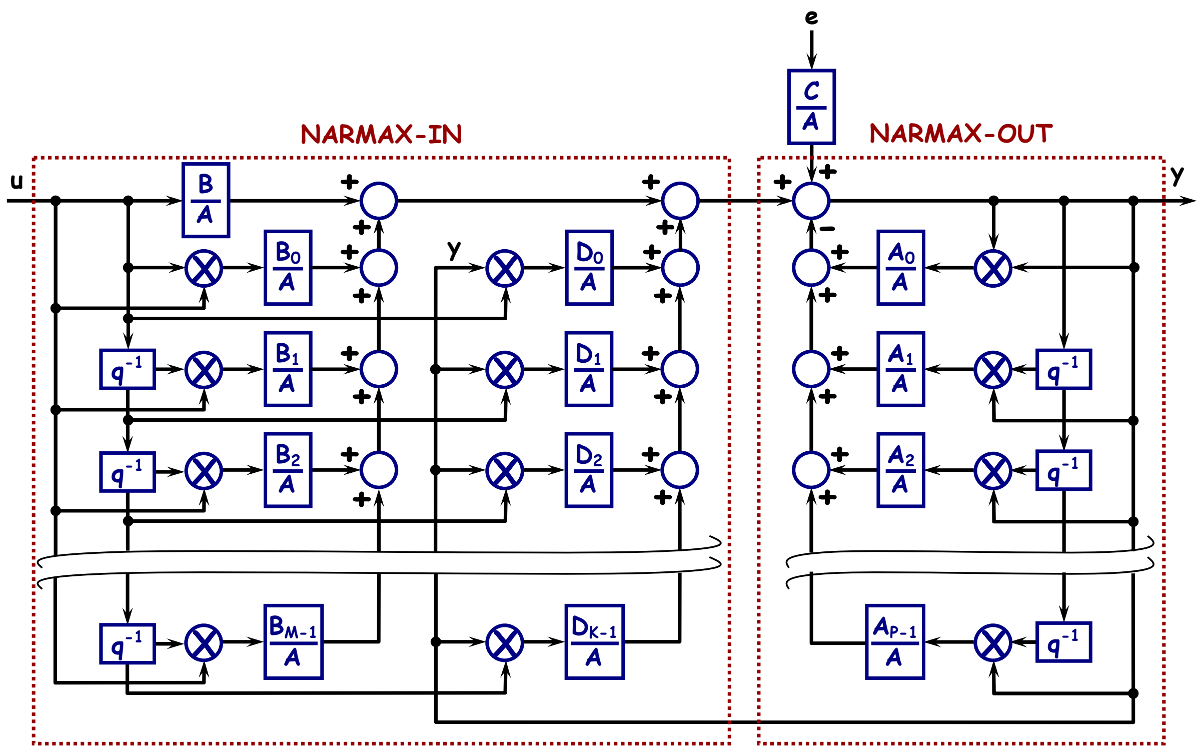

Figure 1. Simulation scheme of SISO-NARMAX model.

From SISO systems, the NARMAX model can be extended to match multivariable systems. To keep the model complexity at an acceptable level, assume that such a system has two inputs and two outputs. Moreover, it is very likely that the system exhibits uncorrelated outputs. Consequently, the system includes two subsystems of *Multiple-Inputs-Single-Output* (MISO) type. In this case, the NARMAX model is described by the following equations:

$$\begin{cases} \mathrm{A}_1(\mathrm{q}^{-1})y_1[n] + \sum\limits_{p=0}^{P_1-1} \mathrm{A}_{1,p}(\mathrm{q}^{-1})y_1[n]y_1[n-p] = \\ \quad = \mathrm{B}_{1,1}(\mathrm{q}^{-1})u_1[n] + \sum\limits_{m=0}^{M_{1,1}-1} \mathrm{B}_{1,1,m}(\mathrm{q}^{-1})u_1[n]u_1[n-m] + \sum\limits_{k=0}^{K_{1,1}-1} \mathrm{D}_{1,1,k}(\mathrm{q}^{-1})y_1[n]u_1[n-k] + \\ \quad + \mathrm{B}_{1,2}(\mathrm{q}^{-1})u_2[n] + \sum\limits_{m=0}^{M_{1,2}-1} \mathrm{B}_{1,2,m}(\mathrm{q}^{-1})u_2[n]u_2[n-m] + \\ \quad + \sum\limits_{k=0}^{K_{1,2}-1} \mathrm{D}_{1,2,k}(\mathrm{q}^{-1})y_1[n]u_2[n-k] + \mathrm{C}_1(\mathrm{q}^{-1})e_1[n] \\ \mathrm{E}\{e_1[n]e_1[m]\} = \lambda_1^2\delta[n-m]\,, \quad \forall\, n,m \in \mathbb{N}\,; \end{cases} \tag{16}$$

$$\begin{cases} \mathrm{A}_2(\mathrm{q}^{-1})y_2[n] + \sum\limits_{p=0}^{P_2-1} \mathrm{A}_{2,p}(\mathrm{q}^{-1})y_2[n]y_2[n-p] = \\ \quad = \mathrm{B}_{2,1}(\mathrm{q}^{-1})u_1[n] + \sum\limits_{m=0}^{M_{2,1}-1} \mathrm{B}_{2,1,m}(\mathrm{q}^{-1})u_1[n]u_1[n-m] + \sum\limits_{k=0}^{K_{2,1}-1} \mathrm{D}_{2,1,k}(\mathrm{q}^{-1})y_2[n]u_1[n-k] + \\ \quad + \mathrm{B}_{2,2}(\mathrm{q}^{-1})u_2[n] + \sum\limits_{m=0}^{M_{2,2}-1} \mathrm{B}_{2,2,m}(\mathrm{q}^{-1})u_2[n]u_2[n-m] + \\ \quad + \sum\limits_{k=0}^{K_{2,2}-1} \mathrm{D}_{2,2,k}(\mathrm{q}^{-1})y_2[n]u_2[n-k] + \mathrm{C}_2(\mathrm{q}^{-1})e_2[n] \\ \mathrm{E}\{e_2[n]e_2[m]\} = \lambda_2^2\delta[n-m]\,, \quad \forall\, n,m \in \mathbb{N}\,; \end{cases} \tag{17}$$

$$\mathrm{E}\{e_1[n]e_2[m]\} = 0, \ [\mathrm{M1}] \ \forall\, n, m \in \mathbb{N}. \tag{18}$$

In Equations (16) and (17), polynomials are denoted like in Equations (2) and (5), with indexing accordingly adapted to the multivariable case. For example, the major indices P, M, and K are replaced by the sets $\{P_1, P_2\}$, $\{M_{1,1}, M_{1,2}, M_{2,1}, M_{2,2}\}$, and $\{K_{1,1}, K_{1,2}, K_{2,1}, K_{2,2}\}$, respectively. One can notice that the number of structural indices, as well as the number of parameters, can rapidly increase to very large values. Equation (18) expresses the natural hypothesis that the two white noises are uncorrelated. If they were correlated, their cross-correlation matrix would add to the list of unknown parameters, which can complicate the identification.

The model (16) and (17) can be expressed either in regression form, like (13), or in filtering form, like (10). The filtering simulation scheme associated to NARMAX model in this case is depicted in Figure 2.

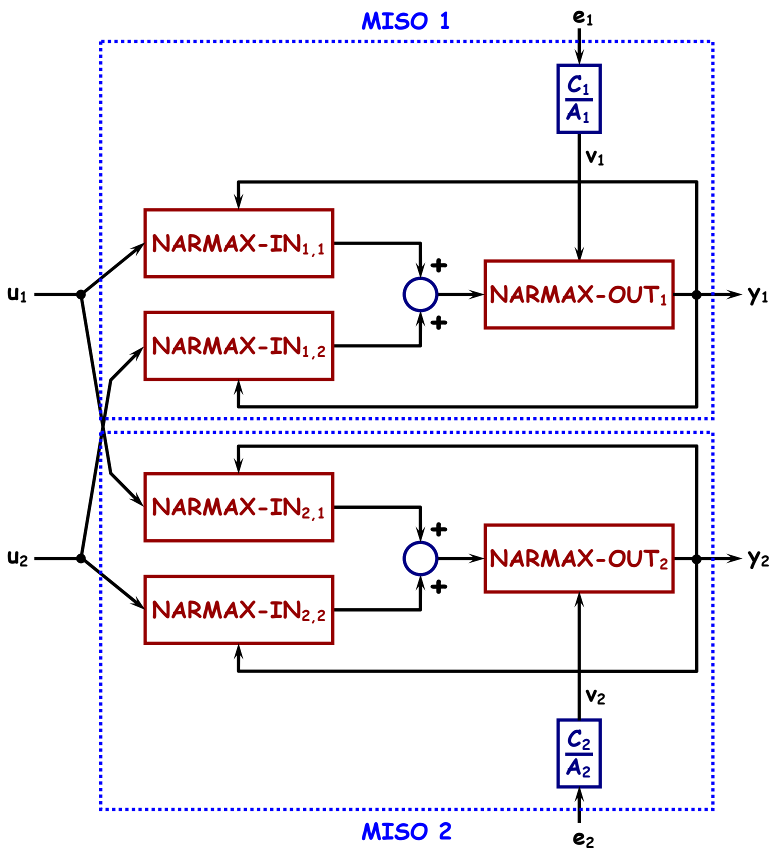

Figure 2. Simulation scheme of a 2-MISO-NARMAX model, using filtering blocks, to be included in a closed loop configuration.

The blocks of NARMAX-IN and NARMAX-OUT type are similar to those in Figure 1. A similar figure can be drawn for models expressed in regression form. The only difference from Figure 2 is that the ARMA filters vanish, and the NARMAX-OUT blocks only receive white noises as secondary inputs.

2.2. Identification of NARMAX Parameters through Multi-Step Orthogonal Least Squares

To easier understand how the parameters of a NARMAX model can be identified, come back, for now, to the SISO case. The model equation of interest is (7), with notations (6), (8) and (9). Assume that an i/o dataset is available, namely: $\mathbf{D}_N = \{u[n], y[n]\}_{n\in\overline{1,N}}$, where the acquisition horizon length, $N \in \mathbb{N}^*$, is sufficiently large (few tens, at least). Also, consider that all the structural indices are set to some known values. Then, the optimal parameters can be determined by solving the following quadratic optimization problem:

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^{n\theta}} \mathbf{V}(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}\in\mathbb{R}^{n\theta}} \sum_{n=1}^{N} \left(y[n] - \boldsymbol{\varphi}^T[n]\boldsymbol{\theta} \right)^2, \tag{19}$$

where $n\theta$ is the length of vector $\boldsymbol{\theta}$, and $\mathbf{V}$ is the quadratic optimization criterion. It can easily be proven that the least squares solution of problem (19) is

$$\hat{\boldsymbol{\theta}}_N = \underset{\boldsymbol{\theta}\in\mathbb{R}^{n\theta}}{\operatorname{argmin}} \sum_{n=1}^{N} \left(y[n] - \boldsymbol{\varphi}^T[n]\boldsymbol{\theta} \right)^2 = \underbrace{\left[\sum_{n=1}^{N} \boldsymbol{\varphi}[n]\boldsymbol{\varphi}^T[n] \right]^{-1}}_{\mathbf{R}_N^{-1}} \underbrace{\sum_{n=1}^{N} \boldsymbol{\varphi}[n]y[n]}_{\mathbf{r}_N}, \tag{20}$$

provided that the matrix $\mathbf{R}_N$ is invertible. Moreover, the variance of the white noise can also be estimated by means of the quadratic criterion as follows:

$$\hat{\lambda}_N^2 = \frac{1}{N - n\theta} \mathbf{V}(\hat{\boldsymbol{\theta}}_N) = \frac{1}{N - n\theta} \sum_{n=1}^{N} \left(y[n] - \boldsymbol{\varphi}^T[n]\hat{\boldsymbol{\theta}}_N \right)^2, \tag{21}$$

which shows that, in fact, the optimal identification model minimizes the variance of the estimated white noise. In Definition (21), the denominator could be replaced by N. Nevertheless, it has been proven that, under some hypotheses, Definition (21) leads to better statistical properties of the variance estimation than by using N as the denominator (for example, N introduces a bias, whereas $N - n\theta$ provides unbiased estimations) [1].

Beside the critical condition of $\mathbf{R}_N$ invertibility, two more requirements need to be fulfilled. Firstly, the matrix $\mathbf{R}_N$ should be numerically balanced to allow inversion without critical errors. Usually, this matrix is unbalanced. Therefore, a balancing technique is necessary. Orthogonalization is one of the most popular techniques. Secondly, the vector $\boldsymbol{\varphi}$ must be numerically available at each normalized instant. In case of NARMAX model, Definitions (8) and (9) show that the component $\boldsymbol{\varphi}_e$ is unavailable, as the white noise cannot be measured separately from the output data. Consequently, the white noise needs to be estimated. One popular approach is to employ an auxiliary identification model, different from NARMAX, by using the same i/o dataset. For the NARMAX model, NARX is the most natural auxiliary model. How these two requirements are met is explained at length in the seque.

Approach, first, the matrix-balancing requirement. Assume that all the transposed vectors $\left\{\boldsymbol{\varphi}^T[n]\right\}_{n\in\overline{1,N}}$ are stacked on rows into the matrix $\boldsymbol{\Phi}_N \in \mathbb{R}^{N\times n\theta}$. Then, obviously, $\mathbf{R}_N = \boldsymbol{\Phi}_N^T\boldsymbol{\Phi}_N$. To make invertible the matrix $\mathbf{R}_N$, it suffices that the matrix $\boldsymbol{\Phi}_N$ becomes monic. Moreover, it results from Equation (20) that $\mathbf{r}_N = \boldsymbol{\Phi}_N^T\mathbf{Y}_N$, where $\mathbf{Y}_N \in \mathbb{R}^N$ is the vector of all acquired output data. Consequently,

$$\hat{\boldsymbol{\theta}}_N = \mathbf{R}_N^{-1}\mathbf{r}_N = \left(\boldsymbol{\Phi}_N^T\boldsymbol{\Phi}_N \right)^{-1} \boldsymbol{\Phi}_N^T\mathbf{Y}_N. \tag{22}$$

The problem is how to invert the product $\boldsymbol{\Phi}_N^T\boldsymbol{\Phi}_N$ through a numerically stable procedure. The solution can rely on the linear independence of the $n\theta$ columns in matrix $\boldsymbol{\Phi}_N$ (which is supposed to be monic). Normally, $n\theta < N$, as the number of model parameters usually is smaller than the acquisition horizon length. Denote by $\{\phi_{N,m}\}_{m\in\overline{1,n\theta}}$ the columns of matrix $\boldsymbol{\Phi}_N$. Geometrically, they span an $n\theta$-sized subspace of $\mathbb{R}^N$, on which the vector $\mathbf{Y}_N$ is projected. The projection coefficients on each column of $\boldsymbol{\Phi}_N$ are, in fact, the elements of the optimal parameter vector (22). Moreover, the projected vector is $\hat{\mathbf{Y}}_N = \boldsymbol{\Phi}_N\hat{\boldsymbol{\theta}}_N$. Note that although the columns $\{\phi_{N,m}\}_{m\in\overline{1,n\theta}}$ are linearly independent, they are not necessarily orthogonal to each other. What if the same subspace would be spanned by an orthogonal set of vectors built from the columns of matrix $\boldsymbol{\Phi}_N$?

The new orthogonal set can easily be obtained by using the well-known GramSchmidt method. Thus, the orthogonal set is recursively built, as follows:

$$\boldsymbol{\phi}^{\perp}_{N,m} = \boldsymbol{\phi}_{\;N,m} - \sum_{k=1}^{m-1} \frac{\left\langle \boldsymbol{\phi}_{N,m}, \boldsymbol{\phi}^{\perp}_{N,k} \right\rangle}{\left\| \boldsymbol{\phi}^{\perp}_{N,k} \right\|^2} \boldsymbol{\phi}^{\perp}_{N,k},\ \forall\, m \in \overline{2, n\theta}, \tag{23}$$

after taking $\boldsymbol{\phi}^{\perp}_{N,1} = \boldsymbol{\phi}_{N,1}$ as initialization. Denote by $\boldsymbol{\Phi}^{\perp}_{N}$ the matrix with columns $\left\{ \phi^{\perp}_{N,m} \right\}_{m \in \overline{1,n\theta}}$. Then, the output data vector $\mathbf{Y}_N$ can also be projected on the orthogonal set, and the resulting coefficients are gathered into the vector:

$$\hat{\boldsymbol{\theta}}^{\perp}_{N} = \left[\left(\boldsymbol{\Phi}^{\perp}_{N} \right)^T \boldsymbol{\Phi}^{\perp}_{N} \right]^{-1} \left(\boldsymbol{\Phi}^{\perp}_{N} \right)^T \mathbf{Y}_N = \Delta_N^{-1} \left(\boldsymbol{\Phi}^{\perp}_{N} \right)^T \mathbf{Y}_N, \tag{24}$$

where $\Delta_N = \left(\boldsymbol{\Phi}^{\perp}_{N} \right)^T \boldsymbol{\Phi}^{\perp}_{N}$ is a diagonal matrix including the quadratic norms of the orthogonal vectors. Thus, from Equation (24), it results that each parameter of $\hat{\boldsymbol{\theta}}^{\perp}_{N}$ can easier be expressed by:

$$\hat{\theta}^{\perp}_{N,m} = \frac{\left\langle \mathbf{Y}, \boldsymbol{\phi}^{\perp}_{m} \right\rangle}{\left\| \boldsymbol{\phi}^{\perp}_{m} \right\|^2},\ \forall\, m \in \overline{1, n\theta}. \tag{25}$$

Evidently, the parameter vector $\hat{\boldsymbol{\theta}}^{\perp}_{N}$ is different, in general, from the vector $\hat{\boldsymbol{\theta}}_N$ and constitutes a suboptimal solution of problem (19). However, there is a link between these two vectors. Observe that the projection of vector $\mathbf{Y}_N$ on the orthogonal set is identical to the projection of vector $\hat{\mathbf{Y}}_N$ on the same set since the difference $\mathbf{Y}_N - \hat{\mathbf{Y}}_N$ (the modeling error) is orthogonal on the spanned subspace. This property can mathematically be written as follows:

$$\left(\boldsymbol{\Phi}^{\perp}_{N} \right)^T \mathbf{Y}_N = \left(\boldsymbol{\Phi}^{\perp}_{N} \right)^T \boldsymbol{\Phi}_N \hat{\boldsymbol{\theta}}_N. \tag{26}$$

According to Equation (24), the vector $\hat{\boldsymbol{\theta}}^{\perp}_{N}$ can be obtained from Equation (26) through a simple multiplication by Δ_N^{-1}, as follows:

$$\hat{\boldsymbol{\theta}}^{\perp}_{N} = \Delta_N^{-1} \left(\boldsymbol{\Phi}^{\perp}_{N} \right)^T \boldsymbol{\Phi}_N \hat{\boldsymbol{\theta}}_N. \tag{27}$$

Focus now on the product $\left(\boldsymbol{\Phi}^{\perp}_{N} \right)^T \boldsymbol{\Phi}_N$ in Equation (27). The elements of this matrix are computed by projecting orthogonal vectors $\left\{ \phi^{\perp}_{N,m} \right\}_{m \in \overline{1,n\theta}}$ on the initial vectors $\{ \phi_{N,m} \}_{m \in \overline{1,n\theta}}$. Before computing the scalar products, notice that the columns of matrix $\boldsymbol{\Phi}_N$ can be expanded into the orthogonal set by means of Equation (23), like below:

$$\boldsymbol{\phi}_{N,1} = \boldsymbol{\phi}^{\perp}_{N,1},\ \boldsymbol{\phi}_{N,m} = \boldsymbol{\phi}^{\perp}_{N,m} + \sum_{k=1}^{m-1} \frac{\left\langle \boldsymbol{\phi}_{N,m}, \boldsymbol{\phi}^{\perp}_{N,k} \right\rangle}{\left\| \boldsymbol{\phi}^{\perp}_{N,k} \right\|^2} \boldsymbol{\phi}^{\perp}_{N,k},\ \forall\, m \in \overline{2, n\theta}, \tag{28}$$

Hence, for each $m, p \in \overline{1, n\theta}$:

$$\left\langle \phi^{\perp}_{N,m}, \phi_{N,p} \right\rangle = \begin{cases} 0 & ,\ m > p \\ \left\| \phi^{\perp}_{N,m} \right\|^2 & ,\ m = p \\ \sum\limits_{k=m}^{p-1} \dfrac{\left\langle \boldsymbol{\phi}_{N,p}, \boldsymbol{\phi}^{\perp}_{N,k} \right\rangle}{\left\| \boldsymbol{\phi}^{\perp}_{N,k} \right\|^2} \left\langle \phi^{\perp}_{N,m}, \boldsymbol{\phi}^{\perp}_{N,k} \right\rangle & ,\ m < p \end{cases}. \tag{29}$$

From Equation (29), it results that the matrix $\left(\boldsymbol{\Phi}_N^{\perp}\right)^T \boldsymbol{\Phi}_N$ is upper triangular. Consequently, the optimal parameters vector $\hat{\boldsymbol{\theta}}_N$ can be obtained by solving the upper triangular system (27) through the back substitution procedure, as follows:

$$\begin{gathered}\hat{\theta}_{N,n\theta} = \frac{\left\langle \mathbf{Y}_N, \boldsymbol{\phi}^{\perp}_{N,n\theta} \right\rangle}{\left\| \boldsymbol{\phi}^{\perp}_{N,n\theta} \right\|^2},\ \hat{\theta}_{N,n\theta-1} = \frac{\left\langle \mathbf{Y}_N, \boldsymbol{\phi}^{\perp}_{N,n\theta-1} \right\rangle - \left\langle \boldsymbol{\phi}_{N,n\theta}, \boldsymbol{\phi}^{\perp}_{N,n\theta-1} \right\rangle \hat{\theta}_{N,n\theta}}{\left\| \boldsymbol{\phi}^{\perp}_{N,n\theta-1} \right\|^2}, \ldots, \\ \hat{\theta}_{N,m} = \frac{\left\langle \mathbf{Y}_N, \boldsymbol{\phi}^{\perp}_{N,m} \right\rangle - \sum\limits_{k=m+1}^{n\theta} \left\langle \boldsymbol{\phi}_{N,k}, \boldsymbol{\phi}^{\perp}_{N,m} \right\rangle \hat{\theta}_{N,k}}{\left\| \boldsymbol{\phi}^{\perp}_{N,m} \right\|^2}, \ldots, \ \hat{\theta}_{N,1} = \frac{\left\langle \mathbf{Y}_N, \boldsymbol{\phi}_{N,1} \right\rangle - \sum\limits_{k=2}^{n\theta} \left\langle \boldsymbol{\phi}_{N,k}, \boldsymbol{\phi}_{N,1} \right\rangle \hat{\theta}_{N,k}}{\left\| \boldsymbol{\phi}_{N,1} \right\|^2}.\end{gathered} \tag{30}$$

Note that Equation (30) can be implemented without prior computation of the suboptimal solution (27).

The drawback of this technique results from the size of the vectors, which is equal to N. If N is too large for the employed computing system, then another balancing technique can be adopted. For example, one can use the QR decomposition of matrix $\boldsymbol{\Phi}_N$. This technique performs orthogonalization by means of rotation matrices and, unfortunately, may be time consuming. Regardless the orthogonalization technique, the Least Squares methods implemented through such balancing procedures are referred to as *Orthogonal Least Squares* (OLS).

To fulfil the second requirement, two major approaches are proposed. The first one relies on GaussNewton optimization method and is quite accurate but time consuming. The second one is based on the ELS method and leads to faster numerical procedures, although lessaccurate results are provided. The MS-EOLS method described next shows how the accuracy of the ELS method can be improved by means of filtering.

Basically, the ELS consists of two steps: first, identify an auxiliary model devoted to the estimation of the white noise; second, apply the OLS to identify the main model, after replacing the true (but unknown) values of the white noise by the estimated values. Usually, the auxiliary model is related to the main model and can be defined after replacing the colored noise by the white noise into the general equation. For example, ARX are natural auxiliary models of ARMAX models, for which the OLS method can be applied directly. The first step of the ELS method is the critical one because the noise estimation errors are amplified in the second step. Some filters can be employed several times to increase the accuracy of the estimation, as explained next. This repeated filtering operation involves applying the ELS more than once, which led to the name of *Multi-Step ELS*. If the OLS method is employed in all the steps, then the overall method is referred to as *Multi-Step EOLS* (MS-EOLS).

As already mentioned, in case of the SISO-NARMAX model, the regression Equation (7) can equivalently be written by isolating the noisy part, like in Equation (13). This suggests selecting the NARX model below as the auxiliary estimator of the white noise:

$$y[n] = \boldsymbol{\psi}^T_{u,y}[n]\ \boldsymbol{\theta}^{\mathrm{aux}}_{u,y} + e[n],\ \forall\, n \in \mathbb{N}, \tag{31}$$

where $\boldsymbol{\theta}^{\mathrm{aux}}_{u,y}$ has the same structure as $\boldsymbol{\theta}_{u,y}$ from Definition (14), and $\boldsymbol{\psi}_{u,y}$ is similar to $\boldsymbol{\varphi}_{u,y}$. Nevertheless, since the NARX model is simpler than the NARMAX model, vectors $\boldsymbol{\theta}^{\mathrm{aux}}_{u,y}$ and $\boldsymbol{\psi}_{u,y}$ should be longer than $n\theta$. The NARX model adds to the set of unknown parameters new major indices of P and M type, as well as corresponding minor indices and polynomial coefficients.

After the identification of NARX model (by directly applying the OLS method since no white noise is included into the vector $\boldsymbol{\psi}_{u,y}$), Equation (31) serves as estimator of the white noise:

$$\hat{e}_N[n] = y[n] - \boldsymbol{\psi}^T_{u,y}[n]\ \hat{\boldsymbol{\theta}}^{\mathrm{aux}}_{N,u,y},\ \forall\, n \in \mathbb{N}, \tag{32}$$

where $\hat{\boldsymbol{\theta}}^{\text{aux}}_{N,u,y}$ is the optimal parameter vector, obtained for some structural indices (already set). The estimated white noise can replace the true values in $\boldsymbol{\varphi}_e$, which becomes $\hat{\boldsymbol{\varphi}}_{N,e}$. Now, the OLS can be applied to identify the NARMAX optimal vector of the parameters by means of Equation (20). Thus, the vector $\hat{\boldsymbol{\theta}}_N$ is obtained. How accurate is such an estimation? To answer this question, one can compute the quadratic criterion from Equation (19) for the optimal vector $\hat{\boldsymbol{\theta}}_N$, with the estimated noise (32), as follows:

$$\mathbf{V}\big(\hat{\boldsymbol{\theta}}_N\big) = \sum_{n=1}^{N}\Big(y[n] - \boldsymbol{\varphi}^T_{u,y}[n]\,\hat{\boldsymbol{\theta}}_{N,u,y} - \hat{\boldsymbol{\varphi}}^T_{N,e}[n]\hat{\mathbf{c}}_N\Big)^2. \tag{33}$$

In subsidiary, the quadratic criterion uses the following output simulated signal:

$$\hat{y}[n] = \boldsymbol{\varphi}^T_{u,y}[n]\,\hat{\boldsymbol{\theta}}_{N,u,y} - \hat{\boldsymbol{\varphi}}^T_{N,e}[n]\hat{\mathbf{c}}_N,\ \forall\, n \in \mathbb{N} \tag{34}$$

and performs a comparison between the acquired output data y and simulated output data $\hat{y}$. The difference between them is another estimation of the white noise, alternatively seen as model error.

If $\mathbf{V}\big(\hat{\boldsymbol{\theta}}_N\big)$ is less than some threshold, $\eta > 0$, then the NARMAX model is accurate enough. Otherwise, the accuracy of the white noise estimation has to be increased. One way to reach for this goal is to change the auxiliary model, or, at least, to change the structural indices of NARX model. Another way is to employ the estimated polynomial $\hat{\mathrm{C}}$ of NARMAX model for designing an all-poles filter that can refine the i/o data for the NARX model. More specifically, to identify the NARX model, the dataset $\mathbf{D}_N = \{u[n], y[n]\}_{n\in\overline{1,N}}$ is replaced by the following filtered dataset:

$$\mathbf{D}^{\hat{\mathrm{C}}}_N = \left\{\frac{1}{\hat{\mathrm{C}}(\mathrm{q}^{-1})}u[n], \frac{1}{\hat{\mathrm{C}}(\mathrm{q}^{-1})}y[n]\right\}_{n\in\overline{1,N}}. \tag{35}$$

Moreover, all the signal products of vector $\boldsymbol{\psi}_{u,y}$ have to be filtered as they are. It is important not to compute such products from filtered i/o data. After the new NARX model is identified, the NARMAX model is upgraded. Note that to identify the NARMAX model, the original dataset $\mathbf{D}_N$ is still employed (no filtering is applied here). With the newly identified NARX and NARMAX models, one can evaluate the criterion (33) again and compare the result with the threshold. Normally, after few iterations, the accuracy requirement is verified. If the threshold is set too low, then a maximum number of filtering steps can be set (e.g., 10 steps), and the model with the minimum value of criterion (33) can be selected.

The explanation for this filtering technique is straightforward. Come back to NARMAX Equation (4). The first equation can equivalently be expressed like below:

$$\begin{aligned} \mathrm{A}(\mathrm{q}^{-1})\left(\frac{1}{\mathrm{C}(\mathrm{q}^{-1})}y\right)[n] = -\sum_{p=0}^{P-1}\mathrm{A}_p(\mathrm{q}^{-1})\left(\frac{1}{\mathrm{C}(\mathrm{q}^{-1})}\{y\cdot(\mathrm{q}^{-p}y)\}\right)[n]+ \\ +\mathrm{B}(\mathrm{q}^{-1})\left(\frac{1}{\mathrm{C}(\mathrm{q}^{-1})}u\right)[n] + \sum_{m=0}^{M-1}\mathrm{B}_m(\mathrm{q}^{-1})\left(\frac{1}{\mathrm{C}(\mathrm{q}^{-1})}\{u\cdot(\mathrm{q}^{-m}u)\}\right)[n]+ \\ +\sum_{k=0}^{K-1}\mathrm{D}_k(\mathrm{q}^{-1})\left(\frac{1}{\mathrm{C}(\mathrm{q}^{-1})}\Big\{y\cdot\Big(\mathrm{q}^{-k}u\Big)\Big\}\right)[n] + e[n]\,,\quad \forall\, n\in\mathbb{N}\,. \end{aligned} \tag{36}$$

Equation (36) describes a NARX model working with filtered signals, such as

$$\frac{1}{\mathrm{C}(\mathrm{q}^{-1})}y,\ \frac{1}{\mathrm{C}(\mathrm{q}^{-1})}\{y\cdot(\mathrm{q}^{-p}y)\},\ \frac{1}{\mathrm{C}(\mathrm{q}^{-1})}u,\ \frac{1}{\mathrm{C}(\mathrm{q}^{-1})}\{u\cdot(\mathrm{q}^{-m}u)\},\ \frac{1}{\mathrm{C}(\mathrm{q}^{-1})}\Big\{y\cdot\Big(\mathrm{q}^{-k}u\Big)\Big\}. \tag{37}$$

Note that the filter is applied not only to i/o signals but also to products of signals, like $y\cdot(\mathrm{q}^{-p}y)$, $u\cdot(\mathrm{q}^{-m}u)$, and $y\cdot\Big(\mathrm{q}^{-k}u\Big)$, as if they were separate signals. Notice that no products between filtered i/o signals exist into the Equation (36). For example, if $y_p \equiv y\cdot(\mathrm{q}^{-p}y)$, then the values of this signal are: $y_p[1] = y[1]\cdot y[1-p]$, $y_p[2] = y[2]\cdot y[2-p]$,

$\ldots, y_p[p] = y[p] \cdot y[0], y_p[p+1] = y[p+1] \cdot y[1], \ldots, y_p[N] = y[N] \cdot y[N-p]$. (Recall that by convention, $y[n] = 0, \forall\, n < 1$.) Hence, the signal $\frac{1}{\mathrm{C}(\mathrm{q}^{-1})}\{y \cdot (\mathrm{q}^{-p} y)\}$ from Equation (37) can be evaluated by simply filtering the signal y_p, i.e., by $\frac{1}{\mathrm{C}(\mathrm{q}^{-1})} y_p$.

In case of 2-MISO-NARMAX model (16)–(17), the balancing technique, as well as the MS-EOLS method, can be applied for each output channel.

Using the MS-EOLS in the identification of the 2-MISO-NARMAX model parameters involves a large number of structural indices to be set. As Equations (16) and (17) reveal, there are 10 major indices for NARMAX models and another 10 major indices for auxiliary NARX models (20 major indices in total). The number of minor indices is even larger (the sum of all the major indices, plus 10 more, coming from the included ARMAX and ARX models on both channels). It is suitable to find optimal structural indices, such that the final identification model performs well on both channels simultaneously. Moreover, the model needs to be validated. An exhaustive search of optimal indices into such a large space is computationally inefficient, even in the case of low upper limits set for all the indices. Therefore, a metaheuristic can help solving this granular optimization problem with reasonable costs in terms of resources and time.

Nowadays, the world of metaheuristics is quite large and in expansion [48]. Within this world, there are some metaheuristics that can be employed to solve this problem. Since comparing results from various metaheuristics is not among the goals of this article, the *Cuckoo Search Algorithm* (CSA) was selected, mainly because several implementations are available, including the one from MATLAB$^{\text{TM}}$ programming environment. Compared with other already implemented metaheuristics (such as the Genetic Algorithms), the CSA is easy to configure and run.

2.3. Defining an Optimization Criterion to Find the Best Structural Indices

Before shortly describe the CSA, a new optimization criterion is necessary to be defined. The criterion employed in Equations (19) and (33) could be extended to work on two output channels by summing the quadratic errors. Unfortunately, the new criterion has the following caveat: although small overall values are obtained and one channel exhibits small quadratic errors, the other channel has unacceptably large quadratic errors. Therefore, the criterion to select the optimal structural indices needs to be defined carefully.

The milestone of such a criterion is the *Signal-to-Noise Ratio* (SNR), which can be defined as follows. Consider the signal y, which includes a zero-mean corrupting stochastic noise of variance λ^2, not necessarily white. Then, the SNR is evaluated by computing the ratio between the standard deviations of the signal and the noise. More specifically, if $\overline{y}$ is the mean of signal y, then the standard deviation σ_y is the square root of the variance computed for the stationary (null mean) signal $\widetilde{y} = y - \overline{y}$. Thus:

$$\mathrm{SNR} = \frac{\sigma_y}{\lambda} = \frac{\sqrt{\mathrm{E}\{\widetilde{y}^2\}}}{\lambda} = \frac{\sqrt{\mathrm{E}\left\{(y-\overline{y})^2\right\}}}{\lambda}. \tag{38}$$

If the signal has N samples and the variance λ^2 was estimated by means of some identification model, then the Ergodic Hypothesis can be employed to estimate the SNR as follows:

$$\widehat{\mathrm{SNR}} = \frac{\sqrt{\frac{1}{N}\sum\limits_{n=1}^{N} \widetilde{y}^2[n]}}{\hat{\lambda}_N}, \tag{39}$$

where $\hat{\lambda}_N^2$ is the estimation of λ^2, and:

$$\widetilde{y}[n] = y[n] - \frac{1}{N}\sum_{n=1}^{N} y[n],\ \forall\, n \in \overline{1,N}. \tag{40}$$

Usually, after being evaluated, the SNR value is converted to *decibels* (dB).

Two types of SNRs can be evaluated for each output channel: one dealing with the colored noise estimation and another one considering the white noise estimation. For the first SNR, the colored noise is estimated by subtracting the simulated useful signal from the acquired output signal. The second SNR is evaluated by means of the estimated white noise, as obtained after subtracting the simulated output from the acquired output signal as well. In case of SISO-NARMAX model identified by using the i/o dataset $\mathbf{D}_N$ and a NARX auxiliary model, the two SNRs are evaluated as follows (according to Equations (14) and (21)):

$$\hat{\text{SNR}}_v = \sqrt{\frac{\sum_{n=1}^{N} \tilde{y}^2[n]}{\sum_{n=1}^{N} (y[n] - \hat{y}_u[n])^2}} = \sqrt{\frac{\sum_{n=1}^{N} \tilde{y}^2[n]}{\sum_{n=1}^{N} \left(y[n] - \boldsymbol{\varphi}_{u,y}^T[n]\, \hat{\boldsymbol{\theta}}_{N,u,y}\right)^2}};$$

$$\hat{\text{SNR}}_e = \sqrt{\frac{(N - n\theta) \sum_{n=1}^{N} \tilde{y}^2[n]}{N\mathbf{V}(\hat{\boldsymbol{\theta}}_N)}} = \frac{\sqrt{\frac{1}{N}\sum_{n=1}^{N} \tilde{y}^2[n]}}{\hat{\lambda}_N}. \quad (41)$$

Note that in Equation (41), the colored noise was estimated by means of the simulated useful output $\hat{y}_u$, with the estimated parameters of NARX part from NARMAX model (not with the NARX auxiliary model). The first SNR is very useful in testing the quality of the extracted useful data from the acquired data. Without considering the $\hat{\text{SNR}}_v$, it is possible to obtain high values of $\hat{\text{SNR}}_e$ not from the simulated useful output, $\hat{y}_u \equiv \boldsymbol{\varphi}_{u,y}^T\, \hat{\boldsymbol{\theta}}_{N,u,y}$, but from the MA-filtered estimated white noise (with the auxiliary model), $\hat{v}_N \equiv \hat{\boldsymbol{\varphi}}_{N,e}^T \hat{\mathbf{c}}_N$ (see again Equation (33)).

Good SISO-NARMAX models should have high values of both SNR. Since no upper limit of such a SNR is known, one can normalize its value by means of the following bijective (but nonlinear) function:

$$f(x) = \frac{1}{1+x},\ \forall\, x \in \mathbb{R}_+. \quad (42)$$

Equation (42) describes a hyperbola that compresses the $\mathbb{R}_+$ set into the interval $(0, 1]$. Moreover, in the vicinity of unit, the function is almost linear. Hence, the following corecriterion can be defined, referred to as *elementary fitness*:

$$F(\text{SNR}) = \frac{100\,\eta}{1 + \frac{\mu}{\text{SNR}}}\ [\%],\ \forall\, \text{SNR} \in \mathbb{R}_+^*, \quad (43)$$

where $\eta \in (0, 1]$ and $\mu \in \mathbb{R}_+^*$ are two parameters to be set by the user. The η parameter has the role of a weight in case several SNR types need to be employed. For a single SNR, $\eta = 1$, which allows the elementary fitness to take any value between 0% and 100%. The μ parameter is a freedom degree that can be set to control the sensitivity of the elementary fitness. For example, a good identification model should have SNR values of at least 15 dB, which means absolute values of more than 5.6234. This should correspond to a fitness value of at least 80% (for $\eta = 1$), meaning that the sensitivity can be set to $\mu = 5.6234 \times (100/80 - 1) \cong 1.4059$.

Of course, in case of 2-MISO-NARMAX model, a couple of SNR like those in (41), can be evaluated for each output channel.

To complete the framework, assume that the acquired i/o dataset is large enough to be split in two subsets: one for identification purposes, $\mathbf{D}_{N_{\text{id}}}^{\text{id}}$, and another one of validation purposes, $\mathbf{D}_{N_{\text{va}}}^{\text{va}}$ (with natural notations). The lengths of the acquisition horizons, N_{id} and N_{va}, can be identical or not. Then, the identification model obtained from the first subset is stimulated by the inputs of the second subset, and its responses are compared to the acquired outputs of the second subset as well. The optimal model has to prove good performances on both subsets simultaneously.

Now, the optimization criterion can be defined. Set, first, the following 4-length vectors (with the help of elementary fitness (43)):

$$\mathbf{f}_v = \frac{1}{\sqrt{2\left(\eta_{\text{id}}^2+\eta_{\text{va}}^2\right)}}\left[\begin{array}{cccc} \dfrac{\eta_{\text{id}}}{1+\dfrac{1}{\text{SNR}_{v,1}^{\text{id}}}} & \dfrac{\eta_{\text{id}}}{1+\dfrac{1}{\text{SNR}_{v,2}^{\text{id}}}} & \dfrac{\eta_{\text{va}}}{1+\dfrac{1}{\text{SNR}_{v,1}^{\text{va}}}} & \dfrac{\eta_{\text{va}}}{1+\dfrac{1}{\text{SNR}_{v,2}^{\text{va}}}} \end{array}\right];$$

$$\mathbf{f}_e = \frac{1}{\sqrt{2\left(\eta_{\text{id}}^2+\eta_{\text{va}}^2\right)}}\left[\begin{array}{cccc} \dfrac{\eta_{\text{id}}}{1+\dfrac{1}{\text{SNR}_{e,1}^{\text{id}}}} & \dfrac{\eta_{\text{id}}}{1+\dfrac{1}{\text{SNR}_{e,2}^{\text{id}}}} & \dfrac{\eta_{\text{va}}}{1+\dfrac{1}{\text{SNR}_{e,1}^{\text{va}}}} & \dfrac{\eta_{\text{va}}}{1+\dfrac{1}{\text{SNR}_{e,2}^{\text{va}}}} \end{array}\right]. \tag{44}$$

In Definitions (44), all the SNR values are computed according to Equation (41) for each output channel. Two weights are employed: one for the identification dataset (η_{id}) and another one for the validation dataset (η_{va}). They are proportional to the lengths N_{id} and N_{va}, respectively. Also, their sum over the 4-length vectors (44) is equal to unity. More specifically:

$$\eta_{\text{id}} = \frac{N_{\text{id}}}{2(N_{\text{id}}+N_{\text{va}})};\ \eta_{\text{va}} = \frac{N_{\text{va}}}{2(N_{\text{id}}+N_{\text{va}})}. \tag{45}$$

Due to the normalizing factor in front of both vectors, their norms are, at most, equal to unity. If $N_{\text{id}} = N_{\text{va}}$ then $\eta_{\text{id}} = \eta_{\text{va}} = 0.25$, and the normalizing factor becomes equal to 2.

Next, two partial fitnesses are defined (one for each type of SNR) as follows:

$$F_v = \frac{1}{2}\left(\frac{1}{1+\sigma_v}+\frac{1}{1+\frac{1}{\|f_v\|}}\right) \in (0,1],\ F_e = \frac{1}{2}\left(\frac{1}{1+\sigma_e}+\frac{1}{1+\frac{1}{\|f_e\|}}\right) \in (0,1], \tag{46}$$

where

$$\sigma_{\{v,e\}} = \frac{1}{2}\sqrt{\sum_{i=1}^{4}\left(\mathbf{f}_{\{v,e\}}[i]-\overline{f}_{\{v,e\}}\right)^2},\ \overline{f}_{\{v,e\}} = \frac{1}{4}\sum_{i=1}^{4}\mathbf{f}_{\{v,e\}}[i]. \tag{47}$$

Definitions (46) take into account that any partial fitness should decrease when the standard deviation of the corresponding vector increases, while it should increase when the norm of the corresponding vector increases. This would keep the fitness balanced on both channels and on both datasets. Thus, it is very unlikely that high partial fitness values will be obtained with very high elementary fitnesses on one output channel and/or one dataset and very small elementary fitnesses on the other output channel and/or the other dataset.

Finally, the (overall) fitness is as follows:

$$\mathbf{F} = \frac{100}{1+2\mu_F\frac{|F_v-F_e|}{F_v+F_e}}\ [\%]. \tag{48}$$

The sensitivity parameter μ_F is set as previously explained. Definition (48) was given by considering that the fitness should increase when the mean of the partial fitnesses increases, and it should decrease when the difference between the two partial fitnesses increases. Thus, it is suitable to keep balanced the performance for both types of fitness, which should ensure a good estimation of both colored and white noises on both output channels.

Following Definition (43), four more partial fitness values can be computed for each channel and dataset as follows:

$$\mathbf{F}_1^{\text{id}} = \frac{100}{1+\dfrac{\mu}{\text{SNR}_{e,1}^{\text{id}}}}\ [\%];\ \mathbf{F}_2^{\text{id}} = \frac{100}{1+\dfrac{\mu}{\text{SNR}_{e,2}^{\text{id}}}}\ [\%];\ \mathbf{F}_1^{\text{va}} = \frac{100}{1+\dfrac{\mu}{\text{SNR}_{e,1}^{\text{va}}}}\ [\%];\ \mathbf{F}_2^{\text{va}} = \frac{100}{1+\dfrac{\mu}{\text{SNR}_{e,2}^{\text{va}}}}\ [\%], \tag{49}$$

where a unique sensitivity parameter μ is set and can be different from μ_F. The partial fitness values allow analyzing in depth the model performance. Moreover, the weighted average of all four fitness values in (49) can be computed like below and compared to the value of the global fitness (48):

$$\overline{\mathbf{F}} = \eta_{\mathrm{id}}\left(\mathbf{F}_1^{\mathrm{id}} + \mathbf{F}_2^{\mathrm{id}}\right) + \eta_{\mathrm{va}}(\mathbf{F}_1^{\mathrm{va}} + \mathbf{F}_2^{\mathrm{va}}) \ \ [\%]. \tag{50}$$

The main caveat of fitness (50) is that high values of $\overline{\mathbf{F}}$ can be obtained even though one or more partial fitnesses (49) have small values. This type of model should be avoided, as it is suitable to obtain high values for all the partial fitnesses as well. Fitness (48) can help to reach for this goal. Moreover, if fitness (48) is high enough then not only the fitness (50) will be sufficiently high but also all four channel fitnesses will be grouped around their average. High average (50) with dispersed channel fitnesses correspond to smaller values of fitness (48), as F_v and F_e are not close to each other.

Nevertheless, it must be outlined that fitness (48) should not be employed to find models, like NARX or ARX, for which the regressor vector has no noise components. In case of NARX model, $\boldsymbol{\varphi}_e$ is missing from vector $\boldsymbol{\varphi}$ (see Equations (8) and (9) again), and, thus, no auxiliary model is needed. The simulated output signal of any such models, $\hat{y}$, is identical to the simulated useful signal, $\hat{y}_u$, which automatically involves $F_v = F_e$ and $\mathbf{F} = 100 \ \ [\%]$, regardless the model structure. Nevertheless, it would be unwise to remove NARX models from the possible optimal nonlinear candidates. To surpass this limitation, one can consider that the output generated by the NARX model after the simulation is still a useful one, while the difference from the acquired output is not an estimation of the white noise anymore but rather an estimation of a colored noise. Consequently, this noise can be obtained by means of an ARMA model. Hence, the NARX model can be enhanced by a linear component of ARMA type as an auxiliary component. Now, the enhanced NARX-ARMA model requires a white noise estimator for the linear part. Naturally, the estimator can be of AR type and belongs to the auxiliary part as well. Therefore, the hybrid nonlinear and linear NARX-ARMA-AR model (abbreviated by NARXA) can be considered as well.

The equations that allow identification and simulation of NARXA model in case of SISO systems are as follows:

$$\left\{\begin{array}{l} y[n] = \underbrace{\boldsymbol{\varphi}_{u,y}^T[n]\ \boldsymbol{\theta}_{u,y}}_{\hat{y}_u[n]} + v[n] \\ v[n] = \boldsymbol{\varphi}_{v,e}^T[n]\ \boldsymbol{\theta}_{v,e} \\ e[n] = \boldsymbol{\varphi}_v^T[n]\ \boldsymbol{\theta}_v \end{array}\right. , \ \forall\, n \in \mathbb{N}, \tag{51}$$

where $\boldsymbol{\varphi}_{u,y}$ and $\boldsymbol{\theta}_{u,y}$ have configurations corresponding to a pure NARX model, while

$$\left[\begin{array}{l} \boldsymbol{\varphi}_{v,e}^T[n] = \left[\begin{array}{cccccc} -v[n-1] & \cdots & -v[n-n\alpha] & e[n-1] & \cdots & e[n-n\beta] \end{array}\right] \\ \boldsymbol{\theta}_{v,e}^T = \left[\begin{array}{ccc|ccc} \alpha_1 & \cdots & \alpha_{n\alpha} & \beta_1 & \cdots & \beta_{n\beta} \end{array}\right] \end{array}\right. , \ \forall\, n \in \mathbb{N}, \tag{52}$$

$$\left[\begin{array}{l} \boldsymbol{\varphi}_v^T[n] = \left[\begin{array}{ccc} -v[n-1] & \cdots & -v[n-n\gamma] \end{array}\right] \\ \boldsymbol{\theta}_e^T = \left[\begin{array}{ccc} \gamma_1 & \cdots & \gamma_{n\alpha} \end{array}\right] \end{array}\right. , \ \forall\, n \in \mathbb{N}. \tag{53}$$

Definitions (52) correspond to the ARMA model of structural indices $n\alpha$ and $n\beta$, and Definitions (53) describe the AR model of structural index $n\gamma$. The ARMA models can be identified through the Minimum Prediction Error method. The OLS and, even better, the LevinsonDurbin method can be employed to identify the AR model. (See, for example, [1] or [2]). For the pure NARX model, the OLS method works, too.

The extension to the multivariable model is straightforward. Thus, the 2-MISO-NARXA model employs 10 major structural indices, their corresponding minor indices, plus 6 auxiliary minor indices: $n\alpha_1$, $n\alpha_2$, $n\beta_1$, $n\beta_2$, $n\gamma_1$, and $n\gamma_2$.

As Equation (51) suggest, the 2-MISO-NARXA model can provide two different types of simulation signals: useful and final. Thus, it is unlikely that $F_v = F_e$, which adds NARXA to the set of models to optimize (of NARMAX and ARMAX type), by means of the unique fitness (48). In the very improbable case when F_v is identical or very close to F_e, the linear component is inexistent or negligible, $\hat{v}$ is very close to a white noise, and the NARXA model (in fact, the pure NARX model), indeed, has great performance. This could occur in case that the NARX structure better matches the structure of the system to identify than the other tested models. A test would be the one in which the data are generated by a malicious user who employed some NARX model and a pseudo-random number generator to obtain the "acquired" data. In this case, evidently, the pure NARX model should be the best candidate to simulate such data.

2.4. On the Stability of Identification Models

Normally, since the black box to identify comes from real life, its dynamics is stable or, at most, at the stability limit (oscillating). Therefore, the fittest identification models are expected to be as close as possible to the black box, i.e., to exhibit intrinsic stability or, at most, oscillating behavior. Unstable models should lead to small values of the fitness. In case of 2-MISO-{NARMAX, NARXA, ARMAX} models, the stability can be tested by means of polynomials A_1 and A_2, which give the poles of all the filters (see Equations (16) and (17) and Figure 1 again). If their roots are inside or, at most, on the unit circle, then the model is considered to be stable.

Even the resulted fittest model is unstable, recall that such a model is devoted to automatic control. Thus, in a closed loop configuration, the controller will stabilize the overall dynamics. Moreover, if the controller is optimal, too, better behavior can be obtained in a closed loop than in an open loop.

Although analyzing the models stability from theoretical point of view is an interesting challenge, this approach rather constitutes the goal of a different article. Usually, theoretical results regarding stability are proven under some hypotheses that cannot always be verified in practice. In this paper, the stability of the identification models is only tested as mentioned above.

2.5. Optimization of Structural Indices through Cuckoo Search Algorithm

The fitness (48) only depends on the structural indices since, once such indices being set, the model parameters are determined through the MS-EOLS method. Assume that the indices are gathered in two vectors (with non-negative integer values): $\boldsymbol{\gamma}_{\mathrm{M}}$ and $\boldsymbol{\gamma}_{\mathrm{m}}$. More specifically, $\boldsymbol{\gamma}_{\mathrm{M}}$ includes all the 20 major indices, while $\boldsymbol{\gamma}_{\mathrm{m}}$ encompasses all the minor indices, from both models NARMAX (main) and NARX (auxiliary). By analogy with Genetic Algorithms, $\boldsymbol{\gamma}_{\mathrm{M}}$ and $\boldsymbol{\gamma}_{\mathrm{m}}$ can be seen as two genes of a large chromosome $\boldsymbol{\gamma} = \begin{bmatrix} \boldsymbol{\gamma}_{\mathrm{M}} & \boldsymbol{\gamma}_{\mathrm{m}} \end{bmatrix}^T$. Thus, the granular optimization problem can be formulated as follows:

$$\max_{\boldsymbol{\gamma}_{\mathrm{M}} \in \mathbb{N}^{20},\ \boldsymbol{\gamma}_{\mathrm{m}} \in \mathbb{N}^{N_{\mathrm{m}}}} \mathbf{F}[\boldsymbol{\gamma}_{\mathrm{M}}, \boldsymbol{\gamma}_{\mathrm{m}}], \tag{54}$$

where N_{m} is the number of minor indices (variable, depending on the values of the major indices). For the full nonlinear model, all the major indices are at least equal to unity (see again Equations (16) and (17)). However, partial nonlinear models, or even the linear models, should be tested as well. Therefore, by convention, if a major index is null, the corresponding term is missing from the general equation of the model. For example, if all the major indices are null, the linear models of ARMAX or ARX type are considered. Another example is concerned with NARXA model type. Here, the null minor index nc should be replaced by the triplet $\{n\alpha, n\beta, n\gamma\}$, as explained at the end of the previous subsection.

To solve the problem (54), the search space needs to be delimited. This means setting some upper bounds for all the structural indices. Such bounds are naturally imposed, on one hand, by the true internal structure of the black box to identify, and, on the other hand, by the length of the acquisition horizon for the identification dataset, N_{id}. In many practical cases, the major indices cannot overpass the value of 5, whereas the minor indices cannot take values beyond $N_{\mathrm{id}}/2$ (or even $N_{\mathrm{id}}/3$).

The CSA can help solving the problem (54). This metaheuristic belongs to the population-based class and was inspired from cuckoo's breeding and survival strategy. The corresponding numerical procedure was originally developed in [47]. As well known, cuckoo is a species of migratory bird that stands out especially for brood parasitism. This means it lays its eggs in the nests of other host birds (usually of other species) to transfer them the responsibility of hatching and raising the chicks. If the host bird discovers the intruded egg, then one of the following three natural phenomena can occur: it throws the alien egg out of the nest; it abandons the nest and builds another one in a different location; it accepts the alien egg among its own eggs. To increase the chances of acceptance for its own eggs by other host birds and, consequently, to enhance the breeding probability, some species of cuckoo have developed the ability to find host nests with similar natural characteristics of the eggs (in terms of pattern and/or color). At the same time, the parasitic cuckoo can carefully choose the egg-laying moment to be perfectly synchronized with that of the host bird. As the cuckoo chick hatches earlier than the host birds, it instinctively removes the host bird's chicks from the nest, thus ensuring it a larger share of the food from the host bird.

The behavior of a cuckoo can be simulated by a numerical algorithm, which aims to find the best host nest for the cuckoo to lay its eggs so that these eggs survive as long as possible to reach maturity (i.e., to maximize its eggs survival rate). A simplified implementation of the algorithm is based on the following assumptions:

(a) Each cuckoo can only lay one egg at a time and put it in a randomly chosen host nest. This means the number of cuckoos, host nests, and eggs is the same. The host nest represents a candidate solution to the optimization problem. Several host nests can simultaneously be searched for by a flock of cuckoos. This means a population of virtual cuckoos can be employed in the algorithm.

(b) The best host nests, with the highest egg quality (i.e., accepted by the host bird), will be carried on to the next stage of the development (the next generation or iteration) based on an elitist survival mechanism. The cost function (such as $\mathbf{F}$) measures the host nest quality, depending on its position in the search space. In this phase, the host nest is abandoned only if a host nest with a better quality was found. This strengthens the exploitation of the search space and contributes to the algorithm convergence towards an optimal point (host nest).

(c) Cuckoo is a suspicious mind bird. Therefore, strangely enough, a second phase occurs. Normally, the worst host nest (in terms of cost function) should be replaced by other newly generated host nest. But this does not always happen. Sometimes, good host nests are abandoned. A probability $P_a \in [0,1]$ of a nest to be agreed by the cuckoo and to become the host of its egg can be set. Thus, there is a probability of $1 - P_a$ that the cuckoo chooses not to leave its egg in that nest, regardless the nest quality, in the hope of finding a better nest. The opposite number $1 - P_a$ is referred to as renewal rate of the nests population. Usually, $P_a < 1 - P_a$, which means the cuckoo is very selective with the future host nests of its eggs. Statistically, the majority of the nests playing the role of possible hosts are under question whether they will be preserved in the population or not. This means the exploitation of the search space is allowed, which increases the chances to find a global, or nearly global, solution.

Assume that the population of $P \in \mathbb{N}^*$ host nests evolves by changing its position during a fixed number of generations. Each host nest $p \in \overline{1,P}$ takes a position $\mathbf{x}_p^k$ at generation $k \in \mathbb{N}$, in the search space $\boldsymbol{S} \subseteq \mathbb{R}^{nx}$, where $nx \in \mathbb{N}^*$ is the length of the position vector (seen as the decision variable of the optimization problem). Each component i of $\mathbf{x}_p^k$

varies within $[x_{\min,i}, x_{\max,i}]$, so that the search space can be specified as a cartesian product of compact intervals as follows:

$$\boldsymbol{S} = \prod_{i=1}^{nx} [x_{\min,i}, x_{\max,i}]. \tag{55}$$

In CSA, as in many nature-inspired optimization algorithms, the cuckoo's behavior in its search for a suitable host nest (i.e., the strategy of laying eggs) can be approximated by the so-called Lévy flight, where the flight direction is generated by a uniform distribution, and the step size is generated by means of Lévy distribution. One of the most efficient manners to implement a Lévy distribution is by applying Mantegna's algorithm [49], which introduces the following expression of the step size s:

$$s = u|v|^{-1/\lambda}. \tag{56}$$

where $\lambda = 1.5$ is the Lévy characteristic parameter, v is a Gaussian (normally distributed) random variable of null mean and unit variance, while u is also a Gaussian random variable of null mean but with variance:

$$\sigma_u^2 = \left(\frac{\Gamma(1+\lambda) \cdot \sin(\pi\lambda/2)}{\Gamma((1+\lambda)/2) \cdot \lambda \cdot 2^{(\lambda-1)/2}} \right)^{1/\lambda} \tag{57}$$

In Definition (57), the well-known Gamma function is employed.

Following Lévy's description of cuckoos' flight, the possible next position of the host nest, $\mathbf{x}_p^{k+1}$, is generated by updating the current position with an additive correction, as follows:

$$\mathbf{x}_p^{k+1} = \mathbf{x}_p^k + \alpha \cdot s_p^k \cdot \left(\mathbf{x}_p^k - \mathbf{x}_{\text{best}}^k \right), \ \forall\, p \in \overline{1,P}, \ \forall\, k \in \mathbb{N} \tag{58}$$

where s_p^k is the flight step size generated for each host nest at each generation by using Equation (56), α is a scaling factor that allows the user to correctly scale s_p^k, while $\mathbf{x}_{\text{best}}^k$ is the best host nest position found up to the current generation k. Usually, a small step $\alpha \in [0.01, 0.1]$ can be chosen to avoid jumping out of the variation domain. Moreover, in this way, $\mathbf{x}_p^{k+1}$ is searched around $\mathbf{x}_{\text{best}}^k$, which enforces the exploitation. Although this is performed to the detriment of exploration, it could prevent oscillations and speed up the search. Nevertheless, if the new position $\mathbf{x}_p^{k+1}$ goes beyond the variation limits of the search space, it can be made viable (i.e., brought back into the search space) through various techniques. The simplest manner of making new positions viable consists of replacing $x_{p,i}^{k+1} \notin [x_{\min,i}, x_{\max,i}]$ $(i \in \overline{1,nx})$ by $x_{\min,i}$, if $x_{p,i}^{k+1} < x_{\min,i}$ or with $x_{\max,i}$, if $x_{p,i}^{k+1} > x_{\max,i}$. However, this saturation technique can lead to many lost iterations, where the new position is glued to the search space frontier. Another more suitable technique is described in [50] and relies on the idea of computing a reminder from an outsider component $x_{p,i}^{k+1}$ to the length of the corresponding admissible interval $[x_{\min,i}, x_{\max,i}]$, i.e., $x_{\max,i} - x_{\min,i}$. Thus, the new viable component belongs now to $[x_{\min,i}, x_{\max,i}]$ and is not necessarily saturated to one of the interval limits. After this necessary operation, the possible new position is evaluated through the cost function. It really becomes the new position if the cost function returns a better value than that for the current position. Otherwise, the current position is preserved and becomes the new position as well.

According to assumption (c), after the new generation was set (by means of Equation (58)), the renewal phase should occur. Assume that all the nests $\left\{ \mathbf{x}_p^{k+1} \right\}_{p \in \overline{1,P}}$ are stored in a matrix, on columns: $\mathbf{X}^{k+1} = \begin{bmatrix} \mathbf{x}_1^{k+1} & \cdots & \mathbf{x}_P^{k+1} \end{bmatrix}$. The current element of the matrix is $x_{i,p}^{k+1}$, where $i \in \overline{1,nx}$ is the row index, while $p \in \overline{1,P}$ is the column index. Then, $x_{i.p}^{k+1}$ has P_a chances to survive and $1 - P_a$ bad luck to be abandoned, being replaced by another component. If replacement must occur (recall that, usually, $1 - P_a > P_a$), then the new component is computed like below:

$$x_{i,p}^{k+1} \leftarrow x_{i,p}^{k+1} + r^{k+1}\left(x_{i,n_p}^{k+1} - x_{i,m_p}^{k+1}\right) \tag{59}$$

where r^{k+1} is a uniformly generated random number in the interval $[0,1]$, while $\mathbf{x}_{n_p}^{k+1}$ and $\mathbf{x}_{m_p}^{k+1}$ are two randomly selected host nests. Note that r^{k+1} is the same for all the nests in the population, whereas the couple $\left\{\mathbf{x}_{n_p}^{k+1}, \mathbf{x}_{m_p}^{k+1}\right\}$ is selected for each nest of index $p \in \overline{1,P}$. Equation (59) suggests that for a nest $\mathbf{x}_p^{k+1}$, only some of its components can change, while the other ones can remain untouched. However, nothing is for sure. Overall, on the long run, since $1 - P_a > P_a$, most components change.

After or during the renewal phase, all new nests are tested for viability. All the outliers are made viable by the chosen technique. Then, the quality of each nest is evaluated again by means of the cost function. The former nest $\mathbf{x}_p^k$ is really replaced only if the new nest $\mathbf{x}_p^{k+1}$ has a better quality.

This mechanism allows exploration of the search space beyond the vicinity of the best found host nest, without losing the fittest nests and, at the same time, preventing the numerical procedure to be trapped into a local optimum vicinity.

By suitably setting the probability P_a, the user can reach a trade-off between the exploitation and exploration mechanisms. According to some scientists, the CSA is one of the simplest and most efficient metaheuristics, easy to implement and use.

3. Design and Implementation Details of Numerical Procedures

The methods described within the previous section allow for designing numerical procedures to be implemented by using a programming environment. This section gives some details regarding the design and implementation of these procedures. Although the explanation is oriented toward using the MATLAB$^{\text{TM}}$ programming language, other high-level languages can be employed as well (such as Python or C++).

The following algorithms are addressed in the sequel:

- identification through MS-EOLS;
- simulation of identification model by multi-step prediction (msp);
- CSA.

The Gram–Schmidt method is easy to implement (it only consists of Equation (23) and the corresponding initialization). Also, the basic OLS method can be implemented by directly using Equation (30). For both methods, no further explanations are necessary.

3.1. MS-EOLS Numerical Procedure

The main steps of the MS-EOLS procedure are listed in Algorithm 1.

Note that if at least one filtering step was performed, the returned polynomials $\mathrm{C}_{\mathrm{f},1}^{\mathrm{opt}}$ and $\mathrm{C}_{\mathrm{f},2}^{\mathrm{opt}}$ come from the previous 2-MISO-NARMAX model and not from the optimal model. (Evidently, if the optimal models are directly found, without filtering, both polynomials are equal to unity—see the third step of initialization.) This is the reason the two polynomials are returned separately. Although not explicitly specified, the variances of the two white noises are estimated (and returned) as well by means of Equation (21).

Algorithm 1 is devoted to 2-MISO-NARMAX model. However, the numerical procedure can easily be adapted to identify linear 2-MISO-ARMAX models. This time, the auxiliary model is of ARX type.

In case of 2-MISO-NARX models, no auxiliary model is necessary. Therefore, the identification of such models can simply be realized by means of the OLS method.

Algorithm 1 MS-EOLS Procedure to Identify a 2-MISO-NARMAX Model

```
➢ Inputs:
    • the identification i/o data set: $\mathbf{D}^{\text{id}}_{N_{\text{id}}} = \{u_1[n], u_2[n], y_1[n], y_2[n]\}_{n\in\overline{1,N_{\text{id}}}}$;
    • the 10 major indices of 2-MISO-NARMAX model to identify: $P_1,\ P_2,\ M_{1,1},\ M_{1,2},\ M_{2,1},\ M_{2,2},\ K_{1,1},\ K_{1,2},\ K_{2,1},\ K_{2,2}$;
    • the 10 major indices of auxiliary 2-MISO-NARX model to identify: $P_1^{\text{aux}},\ P_2^{\text{aux}},\ M_{1,1}^{\text{aux}},\ M_{1,2}^{\text{aux}},\ M_{2,1}^{\text{aux}},\ M_{2,2}^{\text{aux}},\ K_{1,1}^{\text{aux}},\ K_{1,2}^{\text{aux}},\ K_{2,1}^{\text{aux}},\ K_{2,2}^{\text{aux}}$;
    • the minor indices of 2-MISO-NARMAX model to identify: $na_1, na_2, \{na_{1,p}\}_{p\in\overline{0,P_1-1}}, \{na_{2,p}\}_{p\in\overline{0,P_2-1}}, nb_{1,1}, nb_{1,2}, nb_{2,1}, nb_{2,2}, \{nb_{1,1,m}\}_{m\in\overline{0,M_{1,1}-1}}, \{nb_{1,2,m}\}_{m\in\overline{0,M_{1,2}-1}}, \{nb_{2,1,m}\}_{m\in\overline{0,M_{2,1}-1}}, \{nb_{2,2,m}\}_{m\in\overline{0,M_{2,2}-1}}, \{nd_{1,1,k}\}_{k\in\overline{0,K_{1,1}-1}}, \{nd_{1,2,k}\}_{k\in\overline{0,K_{1,2}-1}}, \{nd_{2,1,k}\}_{k\in\overline{0,K_{2,1}-1}}, \{nd_{2,2,k}\}_{k\in\overline{0,K_{2,2}-1}}, nc_1, nc_2$;
    • the minor indices of auxiliary 2-MISO-NARX model to identify: $na_1^{\text{aux}}, na_2^{\text{aux}}, \left\{na_{1,p}^{\text{aux}}\right\}_{p\in\overline{0,P_1^{\text{aux}}-1}}, \left\{na_{2,p}^{\text{aux}}\right\}_{p\in\overline{0,P_2^{\text{aux}}-1}}, nb_{1,1}^{\text{aux}}, nb_{1,2}^{\text{aux}}, nb_{2,1}^{\text{aux}}, nb_{2,2}^{\text{aux}}, \left\{nb_{1,1,m}^{\text{aux}}\right\}_{m\in\overline{0,M_{1,1}^{\text{aux}}-1}}, \left\{nb_{1,2,m}^{\text{aux}}\right\}_{m\in\overline{0,M_{1,2}^{\text{aux}}-1}}, \left\{nb_{2,1,m}^{\text{aux}}\right\}_{m\in\overline{0,M_{2,1}^{\text{aux}}-1}}, \left\{nb_{2,2,m}^{\text{aux}}\right\}_{m\in\overline{0,M_{2,2}^{\text{aux}}-1}}, \left\{nd_{1,1,k}^{\text{aux}}\right\}_{k\in\overline{0,K_{1,1}^{\text{aux}}-1}}, \left\{nd_{1,2,k}^{\text{aux}}\right\}_{k\in\overline{0,K_{1,2}^{\text{aux}}-1}}, \left\{nd_{2,1,k}^{\text{aux}}\right\}_{k\in\overline{0,K_{2,1}^{\text{aux}}-1}}, \left\{nd_{2,2,k}^{\text{aux}}\right\}_{k\in\overline{0,K_{2,2}^{\text{aux}}-1}}$;
    • the accuracy threshold: $\eta > 0$;
    • the maximum number of filtering steps: $L \in \mathbb{N}^*$ (by default, $L = 10$).
Initialization:
    for $j = 1:2$
    1. Call the OLS numerical procedure to identify the MISO-NARX model on output channel $j$.
        • Notes
            1. According to Equation (31), the generic equation of model is
               $$y_j[n] = \boldsymbol{\psi}^T_{u,y,j}[n]\,\boldsymbol{\theta}^{\text{aux}}_{u,y,j} + \varepsilon_j\left[n, \boldsymbol{\theta}^{\text{aux}}_{u,y,j}\right],\ \forall\, n \in \mathbb{N},$$
               where $\varepsilon_j$ stands for model error and will approximate the white noise, after identification. The vector $\boldsymbol{\psi}_{u,y,j}$ has to be evaluated for each normalized instant $\forall\, n \in \overline{1, N_{\text{id}}}$, by using the i/o data set $\mathbf{D}^{\text{id}}_{N_{\text{id}}}$. By convention, signals take null values for $\forall\, n \leq 0$. It is suitable to stack all regressor vectors into the matrix $\boldsymbol{\Psi}_{u,y,j}$ and all acquired output values into the vector $\mathbf{Y}_j$, if possible.
            2. According to Equation (20), the OLS procedure returns the optimal parameters vector:
               $$\hat{\boldsymbol{\theta}}^{\text{aux}}_{u,y,j} = \left(\boldsymbol{\Psi}^T_{u,y,j}\boldsymbol{\Psi}_{u,y,j}\right)^{-1}\boldsymbol{\Psi}^T_{u,y,j}\mathbf{Y}_j = \left[\sum_{n=1}^{N}\boldsymbol{\psi}_{u,y,j}[n]\boldsymbol{\psi}^T_{u,y,j}[n]\right]^{-1}\sum_{n=1}^{N}\boldsymbol{\psi}_{u,y,j}[n]y_j[n].$$
               The noise variance is not needed for this model. The matrix is inverted through Gram-Schmidt orthogonalization procedure.
            3. Since the filtering operation might occur, one can change the notations into the model equation above, to outline the number of filtering steps:
               $$y_j^l[n] = \left(\boldsymbol{\psi}^l_{u,y,j}\right)^T[n]\,\boldsymbol{\theta}^{\text{aux},l}_{u,y,j} + \varepsilon_j^l\left[n, \boldsymbol{\theta}^{\text{aux},l}_{u,y,j}\right],\ \forall\, n \in \mathbb{N},$$
               where $l \in \overline{0, L}$. At this initial step, $l = 0$ (no filtering was applied).
    2. Set to infinity the initial minimum value of quadratic criterion: $\mathbf{V}_{\min,j} = \infty$.
    3. Set the initial characteristic polynomial of filter to unit: $\mathrm{C}^0_{\mathrm{f},j}\left(\mathrm{q}^{-1}\right) = 1$.
end for $j = 1:2$
```

Algorithm 1 *Cont.*

```
Main loop:
for $j = 1:2$
 for $l = 0:L$
  1.1. Use the current auxiliary model to estimate the white noise, according to Equation (32):
       $\varepsilon_j^l\left[n, \theta^{\mathrm{aux},l}_{u,y,j}\right] = y_j^l[n] - \left(\boldsymbol{\psi}^l_{u,y,j}\right)^T[n]\,\theta^{\mathrm{aux},l}_{u,y,j},\ \forall n \in \overline{1,N_{\mathrm{id}}}.$
  1.2. Build the noise component of vector (8), following Definition (9):
       $\boldsymbol{\varphi}^l_{e,j}[n] = \left[\varepsilon_j^l\left[n-1, \theta^{\mathrm{aux},l}_{u,y,j}\right] \quad \cdots \quad \varepsilon_j^l\left[n-nc, \theta^{\mathrm{aux},l}_{u,y,j}\right]\right]^T,\ \forall n \in \overline{1,N_{\mathrm{id}}}.$
  1.3. Call the OLS numerical procedure to identify the MISO-NARMAX model. According to
       Equation (20), the optimal parameters are:
       $\begin{bmatrix} \hat{\theta}^l_{u,y,j} \\ \hat{\mathbf{c}}^l_j \end{bmatrix} = \left(\sum_{n=1}^{N}\begin{bmatrix} \boldsymbol{\varphi}_{u,y,j}[n]\boldsymbol{\varphi}^T_{u,y,j}[n] & \boldsymbol{\varphi}_{u,y,j}[n]\left(\boldsymbol{\varphi}^l_{e,j}[n]\right)^T \\ \boldsymbol{\varphi}^l_{e,j}[n]\boldsymbol{\varphi}^T_{u,y}[n] & \boldsymbol{\varphi}^l_{e,j}[n]\left(\boldsymbol{\varphi}^l_{e,j}[n]\right)^T \end{bmatrix}\right)^{-1}\left(\sum_{n=1}^{N}\begin{bmatrix} \boldsymbol{\varphi}_{u,y,j}[n] \\ \boldsymbol{\varphi}^l_{e,j}[n] \end{bmatrix} y_j[n]\right).$
  1.4. Estimate again the white noise, this time by means of identified MISO-NARMAX model:
       $e_j^l[n] = y_j[n] - \boldsymbol{\varphi}^T_{u,y,j}[n]\,\hat{\theta}^l_{u,y,j} - \left(\boldsymbol{\varphi}^l_{e,j}[n]\right)^T\hat{\mathbf{c}}^l_j,\ \forall n \in \overline{1,N_{\mathrm{id}}}.$
  1.5. Evaluate the quadratic criterion according to Equation (33):
       $\mathbf{V}\left(\hat{\theta}^l_{u,y,j}, \hat{\mathbf{c}}^l_j\right) = \sum_{n=1}^{N}\left(e_j^l[n]\right)^2.$
  1.6. If $\mathbf{V}\left(\hat{\theta}^l_{u,y,j}, \hat{\mathbf{c}}^l_j\right) < \eta$, then:
       1.6.1. Update $\mathbf{V}_{\min,j} = \mathbf{V}\left(\hat{\theta}^l_{u,y,j}, \hat{\mathbf{c}}^l_j\right)$ and store in memory the optimal NARMAX and NARX
              models, together with the current characteristic polynomial of all-poles filter (for
              simulation purpose):
              $\hat{\theta}^{\mathrm{opt}}_{u,y,j} = \hat{\theta}^l_{u,y,j},\ \hat{\mathbf{c}}^{\mathrm{opt}}_j = \hat{\mathbf{c}}^l_j,\ \hat{\lambda}^2_{\min} = \frac{\mathbf{V}_{\min,j}}{N_{\mathrm{id}} - n\theta_j};$
              $\theta^{\mathrm{aux,opt}}_{u,y,j} = \theta^{\mathrm{aux},l}_{u,y,j},\ \mathrm{C}^{\mathrm{opt}}_{\mathrm{f},j}(\mathrm{q}^{-1}) = \mathrm{C}^l_{\mathrm{f},j}(\mathrm{q}^{-1}).$
       1.6.2. Break the inner **for** cycle, as the identification of optimal model was accomplished.
  1.7. Otherwise:
       1.7.1. If $\mathbf{V}\left(\hat{\theta}^l_{u,y,j}, \hat{\mathbf{c}}^l_j\right) < \mathbf{V}_{\min,j}$, proceed like in step 1.6.1.
       1.7.2. Update the characteristic polynomial of all-poles filter:
              $\mathrm{C}^{l+1}_{\mathrm{f},j}\left(\mathrm{q}^{-1}\right) = 1 + \left[\mathrm{q}^{-1} \quad \mathrm{q}^{-2} \quad \cdots \quad \mathrm{q}^{-nc_j}\right]\hat{\mathbf{c}}^l_j.$
       1.7.3. Filter the i/o data set : $u_j^{l+1} \equiv \frac{1}{\mathrm{C}^{l+1}_{\mathrm{f},j}(\mathrm{q}^{-1})}u_j,\ y_j^{l+1} \equiv \frac{1}{\mathrm{C}^{l+1}_{\mathrm{f},j}(\mathrm{q}^{-1})}y_j.$
       1.7.4. Filter the initial regressors vector of auxiliary model: $\boldsymbol{\psi}^{l+1}_{u,y,j} \equiv \frac{1}{\mathrm{C}^{l+1}_{\mathrm{f},j}(\mathrm{q}^{-1})}\boldsymbol{\psi}^0_{u,y,j}.$
              * Note
                In fact, the vector $\boldsymbol{\psi}^{l+1}_{u,y,j}$ includes filtered data from signals $u_j^{l+1}$ and $y_j^{l+1}$ (as
                obtained in step 1.7.3). Moreover, filtering the products between such signals also
                is necessary. They cannot be obtained by performing products between filtered
                i/o data (see again Equations (35) and (36)).
       1.7.5. Call the OLS numerical procedure to identify the auxiliary MISO-NARX model.
              One obtains:
              $\hat{\theta}^{\mathrm{aux},l+1}_{u,y,j} = \left[\sum_{n=1}^{N}\boldsymbol{\psi}^{l+1}_{u,y,j}[n]\left(\boldsymbol{\psi}^{l+1}_{u,y,j}[n]\right)^T\right]^{-1}\sum_{n=1}^{N}\boldsymbol{\psi}^{l+1}_{u,y,j}[n]y_j^{l+1}[n].$
 end for $l = 0:L$
end for $j = 1:2$
```

Algorithm 1 *Cont.*

```
⋞ Outputs:
    • the optimal 2-MISO-NARMAX model;
    • the optimal auxiliary 2-MISO-NARX model;
    • the characteristic polynomials of all-poles filters to employ in simulations: $C_{f,1}^{opt}$, $C_{f,2}^{opt}$.
```

3.2. Numerical Procedure to Simulate MISO-NARMAX Models

Generally speaking, there are two approaches when simulating an identification model, depending on the type of output data employed to build the regression vectors, like $\boldsymbol{\varphi}$ or $\boldsymbol{\psi}$. For example, the ARMAX model (1) can be expressed by the regression Equation (7) as well. This time, the regression vector $\boldsymbol{\varphi}$ is:

$$\begin{aligned}\boldsymbol{\varphi}[n] = &\left[\begin{array}{ccccc} \boldsymbol{\varphi}_y^T[n] & | & \boldsymbol{\varphi}_u^T[n] & | & \boldsymbol{\varphi}_e^T[n] \end{array}\right]^T = \\ = &\left[\begin{array}{ccccc} -y[n-1] & -y[n-2] & \cdots & -y[n-na] & | \\ u[n-1] & u[n-2] & \cdots & u[n-nb] & | \\ e[n-1] & e[n-2] & \cdots & e[n-nc] & \end{array}\right]^T, \quad \forall\, n \in \mathbb{N}\,. \end{aligned} \tag{60}$$

When carefully looking at Equation (60), one can notice that in spite of notation "$\boldsymbol{\varphi}[n]$", the most recent sample of any signal inside $\boldsymbol{\varphi}$ is the one taken for the normalized instant $n-1$ and not n. This involves that, after the identification of ARMAX model, from Equation (7), a predicted output value can be obtained by simply ignoring the current sample of the white noise. More specifically:

$$\hat{y}[n] = \boldsymbol{\varphi}[n]\hat{\boldsymbol{\theta}}_N,\ \forall\, n \in \overline{1,N}, \tag{61}$$

provided that an estimation of the white noise is available for the previous normalized instants $n-1$, $n-2$, etc. (e.g., by means of an auxiliary ARX model). This operation is possible thanks to the fact that, in Equation (61), the only sample at current normalized instant n is the predicted output, while all the other samples are regressed. (The prediction can be realized not only on the acquisition horizon $\overline{1,N}$, but also beyond the upper limit N, if acquired input samples are available.)

Focus now on the first component of vector $\boldsymbol{\varphi}$ from Equation (60), namely, $\boldsymbol{\varphi}_y$, which includes regressed output values. In many scientific articles, this component is evaluated by using acquired output data. This means that, although the predicted value $\hat{y}[n]$ is available, when computing the next predicted value $\hat{y}[n+1]$, since the acquired sample $y[n]$ is available as well, the first component of $\boldsymbol{\varphi}$ is still updated to

$$\boldsymbol{\varphi}_y[n+1] = -\big[y[n] \quad y[n-1] \quad \cdots \quad y[n-na+1]\big]. \tag{62}$$

The insertion of $\hat{y}[n]$ into $\boldsymbol{\varphi}_y$ is enforced only in the case of prediction beyond the instant $N+1$ because no acquired output data are available. For example, to predict $\hat{y}[N+2]$, the component below is needed:

$$\boldsymbol{\varphi}_y[N+2] = -\big[\hat{y}[N+1] \quad y[N] \quad \cdots \quad y[N-na+2]\big]. \tag{63}$$

This technique is referred to as *one-step prediction* (osp). If the identification model is optimal and valid, then the simulation on the acquisition horizon by employing the osp technique can lead to predicted outputs extremely close to the acquired outputs, as no prediction errors accumulate.

The other technique is based on the possibility to mainly employ predicted values when updating the component $\boldsymbol{\varphi}_y$. Thus, the general structure of component $\boldsymbol{\varphi}_y$ is:

$$\boldsymbol{\varphi}_y[n] = -\big[\hat{y}[n-1] \quad \hat{y}[n-2] \quad \cdots \quad \hat{y}[n-na]\big]. \tag{64}$$

The acquired output data are not useless, though. First, they have been employed to identify the optimal model. Second, for the first predicted values, an initialization is needed. As full initialization, one can extract the first na output data and start prediction from instant $na+1$. In this case, acquired samples are mixed with predicted values in construction of $\boldsymbol{\varphi}_y$, like in Equation (63), until the prediction instant moves beyond $2na$, when $\boldsymbol{\varphi}_y$ only includes predicted output values, like in Equation (64). Another initialization manner, the hardest one, is to compute the average of output data, $\overline{y}$, as the first predicted value. Moreover, the first component $\boldsymbol{\varphi}_y$ can be set like below, for the next prediction:

$$\boldsymbol{\varphi}_y[2] = \begin{bmatrix} -\overline{y} & 0 & \cdots & 0 \end{bmatrix}. \tag{65}$$

With Equations (61) and (65), the second predicted value is:

$$\hat{y}[2] = -a_1\overline{y} + b_1 u[1] + c_1\hat{e}[1], \tag{66}$$

where $\hat{e}[1]$ is the first estimated value of the white noise. As the prediction instant increases, the zeros of $\boldsymbol{\varphi}_y$ are gradually replaced by predicted values, until they completely vanish. Of course, one expects that in the beginning, the predicted values will not be so accurate. However, if the identification model has a good quality, the accuracy of the prediction may improve along the simulation horizon. Partial initializations can be adopted as well (i.e., with more than one sample but fewer than na samples). In any case, the null initialization should be avoided because the acquired output signals usually start with non-null samples.

This technique is referred to as *multi-step prediction* (msp). Its main caveat is this: the more reduced the initialization size, the larger the prediction errors, which, by difference from the osp technique, accumulate along the simulation horizon.

Nevertheless, if the identification model is devoted to integration into an application of automatic control, then the msp technique is more suitable for simulations than the osp technique. The main argument is that, in the absence of any internal controller, the identified black box (which often is a process to be controlled) receives no direct feedback from its output(s). Hence, no output data determine its dynamics. Since the associated model was identified in open loop, naturally, the model also should not make use of the measured output(s) in simulation to follow the black box dynamics as close as possible. Another argument relies on the filtering operation. When computing the output of a rational filter described, for example, by the system function:

$$\mathrm{H}\left(\mathrm{q}^{-1}\right) = \frac{\mathrm{B}(\mathrm{q}^{-1})}{\mathrm{A}(\mathrm{q}^{-1})}, \tag{67}$$

one solves, in fact, a difference equation:

$$y[n] = -a_1 y[n-1] - \cdots - a_{na} y[n-na] + b_1 u[n-1] + \cdots + b_{nb} u[n-nb],\ \forall\, n \in \mathbb{N}. \tag{68}$$

By Equation (68), the output y is computed at every instant (starting from some initialization) without using previously acquired output samples. In this aim, only input samples are necessary to be specified. As Figure 1 illustrates, filtering is essential when simulating an identification model, especially if integrated into a scheme of automatic control. Moreover, if the msp technique would be employed to compute the fitness (47), the optimal 2-MISO-NARMAX model could be more reliable in the automatic control application than by using the osp technique.

Consequently, the scheme in Figure 3 can be employed to simulate the 2-MISO-NARMAX model, together with its auxiliary 2-MISO-NARX model, for each output channel.

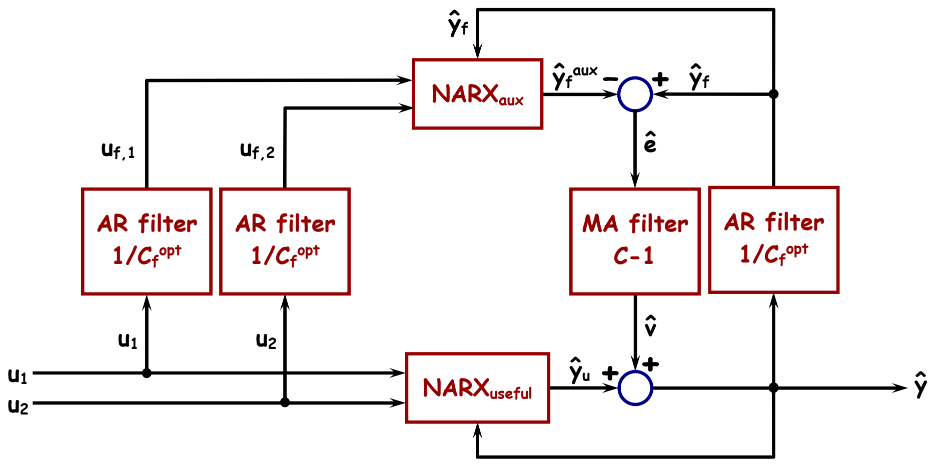

Figure 3. Simulation scheme of a 2-MISO-NARMAX model to be employed in the evaluation of fitness.

To implement this scheme, only one MISO-NARX routine and one filtering routine are necessary. The first routine is invoked for any of the blocks NARX$_{\text{useful}}$ and NARX$_{\text{aux}}$. It returns the output by computing products, like $\boldsymbol{\varphi}_{u,y}^{T}[n]\ \boldsymbol{\theta}_{u,y}$ or $\boldsymbol{\psi}_{u,y}^{T}[n]\ \boldsymbol{\theta}_{u,y}^{\text{aux}}$. The second one is based on Equation (64), where either the AR or the MA part is missing. Note that the AR filters are using $C_{\text{f},\{1,2\}}^{\text{opt}}$ polynomials, while the MA filter is designed by means of C polynomials extracted from the optimal 2-MISO-NARMAX model. Also, the notations $\hat{e}$ and $\hat{v}$ stand here for the estimated white and colored noises, respectively.

The numerical procedure based on the scheme in Figure 3 and the msp technique is summarized in Algorithm 2.

The procedure of Algorithm 2 can be employed for any output channel of 2-MISO-NARMAX model and for any of the two datasets $\mathbf{D}_{N_{\text{id}}}^{\text{id}}$, $\mathbf{D}_{N_{\text{va}}}^{\text{va}}$. Also, the 2-MISO-ARMAX linear model can be simulated with the same procedure. This time, the structure of the regressor vectors is simpler, as the major indices are missing.

Algorithm 2 Simulation Procedure of a MISO-NARMAX Model, Based on msp Technique

➢ **Inputs:**
- the identification model of MISO-NARMAX type (one output channel);
- the auxiliary model of MISO-NARX type (one output channel);
- the characteristic polynomial of filter, $C_{\text{f}}^{\text{opt}}$;
- initialization of model equations (as previously explained; for example, the average of output data, $\overline{y}$, padded with null values, can be used);
- the input data, $\mathbf{D}_{N}^{\text{u}} = \{u_1[n], u_2[n]\}_{n\in\overline{1,N}}$.

Initialization:
1. Set the first filtered inputs (for example, $u_{\text{f},\{1,2\}}[1] = u_{\{1,2\}}[1]$).
2. Set the first predicted output (for example, $\hat{y}[1] = \overline{y}$).
3. Set the first filtered output (for example, $\hat{y}_{\text{f}}[1] = \overline{y}$).
4. Set the first auxiliary output (for example, $\hat{y}_{\text{f}}^{\text{aux}}[1] = \overline{y}$).
5. Set the first estimated value of white noise (for example, $\hat{e}[1] = 0$).

Algorithm 2 *Cont.*

```
Main loop:
  for $n = 2 : N$
    1. Update the vector $\boldsymbol{\varphi}_{u,\hat{y}}[n]$ coming from the MISO-NARMAX model. The vector includes the latest input data ($u_{\{1,2\}}[n]$, $u_{\{1,2\}}[n-1], \ldots$), the latest predicted output data ($\hat{y}[n-1]$, $\hat{y}[n-2]$, $\ldots$) and corresponding products between such data, according to the model structure.
    2. Call the NARX routine to compute the current useful output, $\hat{y}_{\mathrm{u}}[n]$ (see the NARX$_{\mathrm{useful}}$ block in Figure 3). The routine needs the vector $\boldsymbol{\theta}_{u,y}$ (coming from the MISO-NARMAX model) and the corresponding vector $\boldsymbol{\varphi}_{u,\hat{y}}[n]$, as updated into the previous step.
    3. Add noise to the useful output, such that the current predicted output, $\hat{y}[n]$, be obtained. More specifically, call the filtering routine to return the current estimated value of colored noise to add, $\hat{v}[n]$. The filter is of MA type and employs the C polynomial extracted from the MISO-NARMAX model. Filtering is applied to the latest estimated values of white noise ($\hat{e}[n-1]$, $\hat{e}[n-2], \ldots$).
    4. Update the vector $\boldsymbol{\psi}_{u,\hat{y}}[n]$ coming from the auxiliary MISO-NARX model. The vector includes the latest input data ($u_{\{1,2\}}[n]$, $u_{\{1,2\}}[n-1], \ldots$), the latest predicted output data ($\hat{y}[n-1]$, $\hat{y}[n-2]$, $\ldots$) and corresponding products between such data, according to the model structure.
    5. If $C_{\mathrm{f}}^{\mathrm{opt}}$ is not unit:
      5.1. Filter the inputs. The current values $u_{\mathrm{f},\{1,2\}}[n]$ are obtained by calling again the filtering routine. Unlike in step 2, the filter is of AR type, with $C_{\mathrm{f}}^{\mathrm{opt}}$ as characteristic polynomial. Obviously, the most recent filtered input values are employed ($u_{\mathrm{f},\{1,2\}}[n]$, $u_{\mathrm{f},\{1,2\}}[n-1]$, $\ldots$), together with the current inputs ($u_{\{1,2\}}[n]$).
      5.2. Filter the predicted output. The same filter as in step 5.1 can be used, with the most recent filtered predicted output values ($\hat{y}_{\mathrm{f}}[n-1]$, $\hat{y}_{\mathrm{f}}[n-2], \ldots$), together with the current predicted (unfiltered) output ($\hat{y}[n]$).
      5.3. Filter the product signals into the vector $\boldsymbol{\psi}_{u,\hat{y}}[n]$, like in steps 5.1 and 5.2. (Recall the example given after Equation (36).)
      5.4. Update the vector $\boldsymbol{\psi}_{u,\hat{y}}[n]$. The vector includes the latest filtered input data ($u_{\mathrm{f},\{1,2\}}[n]$, $u_{\mathrm{f},\{1,2\}}[n-1], \ldots$), the latest filtered predicted output data ($\hat{y}_{\mathrm{f}}[n-1]$, $\hat{y}_{\mathrm{f}}[n-2], \ldots$) and the latest filtered corresponding products between such data, according to the model structure (as computed in steps 5.1–5.3).
    6. Call the NARX routine again, to return the current value $\hat{y}_{\mathrm{f}}^{\mathrm{aux}}[n]$ (see the NARX$_{\mathrm{aux}}$ block in Figure 3). This time, its input arguments are the vector $\boldsymbol{\theta}_{u,y}^{\mathrm{aux}}$ (coming from the auxiliary MISO-NARX model) and the corresponding vector $\boldsymbol{\psi}_{u,\hat{y}}[n]$ from steps 4 or 5.
    7. Estimate the current value of white noise: $\hat{e}[n] = \hat{y}_{\mathrm{f}}[n] - \hat{y}_{\mathrm{f}}^{\mathrm{aux}}[n]$.
  end for $n = 2 : N$

Outputs:
  • the simulated (predicted) output data, $\{\hat{y}[n]\}_{n\in\overline{1,N}}$;
  • optionally, the simulated useful data, $\{\hat{y}_{\mathrm{u}}[n]\}_{n\in\overline{1,N}}$.
```

3.3. Numerical Procedure to Simulate MISO-NARXA Models

Another model of interest is 2-MISO-NARXA, for which the simulation procedure is simpler than in case of 2-MISO-NARMAX or 2-MISO-ARMAX models. For this model, Algorithm 3 can be implemented, following Equations (51)–(53).

Both simulation procedures above are solely devoted to serve in the evaluation of the fitness (48), by using the msp technique. Normally, such procedures are not suitable to be included in a closed loop. Schemes based on filters (like the one in Figure 1) are better candidates for this aim since an approximation of the white noise can be generated artificially with the same variance as the estimated one.

Algorithm 3 Simulation Procedure of a MISO-NARXA Model, Based on msp Technique

```
➢ Inputs:
    • the identification model of (pure) MISO-NARX type (one output channel);
    • the auxiliary models of SISO-AR and SISO-ARMA type;
    • initialization of model equations (as previously explained; for example, the average of output data,
      $\overline{y}$, padded with null values, can be used; the first value of output data, $y[1]$, is necessary as well);
    • the input data, $\mathbf{D}_N^{\mathrm{u}} = \{u_1[n], u_2[n]\}_{n\in\overline{1,N}}$.
Initialization:
    1. Set the first predicted output (for example, $\hat{y}[1] = \overline{y}$).
    2. Set the first estimated value of white noise (for example, $\hat{e}[1] = \overline{y} - y[1]$).
        • Note. It is very important that $\hat{e}[1] \neq 0$, otherwise all simulated data concerning the white and
                colored noises will be null. For example, if $y[1] = \overline{y}$, another output data value must be
                selected, say $y[m]$ (with $m > 1$), such that $y[m] \neq \overline{y}$.
    3. Set the first estimated value of colored noise (for example, $\hat{v}[1] = \hat{e}[1]$).
    4. Compute the first useful output (for example, $\hat{y}_{\mathrm{u}}[1] = \hat{y}[1] - \hat{v}[1] = y[1]$).
Main loop:
  for $n = 2 : N$
    1. Update the vector $\boldsymbol{\varphi}_{u,\hat{y}}[n]$ coming from the MISO-NARX model. The vector includes the latest
       input data ($u_{\{1,2\}}[n]$, $u_{\{1,2\}}[n-1], \ldots$), the latest predicted output data ($\hat{y}[n-1]$, $\hat{y}[n-2], \ldots$) and
       corresponding products between such data, according to the model structure.
    2. Call the NARX routine to compute the current useful output, $\hat{y}_{\mathrm{u}}[n]$. The routine needs the vector
       $\boldsymbol{\theta}_{u,y}$ and the corresponding vector $\boldsymbol{\varphi}_{u,\hat{y}}[n]$, as updated into the previous step.
    3. Call the filtering routine to return the current estimated value of white noise, $\hat{e}[n]$. The filter is of
       MA type and employs the characteristic polynomial of auxiliary AR model. See the last Equation
       in (51) and Equation (53). Filtering is applied to the latest estimated values of colored noise
       ($\hat{v}[n-1]$, $\hat{v}[n-2], \ldots$).
    4. Call the filtering routine again to return the current estimated value of colored noise, $\hat{v}[n]$. The
       filter is of ARMA type and employs the characteristic polynomial of auxiliary ARMA model. See
       the second Equation in (51) and Equation (52). Now, the filtering is applied to the latest estimated
       values of colored noise ($\hat{v}[n-1]$, $\hat{v}[n-2], \ldots$), as well as to the latest estimated values of white
       noise ($\hat{e}[n-1]$, $\hat{e}[n-2], \ldots$).
    5. Add the colored noise with the useful output, to obtain the current predicted output, $\hat{y}[n]$,
       according to the first Equation in (51).
  end for $n = 2 : N$
Outputs:
    • the simulated (predicted) output data, $\{\hat{y}[n]\}_{n\in\overline{1,N}}$;
    • optionally, the simulated useful data, $\{\hat{y}_{\mathrm{u}}[n]\}_{n\in\overline{1,N}}$.
```

3.4. CSA Numerical Procedure

Like in the case of all metaheuristics, CSA requires a routine to generate pseudo-random numbers. Such a routine should generate either uniformly distributed sequences of numbers or discrete signals with some prescribed probability distribution. The most employed distribution in applications is the Gaussian one. Several algorithms are available to implement *pseudo-random generators* (prgs). For example, Baker's algorithm can easily be implemented for this purpose [48].

For the numerical procedure in Algorithm 4, two types of prg are necessary: one with a uniform probability distribution (upd-prg) and another one with a prescribed probability distribution (ppd-prg), different from the uniform one. For example, to generate pseudo-random Gaussian signals, the ppd-prg can be used.

Although improvements of Algorithm 4 are possible, this is not one of this article goals. Basically, in MATLAB$^{\mathrm{TM}}$ programming environment, the CSA was implemented following the steps in Algorithm 4.

Algorithm 4 Basic Numerical Procedure of CSA Metaheuristic

```
➢ Inputs:
    • the cost function to optimize, $\mathbf{F}$ (associated with the quality of host nests);
    • the boundaries of search space $\boldsymbol{S}$ (see Equation (55));
    • configuring parameters:
        - population size of the host nests, $P$ (by default, $P = 15$);
        - Lévy's constant, $\lambda$ (by default, $\lambda = 1.5$);
        - scaling factor, $\alpha$ (by default, $\alpha = 0.01$);
        - probability of keeping the host nests into the population $P_a$ (by default, $P_a = 25\%$; recall that
          $1 - P_a$ is the population renewal rate);
        - maximum number of iterations $K$ (by default, $K = 15$).
✚ Initialization:
    1. Use the upd-prg to uniformly spread the first population of host nests over the search space. This
       would be the generation zero of population: $\left\{\mathbf{x}_p^0\right\}_{p\in\overline{1,P}}$.
    2. Evaluate the fitness of each nest in the initial generation: $\left\{\mathbf{F}\left(\mathbf{x}_p^0\right)\right\}_{p\in\overline{1,P}}$.
    3. Determine the best host nest and its performance: $\mathbf{x}_{\text{best}}^0 = \underset{p\in\overline{1,P}}{\text{argopt}}\left\{\mathbf{F}\left(\mathbf{x}_p^0\right)\right\}$, $\mathbf{F}_{\text{best}}^0 = \underset{p\in\overline{1,P}}{\text{opt}}\left\{\mathbf{F}\left(\mathbf{x}_p^0\right)\right\}$.
    4. Compute the variance of variable $u$, namely $\sigma_u^2$, according to Equation (57).
```

Algorithm 4 *Cont.*

```
Main loop:
  for $k = 0 : (K-1)$
  1.  for $p = 1 : P$
      1.1.  Use the ppd-prg to generate current values for variables $u$ and $v$, namely $u_p^k$ and $v_p^k$, respectively. As already mentioned, although both are Gaussian with null mean, the variance of $v$ is equal to unity, while the variance of $u$ is $\sigma_u^2$.
      1.2.  Compute the step size $s$, according to Equation (56), where $u = u_p^k$ and $v = v_p^k$. Thus, $s$ becomes $s_p^k$.
      1.3.  Relocate the current host nest according to Equation (58). Since $\mathbf{x}_{\text{best}}^{k+1}$ would be equal to $\mathbf{x}_{\text{best}}^k$, one can skip this host nest.
      1.4.  Make viable $\mathbf{x}_p^{k+1}$ (excepting for $\mathbf{x}_{\text{best}}^k$, which is already viable).
      1.5.  Evaluate the performance of $\mathbf{x}_p^{k+1}$, i.e., compute $\mathbf{F}(\mathbf{x}_p^{k+1})$.
      1.6.  If $\mathbf{F}(\mathbf{x}_p^{k+1})$ has a better value than $\mathbf{F}(\mathbf{x}_p^k)$, replace the current host nest $\mathbf{x}_p^k$ by the next host nest $\mathbf{x}_p^{k+1}$.
      1.7.  Otherwise, keep unchanged the current host nest and its performance for next generation: $\mathbf{x}_p^{k+1} = \mathbf{x}_p^k$ and $\mathbf{F}(\mathbf{x}_p^{k+1}) = \mathbf{F}(\mathbf{x}_p^k)$.
      1.8.  If $\mathbf{F}\left(\mathbf{x}_p^{k+1}\right)$ has a better value than $\mathbf{F}_{\text{best}}^k$, update the best host nest: $\mathbf{x}_{\text{best}}^{k+1} = \mathbf{x}_p^{k+1}$, $\mathbf{F}_{\text{best}}^{k+1} = \mathbf{F}\left(\mathbf{x}_p^{k+1}\right)$.
      1.9.  Otherwise, keep unchanged the current best solution $\left\{\mathbf{x}_{\text{best}}^k, \mathbf{F}_{\text{best}}^k\right\}$.
      end for $p = 1 : P$
  2.  If $\mathbf{x}_{\text{best}}^k$ and $\mathbf{F}_{\text{best}}^k$ did not change, set $\mathbf{x}_{\text{best}}^{k+1} = \mathbf{x}_{\text{best}}^k$ and $\mathbf{F}_{\text{best}}^{k+1} = \mathbf{F}_{\text{best}}^k$.
  3.  Use the upd-prg to generate the current value $r^{k+1}$ between 0 and 1.
  4.  Use the ppd-prg to generate a matrix of binary values, $\mathbf{B} = [b_{i,p}]_{i \in \overline{1,nx}, p \in \overline{1,P}}$, where 0 occurs with probability $P_a$, while 1 occurs with probability of $1 - P_a$. The matrix has $nx$ rows (the length of decision variable) and $P$ columns (the size of host nests population). The matrix $\mathbf{B}$ matches the matrix $\mathbf{X}^{k+1}$ obtained by gathering all host nests of next population in columns.
  5.  for $p = 1 : P$
      5.1.  If the column $p$ of binary matrix $\mathbf{B}$ is null, keep the host nest $\mathbf{x}_p^{k+1}$ and its performance $\mathbf{F}(\mathbf{x}_p^{k+1})$ unchanged.
      5.2.  Otherwise:
            5.2.1. Use the upd-prg to select two integers $n_p$ and $m_p$ in range $1 : P$, in order to employ the host nests $\mathbf{x}_{n_p}^{k+1}$ and $\mathbf{x}_{m_p}^{k+1}$ into the next step.
            5.2.2. for $i = 1 : nx$
                   5.2.2.1.  If $b_{i,p} = 0$, leave unchanged the component $x_{i,p}^{k+1}$ into the matrix $\mathbf{X}^{k+1}$.
                   5.2.2.2.  Otherwise:
                      5.2.2.2.1. Modify the component $x_{i,p}^{k+1}$ according to Equation (59). The corresponding components of nests $\mathbf{x}_{n_p}^{k+1}$ and $\mathbf{x}_{m_p}^{k+1}$ are involved in this evaluation.
                      5.2.2.2.2. Make viable the new component $x_{i,p}^{k+1}$. end for $i = 1 : nx$
      5.3.  If the host nest $\mathbf{x}_p^{k+1}$ was modified, resulting $\tilde{\mathbf{x}}_p^{k+1}$:
            5.3.1. Compute $\mathbf{F}\left(\tilde{\mathbf{x}}_p^{k+1}\right)$.
            5.3.2. If $\mathbf{F}\left(\tilde{\mathbf{x}}_p^{k+1}\right)$ is better than $\mathbf{F}\left(\mathbf{x}_p^{k+1}\right)$:
                   5.3.2.1.  Renew the host nest: $\mathbf{x}_p^{k+1} \leftarrow \tilde{\mathbf{x}}_p^{k+1}$; $\mathbf{F}\left(\mathbf{x}_p^{k+1}\right) \leftarrow \mathbf{F}\left(\tilde{\mathbf{x}}_p^{k+1}\right)$.
                   5.3.2.2.  If $\mathbf{F}\left(\mathbf{x}_p^{k+1}\right)$ is even better than $\mathbf{F}_{\text{best}}^k$, update the best host nest: $\mathbf{x}_{\text{best}}^{k+1} = \mathbf{x}_p^{k+1}$, $\mathbf{F}_{\text{best}}^{k+1} = \mathbf{F}\left(\mathbf{x}_p^{k+1}\right)$.
                   5.3.2.3.  Otherwise, keep unchanged the best solution $\left\{\mathbf{x}_{\text{best}}^{k+1}, \mathbf{F}_{\text{best}}^{k+1}\right\}$.
            5.3.3  Otherwise, keep $\left\{\mathbf{x}_p^{k+1}, \mathbf{F}\left(\mathbf{x}_p^{k+1}\right)\right\}$ and discard $\left\{\tilde{\mathbf{x}}_p^{k+1}, \mathbf{F}\left(\tilde{\mathbf{x}}_p^{k+1}\right)\right\}$.
      end for $p = 1 : P$
Output:
  • optimal solution: $\{\mathbf{x}_{\text{best}}^K, \mathbf{F}_{\text{best}}^K\}$.
```

4. Simulation Results and Discussion

4.1. Simulation Settings

The programming environment selected to perform the simulations is MATLAB^{TM}. The implemented numerical procedures were designed according to the identification and simulation models described within the previous sections. To test the models, a didactical installation was employed, namely, ASTANK2. An in-depth description of this plant can be found in [46,50]. Therefore, only a succinct presentation of its constructive characteristics is given next. The installation photo is illustrated in Figure 4. As the name suggests and the figure above shows, ASTANK2 consists of two water tanks of approximately 50 cm high, which can be filled with liquid from a lower basin by means of a central pump. The liquid flows back to the basin by a nozzle placed at the bottom of each tank. The leaking flows can be manually controlled by the user through mechanical valves placed on nozzles. The tank on the right side is rectangular, while the tank on the left side has a slanted face, which gives a longitudinal section profile of a trapezoidal form. The slanted face was designed to increase the nonlinearity of the plant dynamics. The tanks can communicate with each other by means of a pipe placed beneath them. A manual valve allows the user to control (and even to completely stop) the communication flow. Two commanded electrovalves are placed on top of the installation and allow automatically controlling the liquid flow. The installation is endowed with liquid flow and level sensors. The two inputs are commanding the electrovalves on top, while the two outputs are given by the liquid levels in the tanks. The installation has an interface panel that can be connected to a computer and communicates with MATLAB^{TM} environment. Thus, i/o data can directly be acquired and stored in MATLAB^{TM} workspace.

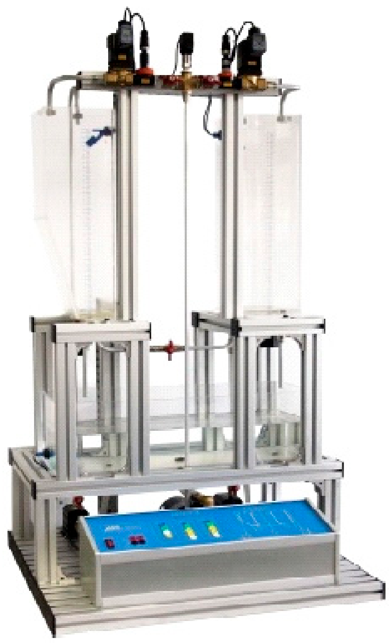

Figure 4. Photo of ASTANK2 installation.

The ultimate goal with this installation is to design controllers for keeping constant liquid levels in the tanks (not necessarily of equal heights), regardless the leaking flows and perturbations (such as the measuring noise or the liquid turbulences).

To reach for this goal, an identification model of ASTANK2 may be needed. Although a controller that does not necessarily need an identification model to be designed (such as PID) can satisfactorily work (see, for example, [50]), it is suitable to aim for more effective controllers and even to optimal, perhaps, nonlinear ones. An identification model is mandatory to be involved in the design of such controllers. Since the installation dynamics is nonlinear, the identification of a multivariable NARMAX model could be suitable.

After few experiments, the static characteristics of ASTANK2 plant were determined. They look like in Figure 5. The horizontal axes correspond to voltages applied to the two electrovalves that determine the filling flows. They are the two inputs of the black box to identify. The electrovalves start to work for voltages of at least 2 V and cannot accept more than 10 V as the maximum operating command without being exposed to failure. The vertical axes correspond to the liquid levels in the tanks and stand for the black box outputs. For each constant couple of voltages, a couple of constant levels was measured. Figure 5 reveals the nonlinear static characteristics, although they are composed of several linear segments, from which the one in the middle is the longest. As one can see, although the voltage u_1 covers the whole operating range (2–10 V), the second voltage is restricted to the range from 6.5 V to 9.5 V. This selection enforced the liquid levels to vary between 8 cm and 15 cm. It has been noticed that this range provides the most accurate i/o data. If the level is too low (below 8 cm), turbulences caused by the leaking of a small amount of liquid strongly affect the measurements. If the level is too high (above 15 cm), the sensors start to introduce systematic errors, and the saturation of voltages at the upper limit of 10 V would be necessary to protect the electrovalves.

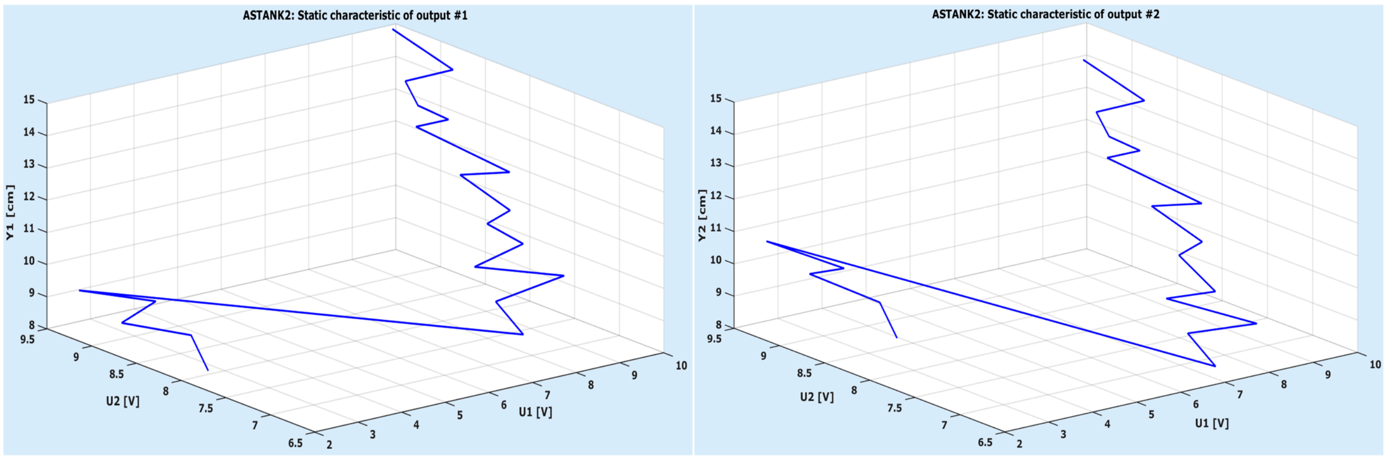

Figure 5. Static characteristics of ASTANK2 plant: for the trapezoidal tank to the left and for the rectangular tank to the right.

Nominal operating points can be selected from the characteristics in Figure 5. For example, one can set the level in the trapezoidal tank to 10.5 cm and the level in the rectangular tank to 9.5 cm. Both levels are placed in the nonlinear zones of the static characteristics.

An important constructive limitation is that the successive voltages applied to electrovalves should not jump with more than 3 V from one value to another. This prevents the electrovalves from overcharging.

Moreover, each voltage should be constant for a duration between 15 s and 60 s such that the flows of liquid passing through the electrovalves have enough time to stabilize and the turbulences to vanish. Consequently, the input signals were generated as batches of steps with pseudo-randomly chosen voltages, as well as durations, with respect to the constructive limitations above. They look like in Figure 6.

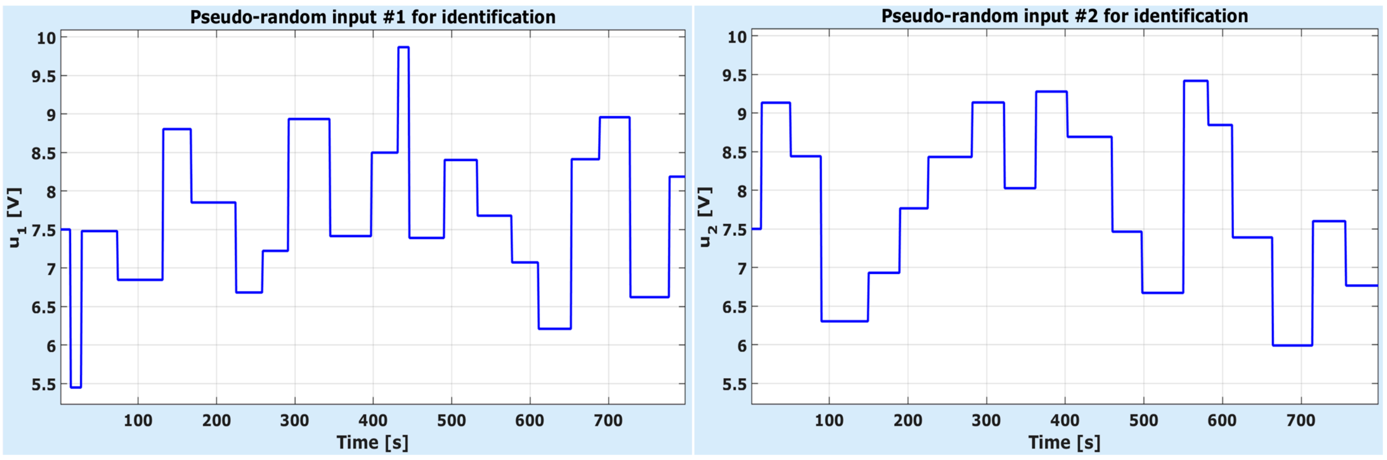

Figure 6. Pseudo-random signals employed to stimulate the ASTANK2 installation, aiming at identification.

The signals above were employed to stimulate the black box to acquire i/o signals for identification, i.e., to build the dataset $\mathbf{D}_{N_{\rm id}}^{\rm id}$. The sampling period was set to 1 s, and the acquisition horizon size is $N_{\rm id} = 798$. Thus, the data acquisition took 13.3 min.

Similar input signals were generated to build the dataset for validation, $\mathbf{D}_{N_{\rm va}}^{\rm va}$, as displayed in Figure 7. The size of the validation horizon is $N_{\rm va} = 798$ (the same as that for the identification horizon). The output acquired data vary around the operating points $\overline{y}_1^{\rm id} = 10.24$ cm and $\overline{y}_2^{\rm id} = 9.37$ cm (for the identification phase) and $\overline{y}_1^{\rm va} = 10.19$ cm and $\overline{y}_2^{\rm va} = 9.29$ cm (for the validation phase). Since the averages of input signals are $\overline{u}_1^{\rm id} = 7.67$ V and $\overline{u}_2^{\rm id} = 7.8$ V (for the identification phase) and $\overline{u}_1^{\rm va} = 7.79$ V and $\overline{u}_2^{\rm va} = 7.66$ V (for the validation phase), one can conclude that both operation points lie into the nonlinear zones of the static characteristics (see Figure 5 again).

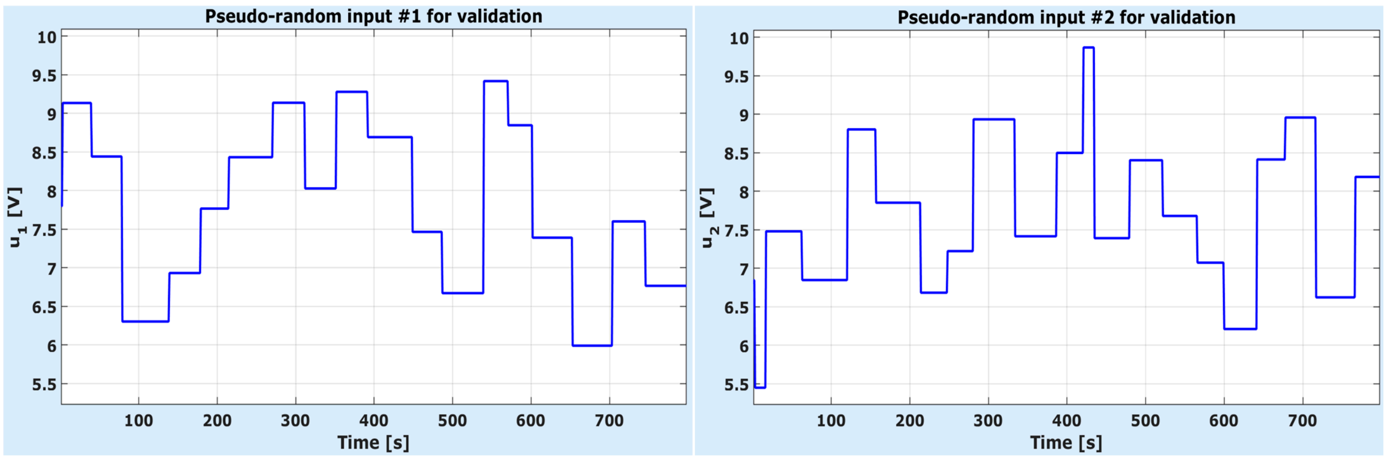

Figure 7. Pseudo-random signals employed to stimulate the ASTANK2 installation, aiming at validation.

After few experiments, one has concluded that the liquid level in one tank does not significantly affect the liquid level in the other tank. Then, one realistically assumed that ASTANK2 consists of two subsystems, each of which having a MISO structure (with two inputs and one output). Thus, the 2-MISO-NARMAX model fits well to the plant. In the same manner, one can consider the 2-MISO-NARXA model and the 2-MISO-ARMAX linear model as two important competitors.

The CSA was set to run hierarchically: one procedure, $\mathrm{CSA}_{\rm sup}$, operates at a superior level, with the 20 major structural indices of vector $\gamma_{\rm M}$, and another procedure, $\mathrm{CSA}_{\rm inf}$, is invoked at an inferior level for any combination of major indices to find the optimal minor indices of vector $\boldsymbol{\gamma}_{\mathrm{m}}$ (with variable length). Note that for the 2-MISO-NARXA model, the number of major indices decreases to 10 since no auxiliary model is necessary. Also, in the case of 2-MISO-ARMAX model, only $\mathrm{CSA}_{\mathrm{inf}}$ suffices, as no major structural indices exist.

The overall search space is delimited by setting the upper bound of the major indices to 3. This choice is motivated by the empirical observation that correlations between signals with delays greater than 2 s have little influence on ASTANK2 dynamics. The polynomials composing the models can have degrees limited to 250 (i.e., approximately $N_{\mathrm{id}}/3$). However, since the polynomials A_1 and A_2 give the poles of all the filters in the models, their degrees were restricted to vary in the set $\overline{0,10}$. In automatic control applications, the models with as small number of poles as possible are preferred; otherwise, the designed controller could become too complex.

The defined search space is huge. For example, only the major indices subspace includes approximately 4^{20} points (close to 10^{12}). Therefore, the search was performed on 10 regular up-to-date computers running in parallel by splitting the domain into 10 disjointed zones. This operation was performed with the help of numbers represented in base 4. More specifically, any combination of major indices can be represented as follows:

$$\gamma_{\mathrm{M},20}4^{19} + \gamma_{\mathrm{M},19}4^{18} + \cdots + \gamma_{\mathrm{M},2}4 + \gamma_{\mathrm{M},1}, \tag{69}$$

where $\boldsymbol{\gamma}_{\mathrm{M}} = \begin{bmatrix} \gamma_{\mathrm{M},1} & \gamma_{\mathrm{M},2} & \cdots & \gamma_{\mathrm{M},19} & \gamma_{\mathrm{M},20} \end{bmatrix}^T$, $\gamma_{\mathrm{M},i} \in \overline{0,3}$, and $\forall\ i \in \overline{1,20}$. (Recall that when some major index $\gamma_{\mathrm{M},i}$ is null, the corresponding component is missing from NARMAX model. If all the major indices are null, the linear ARMAX model is identified.) The interval $\left(0, 4^{20}-1\right]$ was split into 10 disjointed sub-intervals, each of which having a length of approximatively 10^{11}. Then, the search on a computer was performed in a unique sub-interval. The representation (69) together with the sub-interval limits were employed to make viable the positions of the host nests during the search. In the end, the best of the 10 optimal solutions was selected.

The configuration parameters of CSA are set like in Table 1.

Table 1. Configuration parameters for $\mathrm{CSA}_{\mathrm{sup}}$ and $\mathrm{CSA}_{\mathrm{inf}}$.

Parameter	P_{sup}	K_{sup}	$P_{a,\mathrm{sup}}$	α_{sup}	P_{inf}	K_{inf}	$P_{a,\mathrm{inf}}$	α_{inf}
Value	15	15	0.25	0.01	20	20	0.25	0.01

The population size as well as the maximum number of iterations are greater for $\mathrm{CSA}_{\mathrm{inf}}$ than for $\mathrm{CSA}_{\mathrm{sup}}$ because the upper limits of the minor indices are sensibly higher than 3 (the upper limit of the major indices).

When computing the fitness, either Algorithm 2 or Algorithm 3 is employed (based on msp technique), depending on identification model to determine. The fitness values are computed with $\mu_F = \mu = 3.5$.

4.2. *Performance Results*

Three optimal identification models are compared next, namely, 2-MISO-NARMAX, 2-MISO-NARXA, and 2-MISO-ARMAX. As already mentioned, the tandem $\mathrm{CSA}_{\mathrm{sup}}$-$\mathrm{CSA}_{\mathrm{inf}}$ was run for the nonlinear models, while the optimal linear model was found by running $\mathrm{CSA}_{\mathrm{inf}}$ only. The runtime shown in Table 2 decreased for each model type, depending on its complexity, as expected. Because the search space is very large, finding the optimal structural indices took several days.

Table 2. Runtime values to find optimal identification models by means of CSA.

Model	2-MISO-NARMAX	2-MISO-NARXA	2-MISO-ARMAX
Runtime h	355	208	97

The optimal models found have the structural indices in Tables 3 and 4 and the fitness values in Table 5. In the case of NARXA model, the minor indices corresponding to auxiliary AR models are denoted by $na_{1,1}^{\text{aux}}$ and $na_{2,1}^{\text{aux}}$, respectively. Also, the auxiliary ARMA models operate with minor indices $\{na_1^{\text{aux}}, nc_1^{\text{aux}}\}$ and $\{na_2^{\text{aux}}, nc_2^{\text{aux}}\}$, respectively.

Table 3. Major structural indices of optimal identification models.

Major Main	P_1	$M_{1,1}$	$M_{1,2}$	$K_{1,1}$	$K_{1,2}$	P_2	$M_{2,1}$	$M_{2,2}$	$K_{2,1}$	$K_{2,2}$
2-MISO-NARMAX	1	2	1	1	3	–	2	3	3	3
2-MISO-NARXA	1	2	–	–	–	1	2	1	–	–
Major auxiliary	P_1^{aux}	$M_{1,1}^{\text{aux}}$	$M_{1,2}^{\text{aux}}$	$K_{1,1}^{\text{aux}}$	$K_{1,2}^{\text{aux}}$	P_2^{aux}	$M_{2,1}^{\text{aux}}$	$M_{2,2}^{\text{aux}}$	$K_{2,1}^{\text{aux}}$	$K_{2,2}^{\text{aux}}$
2-MISO-NARX	1	3	2	–	–	–	1	–	–	–

Table 4. Minor structural indices of optimal identification models.

Minor Main	na_1	$na_{1,1}$	$nb_{1,1}$	$nb_{1,2}$	$nb_{1,1,1}$	$nb_{1,1,2}$	$nb_{1,1,3}$	$nb_{1,2,1}$	$nb_{1,2,2}$	$nb_{1,2,3}$	$nd_{1,1,1}$	$nd_{1,1,2}$	$nd_{1,1,3}$	$nd_{1,2,1}$	$nd_{1,2,2}$	$nd_{1,2,3}$	nc_1
MISO-NARMAX$_1$	5	26	30	30	30	26	–	26	–	–	10	–	–	4	30	8	8
MISO-NARXA$_1$	10	15	9	24	10	8	–	–	–	–	–	–	–	–	–	–	–
MISO-ARMAX$_1$	10	–	4	28	–	–	–	–	–	–	–	–	–	–	–	–	29
Minor auxiliary	na_1^{aux}	$na_{1,1}^{\text{aux}}$	$nb_{1,1}^{\text{aux}}$	$nb_{1,2}^{\text{aux}}$	$nb_{1,1,1}^{\text{aux}}$	$nb_{1,1,2}^{\text{aux}}$	$nb_{1,1,3}^{\text{aux}}$	$nb_{1,2,1}^{\text{aux}}$	$nb_{1,2,2}^{\text{aux}}$	$nb_{1,2,3}^{\text{aux}}$	$nd_{1,1,1}^{\text{aux}}$	$nd_{1,1,2}^{\text{aux}}$	$nd_{1,1,3}^{\text{aux}}$	$nd_{1,2,1}^{\text{aux}}$	$nd_{1,2,2}^{\text{aux}}$	$nd_{1,2,3}^{\text{aux}}$	nc_1^{aux}
MISO-NARX$_1$	2	23	8	20	13	5	4	12	10	–	–	–	–	–	–	–	–
SISO-AR-ARMA$_1$	100	41	–	–	–	–	–	–	–	–	–	–	–	–	–	–	5
MISO-ARX$_1$	10	–	20	2	–	–	–	–	–	–	–	–	–	–	–	–	–
Minor main	na_2	$na_{2,1}$	$nb_{2,1}$	$nb_{2,2}$	$nb_{2,1,1}$	$nb_{2,1,2}$	$nb_{2,1,3}$	$nb_{2,2,1}$	$nb_{2,2,2}$	$nb_{2,2,3}$	$nd_{2,1,1}$	$nd_{2,1,2}$	$nd_{2,1,3}$	$nd_{2,1,1}$	$nd_{2,1,2}$	$nd_{2,1,3}$	nc_2
MISO-NARMAX$_2$	4	–	27	9	18	1	–	30	1	14	1	30	30	1	1	24	30
MISO-NARXA$_2$	1	1	11	19	18	28	–	4	–	–	–	–	–	–	–	–	–
MISO-ARMAX$_2$	1	–	17	30	–	–	–	–	–	–	–	–	–	–	–	–	30
Minor auxiliary	na_2^{aux}	$na_{2,1}^{\text{aux}}$	$nb_{2,1}^{\text{aux}}$	$nb_{2,2}^{\text{aux}}$	$nb_{2,1,1}^{\text{aux}}$	$nb_{2,1,2}^{\text{aux}}$	$nb_{2,1,3}^{\text{aux}}$	$nb_{2,2,1}^{\text{aux}}$	$nb_{2,2,2}^{\text{aux}}$	$nb_{2,2,3}^{\text{aux}}$	$nd_{2,1,1}^{\text{aux}}$	$nd_{2,1,2}^{\text{aux}}$	$nd_{2,1,3}^{\text{aux}}$	$nd_{2,2,1}^{\text{aux}}$	$nd_{2,2,2}^{\text{aux}}$	$nd_{2,2,3}^{\text{aux}}$	nc_2^{aux}
MISO-NARX$_2$	7	–	27	15	22	–	–	–	–	–	–	–	–	–	–	–	–
SISO-AR-ARMA$_2$	5	32	–	–	–	–	–	–	–	–	–	–	–	–	–	–	64
MISO-ARX$_2$	1	–	1	29	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 5. Fitness values of optimal identification models.

Model	2-MISO-NARMAX	2-MISO-NARXA	2-MISO-ARMAX
Average fitness $\overline{\mathbf{F}}$ [%]	84.07	85.71	81.38
Fitness F [%]	91.21	84.68	75.37

One can notice that no optimal structural index is greater than 30, except for those of the AR and ARMA linear models. Normally, both linear models operate with a higher number of coefficients to achieve the required level of accuracy. Also, the major indices P_1 and P_2 are, at most, unit for both nonlinear models and even for the auxiliary models. This means that for ASTANK2 plant, the autocorrelation of each output can be neglected beyond the unit lag.

Surprisingly, the highest average fitness in Table 5 is provided by the 2-MISO-NARXA model, although its value is close to that of 2-MISO-NARMAX model. Nevertheless, the overall fitness is smaller. Thus, despite the higher average of the four partial fitness values, their variance is quite large. As previously mentioned, as a result of how the overall fitness (48) was defined, when it takes high values, all four channel fitnesses are not only high enough but also grouped as much as possible around their averages. It seems that the 2-MISO-NARXA model exhibits more scattered channel fitness values than the 2-MISO-NARMAX model.

In the sequel, the displayed figures have the following structure: to the left for 2-MISO-NARMAX, in the middle for 2-MISO-NARXA and to the right for 2-MISO-ARMAX. In each figure, output channels are naturally placed: #1 on top and #2 at bottom.

All three optimal models are stable. As already mentioned in Section 2.4, the stability is tested by observing the placement of the roots (poles) coming from polynomials A_1 and A_2 in the complex plane. The pole placement is shown in Figure 8.

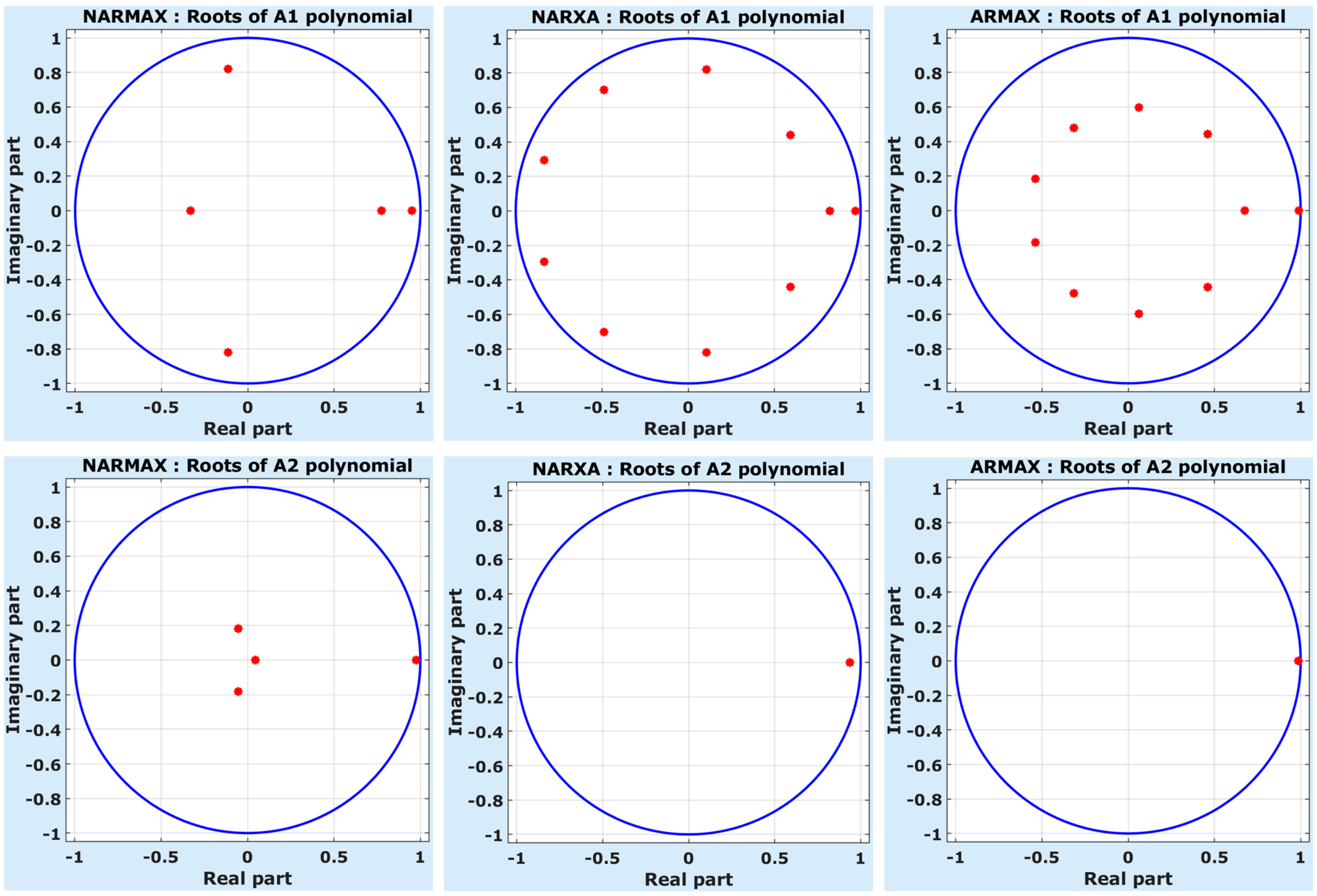

Figure 8. Pole placement of optimal identification models.

In all the cases, there is a pole close to the unit circle, which means the stability margin is quite narrow. The NARMAX and NARXA models are slightly better in this respect, as their critical poles are a bit farther from the unit circle than the critical poles of ARMAX model. What brings superiority to NARMAX model is the reduced number of poles, compared to the other two models: {5,4} versus {10,1} (the same for NARXA and ARMAX). Moreover, the NARMAX poles are almost equally distributed between the two output channels (5 and 4, respectively), whereas NARXA and ARMAX exhibit unbalanced distribution of their poles (10 and 1, respectively). From automatic control point of view, the less poles, the better. Also, in case of ASTANK2 plant, since there is not big difference between the two tanks in terms of configuration and placement, it is more realistic to have almost the same number of poles on both channels, than to allocate sensibly higher number of poles on one channel comparing to the other channel. Thus, NARMAX model seems to be closer to real plant than the other two models, in this respect.

Note that the SchurCohn stability test [1,2] can be employed as well. However, through this test, no polynomial roots are computed, which makes it impossible to visualize the pole placement.

The next figures start with the variations for identification dataset, followed by the variations for validation dataset. Also, a color system was adopted for easy reading. When two variations are drawn, the one in blue stands for the acquired data, while the one in red is associated to simulated signal. The magenta color was allocated to model error signals.

Figure 9 reveals the performance of the useful part for all the models.

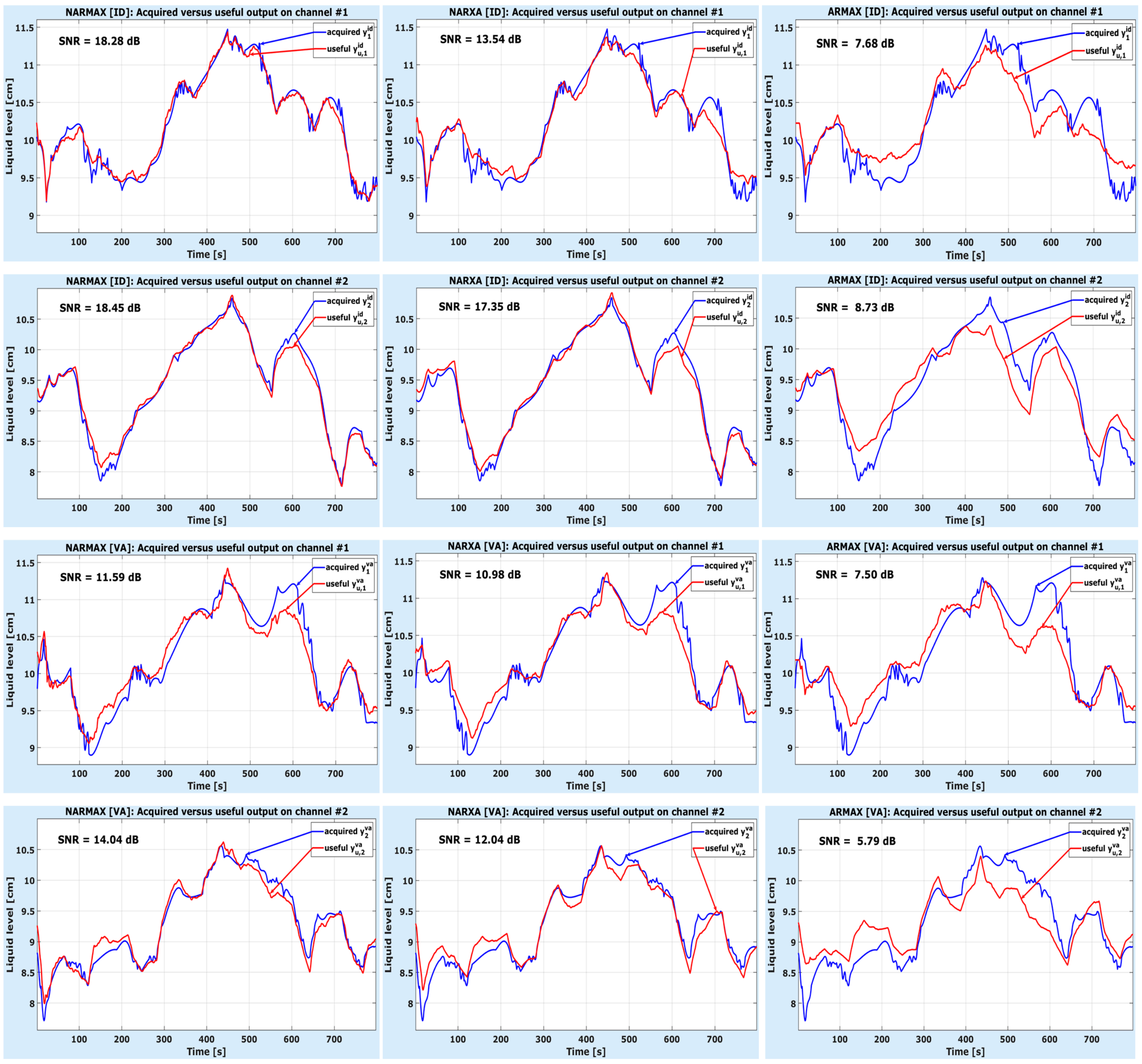

Figure 9. Performance of the useful component for the three optimal identification models.

The second and third columns in Figure 9 clearly show that both 2-MISO-NARXA and 2-MISO-ARMAX models are unable to correctly extract the useful information from the data. One can notice the biases between the acquired output data and the simulated data. By comparison, in the left column, the 2-MISO-NARMAX model exhibits smaller biases. This is not only a subjective opinion, as the displayed SNR values confirm this observation. They are greater for the 2-MISO-NARMAX model than for the other two models in all four windows. Also, for all the models, the SNR values are smaller in the validation phase than in the identification phase, as expected.

The simulated output signals are compared with the acquired output data in Figure 10. The 2-MISO-ARMAX model performs the worst because of its linear nature.

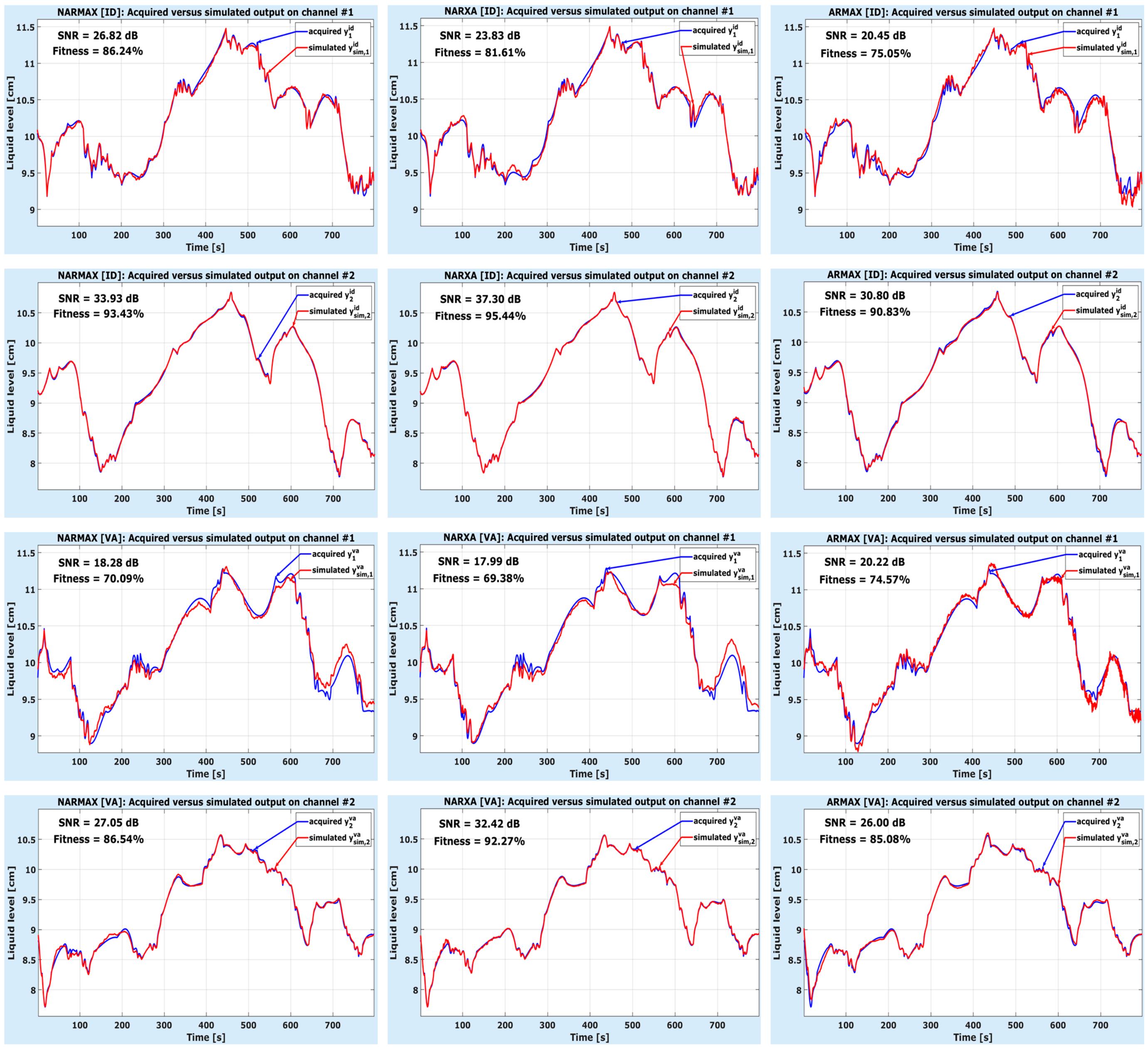

Figure 10. Overall performance of the three optimal identification models.

Nevertheless, although the useful simulated signals are the most biased from the acquired data, the overall simulated outputs resulted closer to those data (see the corresponding SNR values). In fact, the same phenomenon can be observed for the 2-MISO-NARXA model. On the contrary, the 2-MISO-NARMAX model succeeded to provide good, simulated signals in both cases. This means NARMAX was able to perform better separation between useful and noisy information from the acquired data. The other two models did not satisfactorily solve the denoising paradigm, as some of the useful information was recovered by the noise filters. However, the 2-MISO-NARXA is closer to 2-MISO-NARMAX than to 2-MISO-ARMAX, in terms of performance.

In Table 6, the partial fitness values, as well as their absolute difference are given for all models. One can easily notice that the absolute difference is approximately 1.8 times bigger for the 2-MISO-NARX model than for the 2-MISO-NARMAX model. Also, the difference increases more than 3.4 times in case of 2-MISO-ARMAX model. This involved reduction of overall fitness, despite the values of partial fitnesses, which, as Table 6 clearly shows, vary in quite a narrow range, from 74.57% to 78.66%.

Table 6. Partial fitness values of optimal identification models.

Model	2-MISO-NARMAX		2-MISO-NARXA		2-MISO-ARMAX	
Partial fitnesses F_v, F_e **[%]**	75.78	74.74	76.37	74.57	78.66	75.07
Absolute difference $\lvert F_v - F_e \rvert$ **[%]**	1.04		1.80		3.59	

The model errors are depicted in Figure 11. They are decoding the biases between the simulated and acquired signals.

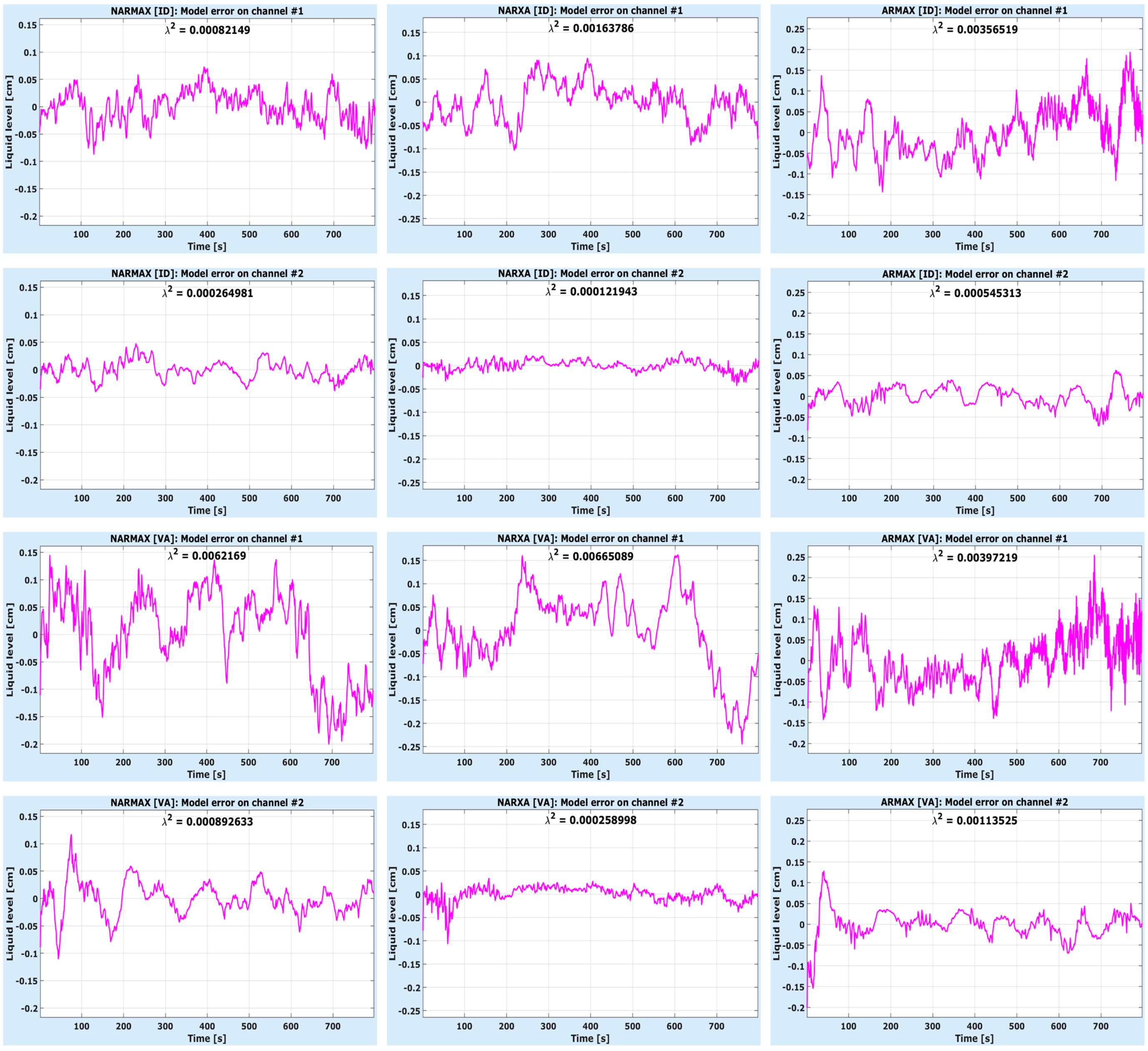

Figure 11. Final output errors of the three optimal identification models.

In each column, the same representation scale was employed. Regardless the model under discussion, no absolute error is greater than 2.5 mm. (Recall that the operating points are above 9.29 cm). Also, the estimated variances of the white noises are specified in each window. The lower, the better. They confirm once more that the final order of the models in terms of performance was 2-MISO-NARMAX (the best), followed by 2-MISO-NARXA, and 2-MISO-ARMAX (the worst).

The accuracy slightly decreases in case of the validation dataset. Moreover, the corresponding errors reveal some auto-correlations, which means they are still colored noises to some extent. Auto-correlation is visible in case of 2-MISO-ARMAX model, even for identification errors. For good identification models, the final errors should have characteristics similar to those of white noises. In fact, the whitening tests (usually employed to validate identification models [1]) were satisfactorily passed only by the 2-MISO-NARMAX model.

The comparison between the acquired and simulated data can be performed not only in the time domain but also in frequency domain. The frequency representations of the inputs are illustrated in Figure 12 (for the identification stage) and Figure 13 (for the validation stage), at a resolution of 2000 frequency samples over the full band.

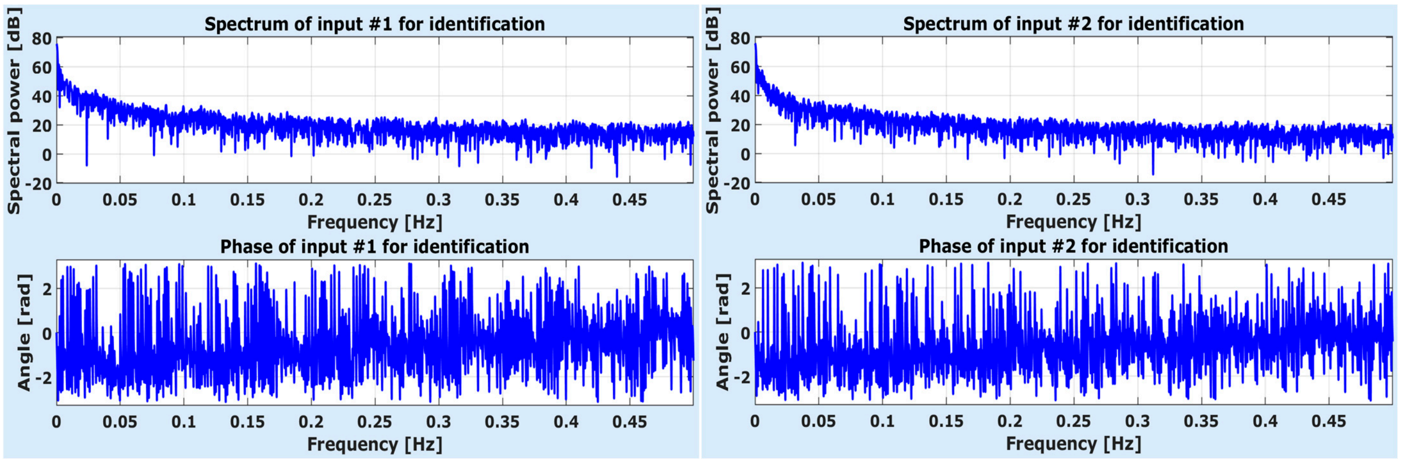

Figure 12. Frequency representations of pseudo-random input signals, aiming at identification.

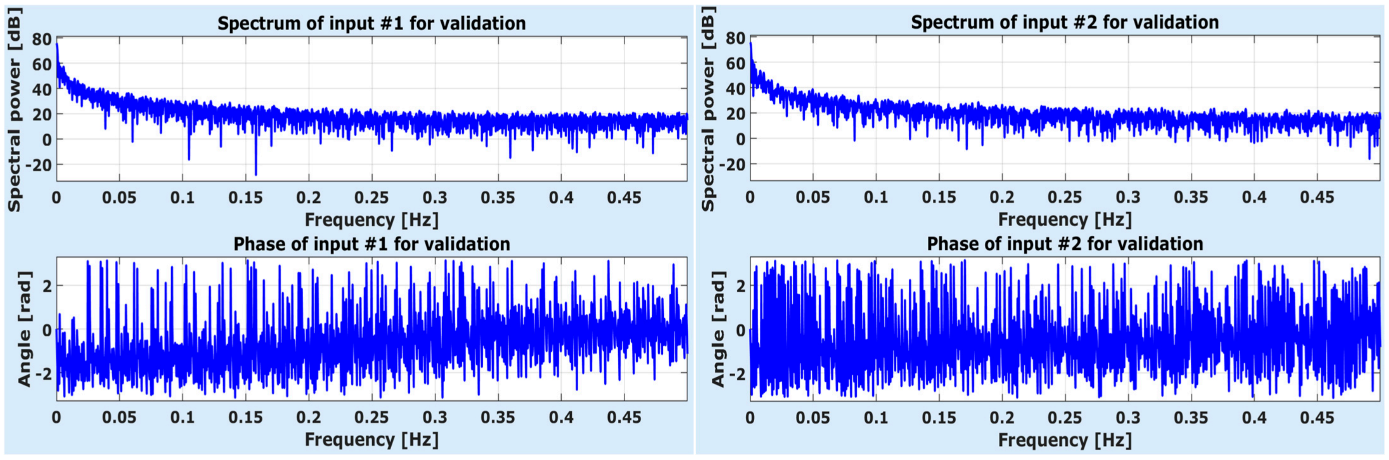

Figure 13. Frequency representations of pseudo-random input signals, aiming at validation.

As one can see, all four signals exhibit almost a constant spectral envelope, which places them nearby white noises. However, because white noise cannot be generated artificially, more energy is allocated in the low-frequency band (due to the limitations of pseudo-random generator). This effect can be helpful, though, since the overwhelming majority of real-life systems behave in frequency like low-pass filters. The ASTANK2 installation is not an exception, as the following figures will prove. Thus, the generated inputs stimulate the plant not only in the low-frequency sub-band (where, very likely, the useful information is encoded) but also towards the medium and high frequency sub-bands to extract information about perturbations corrupting the output responses.

The sampling period of 1 s, which gives the sampling rate of 1 Hz, was selected after carefully inspecting the frequency responses of ASTANK2 installation to the stimulating inputs. Figures 14 and 15 display the frequency responses in both stages (identification and validation, respectively).

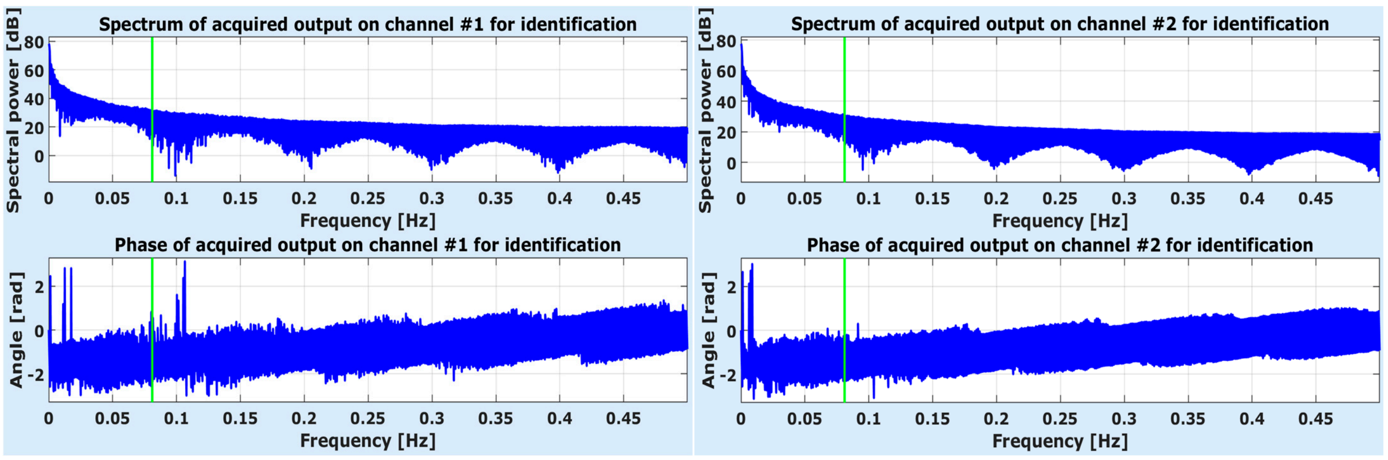

Figure 14. Frequency representations of output signals employed in identification stage.

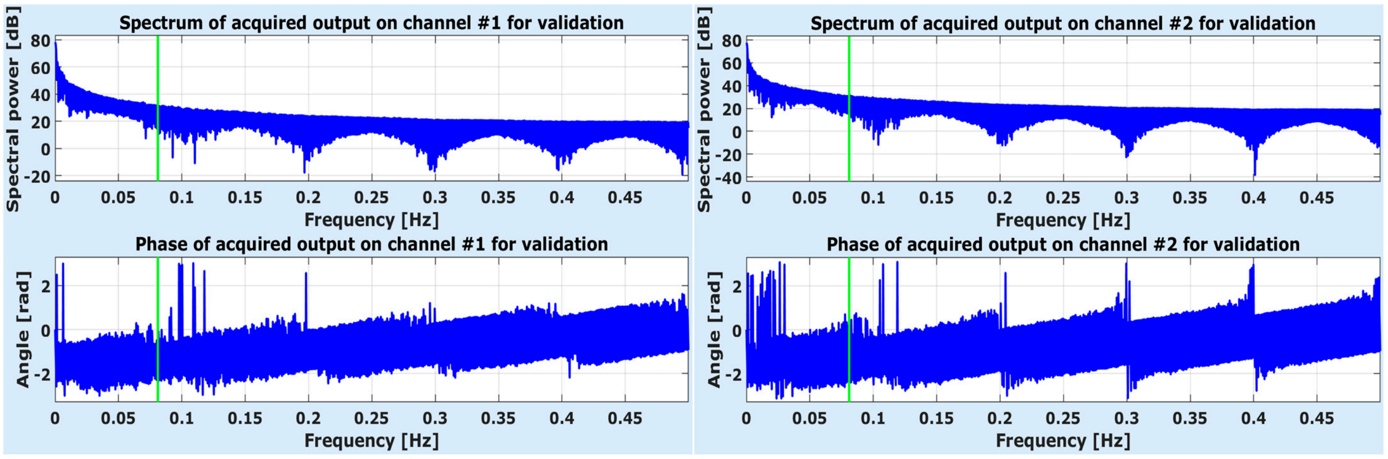

Figure 15. Frequency representations of output signals employed in validation stage.

Indeed, ASTANK2 acts like a low-pass filter, as revealed by the spectral envelopes of all the output acquired data. To estimate its cut-off frequency, an engineering point of view was adopted. Thus, the first frequency at which the spectrum decreases with at least 50 dB from the maximum (obtained at the null frequency) is a good candidate for the cut-off frequency (This means the spectral power decreased more than 350 times compared to the maximum). Since four output datasets are available, the cut-off frequency was estimated by taking the minimum of the four corresponding cut-off frequencies, which returned 0.081 Hz (approximately). A vertical line in green was drawn on all the spectral representations in Figures 14 and 15 to mark the estimated cut-off frequency. Practically, the useful information about ASTANK2 dynamics is roughly encoded into the sub-band to the left of the green line. The noisy information is then encoded into the complementary sub-band, to the right.

According to ShannonNyquist sampling rule [51,52], aliasing is avoided if the sampling rate is at least twice the cut-off frequency. Thus, the minimum value of sampling rate is 0.162 Hz, which gives a maximum sampling period of approximately 6.17 s. Nevertheless, the sampling rate was set to 1 s (more than 6 times smaller) for two main reasons. First, although overall, ASTANK2 rather is a process with slow dynamics, the electrovalves can react quite fast. Setting a sampling period towards the maximum implies the risk of unrealistically reducing the information about perturbations corrupting the acquired output data. Second, decreasing the sampling period under 1 s, involves the insertion of an artificial noise in acquired data, which can cause the numerical instability of algorithms devoted to solving the differential equations in simulation.

Figures 16 and 17 display zooms on pass-band of ASTANK2 installation.

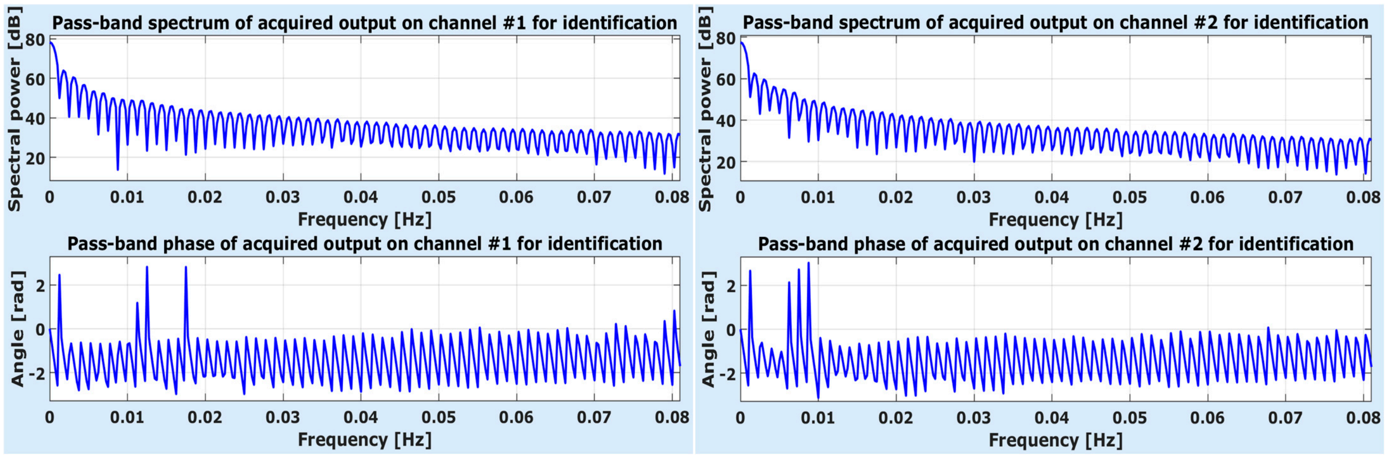

Figure 16. Zoom on pass-band in frequency representations of output signals employed in identification stage.

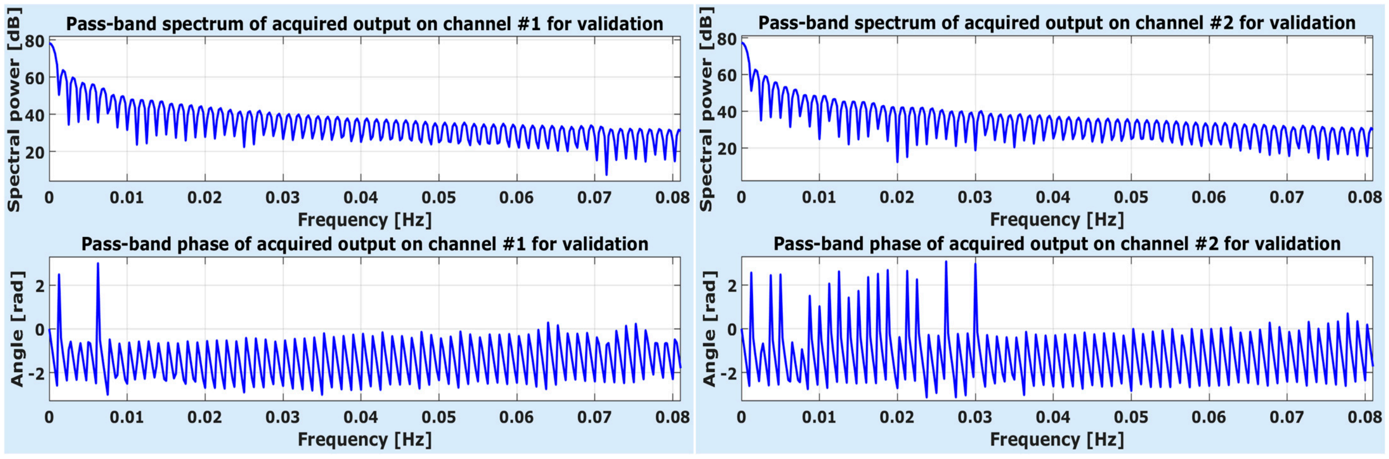

Figure 17. Zoom on pass-band in frequency representations of output signals employed in validation stage.

Although not so easy to notice, differences between the four spectra exist. Much more visible are the differences between the four spectral phases.

The frequency representations of the simulated outputs provided by the three optimal identification models are compared next to the above ones, and the corresponding errors are emphasized. Analyze first the frequency responses of the models over the full-frequency band. The results are displayed in Figure 18 (for the identification phase) and Figure 19 (for the validation phase). In the figures, each column corresponds to an identification model. In every column, the depicted variations are as follows: the spectrum, the phase, the spectral relative error, and the phase relative error first for output channel 1 and then for output channel 2 (i.e., 8 variations per column).

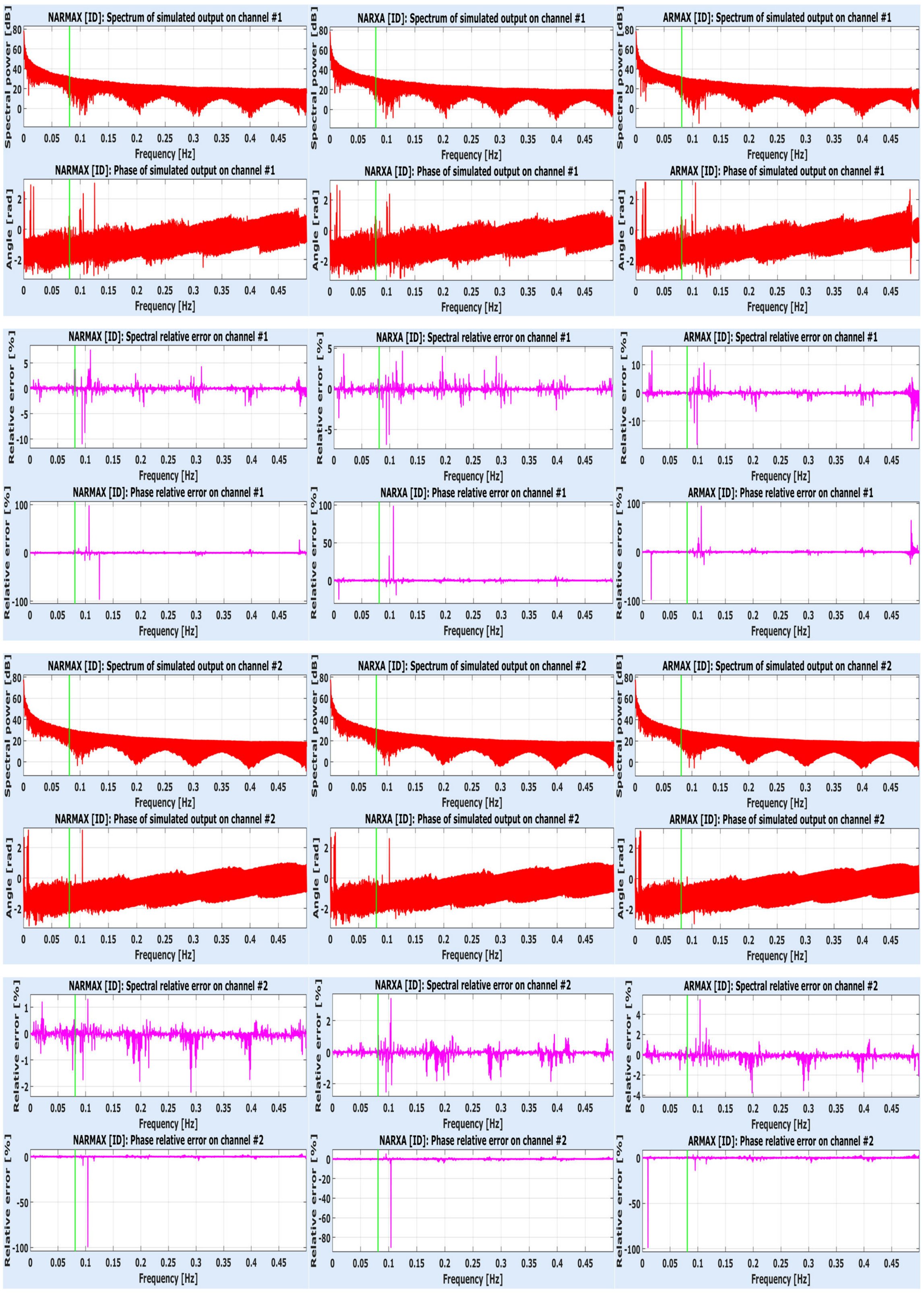

Figure 18. Frequency representations and spectral errors of simulated outputs provided by 2-MISO-NARMAX (left column), 2-MISO-NARXA (middle column), and 2-MISO-ARMAX (right column) for the full frequency band in the identification phase.

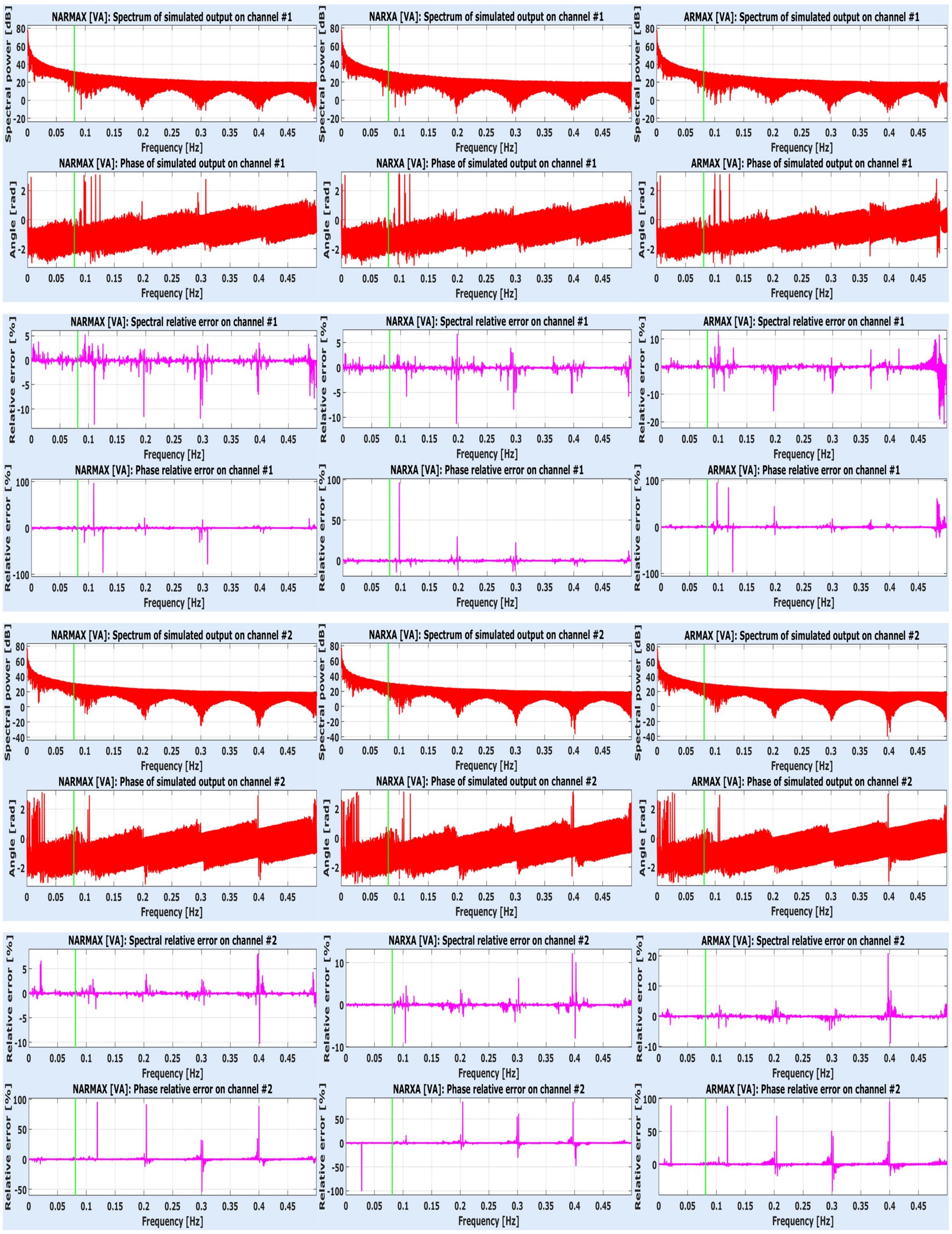

Figure 19. Frequency representations and spectral errors of simulated outputs provided by 2-MISO-NARMAX (left column), 2-MISO-NARXA (middle column), and 2-MISO-ARMAX (right column) for the full frequency band, in the validation phase.

The relative errors were computed by normalization with peak-to-peak of frequency representations resulted from acquired output data. More specifically, in case of spectrum (which is expressed in dB), the absolute error, computed as the difference between the acquired data spectrum and the simulated data spectrum, was divided by the peak-to-peak of the first spectrum and multiplied by 100. Thus, the error is expressed in percents, relatively to the highest variation of acquired data spectrum. In case of phase, the evaluation is simpler, as any angle (measured in radians) is included into $[-\pi, +\pi]$ interval. Thus, the absolute error between phases is divided by 2π and multiplied by 100. The resulted relative error of phase is then expressed in percents with respect to 2π.

Figures 18 and 19 reveal first that, without computing the errors, it is very difficult to outline the differences between all the simulated spectra and the acquired spectra in all four datasets. The same applies to the phases. However, errors exist. At first sight, for all the models, the errors are reasonable, except for the ones corresponding to spikes. In engineering, although relative errors between -5% and +5% are acceptable, it is more suitable that they do not overpass $\pm 3\%$.

Like in case of Figures 14 and 15, the low frequency pass-band of ASTANK2 was separated by a green vertical line. Seemingly, more spikes lie in the high frequency (larger) sub-band than in the pass-band, which means the information about noises is less accurately decoded than the useful information. When looking at the relative error amplitude, one can see that the 2-MISO-ARMAX model performs the worst, while the nonlinear models exhibit similar performance. Moreover, the 2-MISO-ARMAX model is less accurate on channel 1 than on channel 2, as expected (Recall that the trapezoidal tank was associated to channel 1, which introduces more nonlinearity than the rectangular tank of channel 2).

For all the models, the errors are slightly higher in the validation phase than in the identification phase, as expected as well. The spectra are better approximated than the phases, where some spikes can reach 100%. Nevertheless, some of 100% spikes are not real, since angles $\alpha - \pi$ and $\alpha + \pi$ produce the same spectral phase, in fact.

To make the difference between the nonlinear models, zooms on pass-band are added in Figure 20 (for the identification phase) and Figure 21 (for the validation phase). The overwhelming majority of the relative errors coming from the 2-MISO-NARMAX model remain in the $\pm 3\%$ band, and most spikes do not overpass $\pm 5\%$. Accidentally, there are few spikes going beyond $\pm 6\%$ in validation phase. Thus, this model can be considered as an acceptable one. The 2-MISO-NARXA model verifies almost all the same properties, except for the relative errors of spectral phase, which have some big spikes. In fact, this model even outperforms 2-MISO-NARMAX on channel 2 in identification and channel 1 in validation. However, this performance rather is due to the ARMA-AR inner model of noise than to the pure NARX model, as the useful information is not as accurately decoded as in the case of 2-MISO-NARMAX model (see Figure 9 again). Still, the 2-MISO-NARXA is a model to consider next the 2-MISO-NARMAX one. Anyway, the 2-MISO-ARMAX model is less accurate. This is also proven by the variances of the errors shown in Table 7.

Table 7. Variances of spectral and phase relative errors.

Model	$\sigma_{s,1}^{id}$	$\sigma_{p,1}^{id}$	$\sigma_{s,2}^{id}$	$\sigma_{p,2}^{id}$	$\sigma_{s,1}^{va}$	$\sigma_{p,1}^{va}$	$\sigma_{s,2}^{va}$	$\sigma_{p,2}^{va}$
2-MISO-NARMAX	0.3908	0.5786	0.1827	0.2716	0.5286	0.9731	0.5372	0.7487
2-MISO-NARXA	0.4181	1.4526	0.0984	0.1495	0.4050	0.6320	0.0848	5.5355
2-MISO-ARMAX	1.0524	5.4345	0.2276	5.4742	0.4378	0.8409	0.3151	4.9893

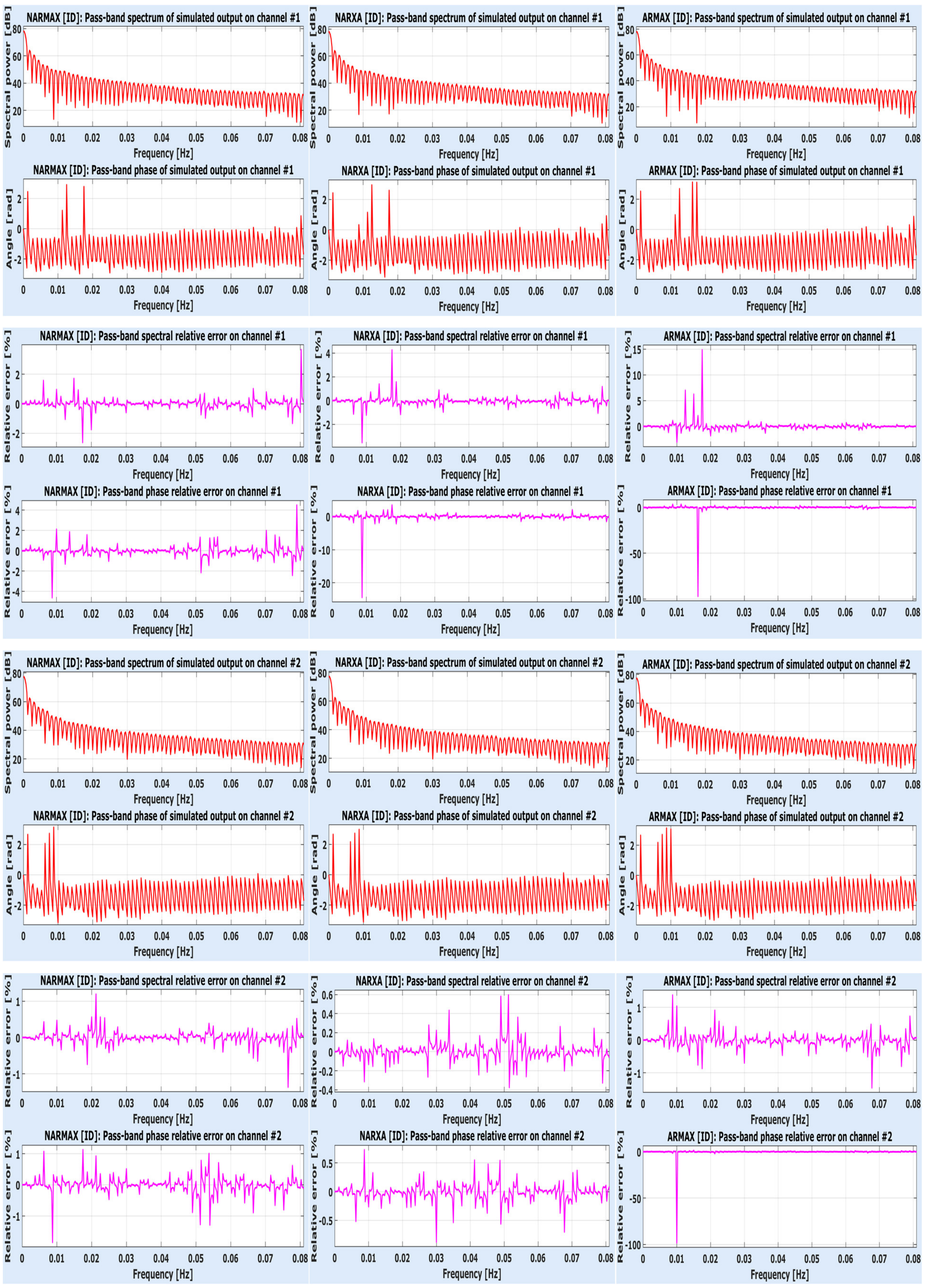

Figure 20. Frequency representations and spectral errors of simulated outputs provided by 2-MISO-NARMAX (left column), 2-MISO-NARXA (middle column), and 2-MISO-ARMAX (right column) for the low frequency sub-band in the identification phase.

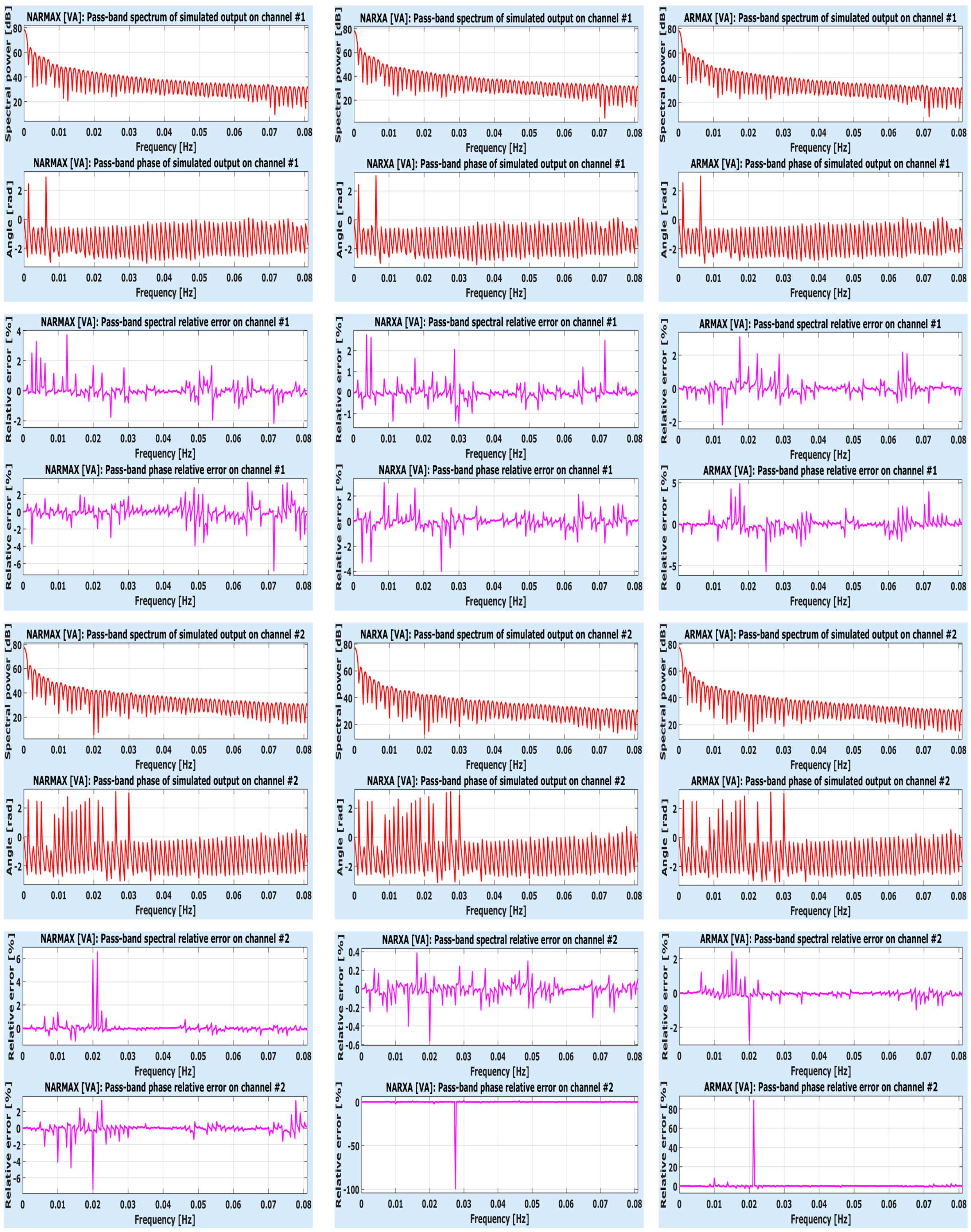

Figure 21. Frequency representations and spectral errors of simulated outputs provided by 2-MISO-NARMAX (left column), 2-MISO-NARXA (middle column), and 2-MISO-ARMAX (right column) for the low frequency sub-band in the validation phase.

In the table above, variances are naturally denoted. For example, $\sigma_{s,1}^{id}$ is the variance of spectral relative error on channel 1, in identification phase, while $\sigma_{p,1}^{id}$ is the variance of twin error, computed on spectral phase. One can easily see that all variances are higher for the 2-MISO-ARMAX model than for the other two nonlinear models. However, the nonlinear models are close to each-other. What pushed the 2-MISO-NARXA model on second place are the high variances of spectral phases errors $\sigma_{p,1}^{id}$ and especially $\sigma_{p,2}^{va}$ (because of a 100% spike, though), beside the incapacity to correctly decode the useful information.

A final remark regard the two tanks of liquid in ASTANK2 installation. Recall that the first output corresponds to the level in the trapezoidal tank, which accentuates the nonlinear behavior, whereas second output measures the level in the rectangular tank, which is closer to the linear behavior. As the results above prove, the accuracy of the first tank model is lower than the accuracy of the second tank models. Evidently, in case of ARMAX model, this result is normal because of its linear nature. Nevertheless, in case of the nonlinear models, the first output should have been more accurately identified. Since the results did not match such expectations, probably there are other nonlinear models that perform better than the ones from NARMAX class in case of ASTANK2 plant.

5. Concluding Remarks

The aim of this article was to determine a nonlinear identification model from i/o-acquired data by using a combination of techniques, including the MS-EOLS identification method, the msp technique for simulation, and the CSA as a selected metaheuristic for finding the optimal structural indices. Thanks to the msp technique, the model can be directly integrated into a closed loop structure to design a good, suitably optimal controller. Moreover, one can say that the proposed identification methodology by using the msp technique can represent a general (yet practical) approach for the identification of various nonlinear multivariable systems as a prior step for control applications. Subsequently, a comparison between two nonlinear models (namely, 2-MISO-NARMAX and 2-MISO-NARXA) and a linear model (2–MISO–ARMAX) was performed in a setting that employs the ASTANK2 installation. Since both the static characteristic and the dynamic behavior of ASTANK2 are nonlinear, the nonlinear models resulted in better performances than those of the linear model. Moreover, the more complex model, of NARMAX type, resulted to perform better not only than the linear model of ARMAX type but also than the other nonlinear model, of NARX type.

The last remark of the previous section suggests that the investigation of other nonlinear models (especially from Wiener and WienerHammerstein classes) should be one of the future research directions. Another direction is to exploit other metaheuristics (such as PSO), to seek for better models that can be identified. And, of course, an interesting and challenging direction is to perform a sound, in-depth analysis concerning the stability of NARMAX models.

Author Contributions: Conceptualization, D.S., J.C., A.-C.V. and V.V.; methodology, D.S., J.C., A.-C.V. and V.V.; software, D.S., J.C., A.-C.V. and V.V.; validation, D.S., J.C., A.-C.V. and V.V.; formal analysis, D.S., J.C., A.-C.V. and V.V.; investigation, D.S., J.C., A.-C.V. and V.V.; resources, D.S., J.C., A.-C.V. and V.V.; data curation, D.S., J.C., A.-C.V. and V.V.; writing—original draft, D.S., J.C., A.-C.V. and V.V.; writing—review and editing, D.S., J.C., A.-C.V. and V.V.; visualization, D.S., J.C., A.-C.V. and V.V.; supervision, D.S., J.C., A.-C.V. and V.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data employed while testing the numerical algorithms can be sent to readers on request.

Conflicts of Interest: The authors declare no conflicts of interest.

Acronyms

ARMAX	Auto-Regressive Moving Average with eXogenous input model
ARMA	Auto-Regressive Moving Average model
ARX	Auto-Regressive with eXogenous input model
CSA	Cuckoo Search Algorithm
ELS	Extended Least Squares
ERR	Error Reduction Ratio
MA	Moving Average model
MISO	Multiple-Inputs-Single-Output (system)
MS-EOLS	Multi-Step Extended OLS
NARMAX	Nonlinear ARMAX model
NARMA	Nonlinear ARMA model
NARX	Nonlinear ARX model
NARXA	NARX model combined with ARMA and AR models
NSI	Nonlinear SI
OLS	Orthogonal Least Squares
PSO	Particle Swarm Optimization
SI	System Identification
SISO	Single-Input-Single-Output (system)
SNR	Signal-to-Noise Ratio(s)
dB	decibels (logarithmic scale)
i/o	input/output (data, behavior, etc.)
id	identification
msa	multi-step ahead (prediction)
osa	one-step ahead (prediction)
ppd-prg	prescribed probability distribution prg
prg	pseudo-random generator
upd-prg	uniform probability distribution prg
va	validation

References

1. Söderström, T.; Stoica, P. *System Identification*; Prentice Hall International Ltd.: London, UK, 1989; ISBN 0-I 3-881236-5.
2. Ljung, L. *System Identification—Theory for the User*, 2nd ed.; Prentice Hall, Upper Saddle River: Bergen, NJ, USA, 1999; ISBN 978-0-13-656695-3.
3. Alessandrini, M.; Falaschetti, L.; Turchetti, C. Nonlinear Dynamic System Identification in the Spectral Domain Using Particle-Bernstein Polynomials. *Electronics* **2022**, *11*, 3100. [CrossRef]
4. Pal, P.S.; Kar, R.; Ghoshal, S.P. Identification of NARMAX Hammerstein Models with Performance Assessment Using Brain Storm Optimization Algorithm. *Int. J. Adapt. Control Signal Process.* **2016**, *30*, 1043–1070. [CrossRef]
5. Rahrooh, A.; Shepard, S. Identification of Nonlinear Systems Using NARMAX Model. *Nonlinear Anal. Theory Methods Appl.* **2009**, *71*, E1198–E1202. [CrossRef]
6. Billings, S.A.; Leontaritis, I.J. Identification of Nonlinear Systems Using Parameter Estimation Techniques. In Proceedings of the IEEE Conference Proceedings on Control and Its Applications, University of Warwick, Coventry, UK, 23–25 March 1981; pp. 183–187.
7. Haber, R.; Keviczky, L. *Nonlinear System Identification—Input-Output Modeling Approach*; Kluwer Academic Publishers: Dordrecht, The Netherlands, 1999; ISBN 978-0-7923-5858-9.
8. Billings, S.A. *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*; John Wiley & Sons: Hoboken, NJ, USA, 2013; ISBN 978-1-119-94359-4.
9. Cao, Y.; Wang, Z.J.; Wang, W.D. Modeling of Weld Penetration Control System in GMAW-P Using NARMAX Methods. *J. Manuf. Process.* **2021**, *65*, 512–524. [CrossRef]
10. Aggoune, L.; Chetouani, Y. Modeling of a distillation column based on NARMAX and Hammerstein models. *Int. J. Model. Simul. Sci. Comput.* **2017**, *8*, 1750034. [CrossRef]
11. Guan, X.C.; Zhao, D.Y.; Zhu, Q.M. NARMAX Modelling and U-Model Control Design for Continuous Stirred Tank Reactor (CSTR). In Proceedings of the 35th Chinese Control Conference (CCC), Chengdu, China, 27–29 July 2016; pp. 1964–1969.
12. Garces, H.O.; Rojas, A.J.; Arias, L.E. Selection of nonlinear structures for total radiation modeling. In Proceedings of the IEEE International Conference on Automatica (ICA-ACCA), Curico, Chile, 19–21 October 2016. [CrossRef]

13. Fagundes, L.P.; Avelar, H.J.; Vincenzi, F. Improvements in Identification of Fuel Cell Temperature Model. In Proceedings of the 13th IEEE Brazilian Power Electronics Conference/1st Southern Power Electronics Conference (COBEP/SPEC), Fortaleza, Brazil, 29 November–2 December 2015. [CrossRef]
14. Deng, Z.H.; Chen, Q.H.; Fu, Z.C. Data Driven NARMAX Modeling for PEMFC Air Compressor. *Int. J. Hydrogen Energy* **2020**, *45*, 20321–20328. [CrossRef]
15. Jazayeri, P.; Rosehart, W.; Westwick, D.T. A Multistage Algorithm for Identification of Nonlinear Aggregate Power System Loads. *IEEE Trans. Power Syst.* **2007**, *22*, 1072–1079. [CrossRef]
16. Fernandes, D.L.; Lopes, F.R.; Ayala, H.V.H. System Identification of an Elastomeric Series Elastic Actuator Using Black-Box Models. In Proceedings of the 31st Mediterranean Conference on Control and Automation (MED), Limassol, Cyprus, 26–29 June 2023; pp. 569–574.
17. Pradhan, S.K.; Subudhi, B. NARMAX Modeling of a Two-Link Flexible Robot. In Proceedings of the Annual IEEE India Conference (INDICON): Engineering Sustainable Solutions, Hyderabad, India, 16–18 December 2011. [CrossRef]
18. Iglesias, R.; Kyriacou, T.; Billings, S. Route Training in Mobile Robotics through System Identification. In Proceedings of the Conference of the World-Academy-of-Science-Engineering-and-Technology, Barcelona, Spain, 22–24 October 2006; Volume 15, pp. 181–186.
19. Kelley, J.; Hagan, M.T. Comparison of Neural Network NARX and NARMAX Models for Multi-Step Prediction Using Simulated and Experimental Data. *Expert Syst. Appl.* **2024**, *237*, 121437. [CrossRef]
20. Oruc, O.; Cook, A.; Mu, B.X. Nonlinear System Identification for Heading and Pitch Control of a Tethered Uncrewed Underwater Vehicle in Changing and Uncertain Environments. In Proceedings of the OCEANS Hampton Roads Conference, Hampton Roads, VA, USA, 17–20 October 2022. [CrossRef]
21. Chiu, H.L. Identification Approach for Nonlinear MIMO Dynamics of Closed-Loop Active Magnetic Bearing System. *Appl. Sci.* **2022**, *12*, 8556. [CrossRef]
22. Barbosa, M.P.S.; da Costa, D.P.; Ayala, H.V.H. Evaluation of Nonlinear System Identification to Model Piezoacoustic Transmission. In Proceedings of the 21st IFAC World Congress on Automatic Control—Meeting Societal Challenges, Berlin, Germany, 12–17 July 2020; Volume 53, pp. 8802–8807.
23. Mohamad, M.S.A.; Yassin, I.M.; Adnan, R. Comparison Between PSO and OLS for NARX Parameter Estimation of a DC Motor. In Proceedings of the IEEE Symposium on Industrial Electronics and Applications (ISIEA), Kuching, China, 22–25 September 2013; pp. 27–32.
24. Faieghi, M.R.; Azimi, S.M. Design an Optimized PID Controller for Brushless DC Motor by Using PSO and Based on NARMAX Identified Model with ANFIS. In Proceedings of the 12th International Conference on Computer Modelling and Simulation (UKSim), Cambridge, UK, 24–26 March 2010; pp. 16–21.
25. Obeid, S.; Ahmadi, G.; Jha, R. NARMAX Identification Based Closed-Loop Control of Flow Separation over NACA 0015. *Airfoil Fluids* **2020**, *5*, 100. [CrossRef]
26. Hall, R.J.; Wei, H.L.; Hanna, E. Complex Systems Modelling of UK Winter Wheat Yield. *Comput. Electron. Agric.* **2023**, *209*, 107855. [CrossRef]
27. Krishnanathan, K.; Anderson, S.R.; Kadirkamanathan, V. A Data-Driven Framework for Identifying Nonlinear Dynamic Models of Genetic Parts. *ACS Synth. Biol.* **2012**, *1*, 375–384. [CrossRef] [PubMed]
28. Gu, Y.L.; Yang, Y.; Wei, H.L. Nonlinear Modeling of Cortical Responses to Mechanical Wrist Perturbations Using the NARMAX Method. *IEEE Trans. Biomed. Eng.* **2021**, *68*, 948–958. [CrossRef] [PubMed]
29. Piskulak, P.; Lewenstein, K. Modeling of Sleep Disordered Breathing Using NARMAX Methodology. In Proceedings of the International Mechatronics Conference—Recent Advances towards Industry 4.0, Warsaw, Poland, 16–19 September 2019; pp. 438–444.
30. Sun, Y.M.; Simpson, I.; Wei, H.L.; Hanna, E. Probabilistic Seasonal Forecasts of North Atlantic Atmospheric Circulation Using Complex Systems Modelling and Comparison with Dynamical Models. *Meteorol. Appl.* **2024**, *31*, e2178. [CrossRef]
31. Udaichi, K.; Nagappan, R.C.; Bhukya, S.N. Large-Scale System Identification Using Self-Adaptive Penguin Search Algorithm. *IET Control Theory Appl.* **2023**, *17*, 2292–2303. [CrossRef]
32. Cheng, C.M.; Peng, Z.K.; Meng, G. Volterra-Series-Based Nonlinear System Modeling and Its Engineering Applications: A State-of-the-Art Review. *Mech. Syst. Signal Process.* **2017**, *87*, 340–364. [CrossRef]
33. Culita, J.; Stefanoiu, D.; Nica, A.M. Nonlinear Identification for Control by Using HARMAX Models. In Proceedings of the 9th International Conference on Control, Decision and Information Technologies (CODIT 2023), Rome, Italy, 3–6 July 2023; pp. 1868–1875.
34. Worden, K.; Becker, W.E.; Cross, E.J. On the Confidence Bounds of Gaussian Process NARX Models and Their Higher-Order Frequency Response Functions. *Mech. Syst. Signal Process.* **2018**, *104*, 188–223. [CrossRef]
35. Li, P.; Wei, H.L.; Boynton, R. Nonlinear Model Identification from Multiple Data Sets Using an Orthogonal Forward Search Algorithm. *J. Comput. Nonlinear Dyn.* **2013**, *8*, 041001. [CrossRef]
36. Zakaria, M.Z.; Mansor, Z.; Ahmad, R. NARMAX Model Identification Using Multi-Objective Optimization Differential Evolution. *Int. J. Integr. Eng.* **2018**, *10*, 188–203. [CrossRef]
37. Yan, J.Y.; Deller, J.R. NARMAX Model Identification Using a Set-Theoretic Evolutionary Approach. *Signal Process.* **2016**, *123*, 30–41. [CrossRef]

38. Zhu, Q.M.; Wang, Y.J.; Billings, S.A. Review of Rational (Total) Nonlinear Dynamic System Modelling, Identification and Control. *Int. J. Syst. Sci.* **2015**, *46*, 2122–2133. [CrossRef]
39. Palanthandalam-Madapusi, H.J.; Edamana, B.; Ridley, A.J. NARMAX Identification for Space Weather Prediction Using Polynomial Radial Basis Functions. In Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans, LA, USA, 12–14 December 2007; pp. 4774–4779.
40. Cordova, J.; Yu, W. Two Types of Haar Wavelet Neural Networks for Nonlinear System Identification. *Neural Process. Lett.* **2012**, *35*, 283–300. [CrossRef]
41. Xia, X.Z.; Han, L.J.; Cheng, L. A Compliant Elbow Exoskeleton with an SEA at Interaction Port. In Proceedings of the 30th International Conference on Neural Information Processing (ICONIP) of the Asia-Pacific-Neural-Network-Society (APNNS), Changsha, China, 20–23 November 2023; pp. 146–157.
42. Bernat, J.; Kolota, J.; Superczynska, P. NARMAX Approach for the Identification of a Dielectric Electroactive Polymer Actuator. *Int. J. Control Autom. Syst.* **2023**, *21*, 3080–3090. [CrossRef]
43. Watanabe, R.N.; Kohn, A.F. System Identification of a Motor Unit Pool Using a Realistic Neuromusculoskeletal Model. In Proceedings of the 5th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob), Sao Paulo, Brazil, 12–15 August 2014; pp. 610–615.
44. Falsone, A.; Piroddi, L.; Prandini, M. A Novel Randomized Approach to Nonlinear System Identification. In Proceedings of the 53rd IEEE Annual Conference on Decision and Control (CDC), Los Angeles, CA, USA, 15–17 December 2014; pp. 6516–6521.
45. Yuan, X.L.; Bai, Y. Stochastic Nonlinear System Identification Using Multi-objective Multi-population Parallel Genetic Programming. In Proceedings of the 21st Chinese Control and Decision Conference, Guilin, China, 17–19 June 2009; pp. 1148–1153.
46. Culita, J.; Stefanoiu, D.; Dumitrascu, A. ASTANK2: Analytical Modeling and Simulation. In Proceedings of the 20th International Conference on Control Systems and Computer Science (CSCS), Bucharest, Romania, 27–29 May 2015; pp. 141–148.
47. Yang, X.S.; Deb, S. Cuckoo Search via Lévy Flights. In Proceedings of the World Congress on Nature & Biologically Inspired Computing, Coimbatore, India, 9–11 December 2009; pp. 210–214.
48. Stefanoiu, D.; Borne, P.; Popescu, D.; Filip, F.G.; El Kamel, A. *Optimization in Engineering Sciences—Metaheuristics, Stochastic Methods and Decision Support*; John Wiley & Sons & ISTE Press: London, UK, 2014; ISBN 978-1-84821-498-9.
49. Mantegna, R.N. Fast, Accurate Algorithm for Numerical Simulation of Lévy Stable Stochastic Processes. *Phys. Rev. E* **1994**, *49*, 4677–4683. [CrossRef]
50. Stefanoiu, D.; Culita, J. Optimal Identification and Metaheuristic PID Control of a Two-Tank System. *Electronics* **2021**, *10*, 1101. [CrossRef]
51. Oppenheim, A.V.; Schafer, R. *Digital Signal Processing*; Prentice Hall International Ltd.: London, UK, 1985; ISBN 0-13-214107-8.
52. Proakis, J.G.; Manolakis, D.G. *Digital Signal Processing. Principles, Algorithms and Applications*; Prentice Hall Inc.: Upper Saddle River, NJ, USA, 1996; ISBN 0-13-394338-9.