Nonlinear Dyn (2020) 99:1709–1761
https://doi.org/10.1007/s11071-019-05430-7

REVIEW

A review on computational intelligence for identification of nonlinear dynamical systems

Giuseppe Quaranta · Walter Lacarbonara · Sami F. Masri

Received: 29 November 2019 / Accepted: 11 December 2019 / Published online: 1 January 2020

Abstract This work aims to provide a broad overview of computational techniques belonging to the area of artificial intelligence tailored for identification of nonlinear dynamical systems. Both parametric and nonparametric identification problems are considered. The examined computational intelligence techniques for parametric identification deal with genetic algorithm, particle swarm optimization, and differential evolution. Special attention is paid to the parameters estimation for a rich class of nonlinear dynamical models, including the Bouc–Wen model, chaotic systems, the Jiles–Atherton model, the LuGre model, the Prandtl–Ishlinskii model, the Preisach model, and the Wiener–Hammerstein model. On the other hand, genetic programming and artificial neural networks are discussed for nonparametric identification applications. Once the identification problem is formulated, a detailed illustration of the considered computational intelligence techniques is provided, together with a comprehensive examination of relevant applications in the fields of structural mechanics and engineering. Possible directions for future research are also addressed.

G. Quaranta (✉) · W. Lacarbonara
Department of Structural and Geotechnical Engineering,
Sapienza University of Rome, Via Eudossiana 18,
00184 Rome, Italy
e-mail: giuseppe.quaranta@uniroma1.it

W. Lacarbonara
e-mail: walter.lacarbonara@uniroma1.it

S. F. Masri
Sonny Astani Department of Civil and Environmental
Engineering, University of Southern California, 3620
South Vermont Avenue, Los Angeles, CA 90089, USA
e-mail: masri@usc.edu

Keywords Artificial neural network · Computational intelligence · Differential evolution · Genetic algorithm · Genetic programming · Nonlinear system identification · Particle swarm optimization

1 Introduction

1.1 Motivations

The development of mathematical models in nonlinear dynamics can be approached in different ways:

- white box modeling, i.e., definition of the model structure and their parameters on a purely physical basis;
- gray box modeling, i.e., construction of models whose mathematical structure is known from physical insights or conceptualization (e.g., a phenomenological model), but whose parameters must be identified by means of data;
- black box modeling, i.e., pure data-driven derivation of models representative of the observed physical phenomena.

The white box modeling has been by far the most popular approach in mechanics and engineering sciences due to the attractiveness of a purely physical interpretation of the described phenomena and the afforded control of the underlying mathematical/physical assumptions/approximations. In this context, the greatest advances in nonlinear dynamic modeling of systems and structures have been achieved thanks to the fine tuning of powerful modeling techniques by which the geometrically complex 3D nonlinear behaviors of solids have been reduced to simpler kinematic and mechanical descriptions leading to specialized theories such as the geometrically exact theories of strings, rods, plates, and shells [9,27,177,242,253]. These geometrically exact theories have been further extended to more complex structures obtained by assembling several specialized substructures and multibody components such as suspension bridges, arch bridges, ropeways, and roller batteries, to mention but a few [10,11,177]. In most of these models, the geometric nonlinearities (i.e., large deformations expressed by nonlinear strain–displacement relations) are either exactly described without approximations or are well described by polynomial expansions. Moreover, the effects of geometric nonlinearities, in general, and especially in slender bodies such as beams, plates, frames, and shells (e.g., hardening associated with midplane stretching or large bending/twisting curvatures, large strains, and large rotations), are well understood.

Conversely, the material nonlinearities (nonlinear stress–strain relationships describing, e.g., softening/hardening elasticity, nonlinear viscoelasticity of polynomial or fractional-order type, elasto-plasticity, elasto-visco-plasticity, rate-independent or rate-dependent hysteresis) arising in traditional as well as in innovative materials (e.g., shape memory alloys and polymers, nanostructured materials with carbonaceous 0D/1D/2D nanofillers, multi-layer laminated composite materials, multi-phase and phase changing materials) are the most challenging aspects to embody and identify in the associated mechanical models. This is because the constitutive theories—albeit being based on axiomatic formulations ruling the physical/thermodynamical restrictions placed on them—rely on simplifying assumptions, which must be necessarily corroborated making use of extensive experimental tests and data. The complexity of the constitutive theories is often so discouraging that a reasonable simplified phenomenological or pure data-driven modeling approach is sought.

Besides the geometric and material nonlinear forces entering the equations of motion, in a large variety of mechanical systems, inertia nonlinearities may be caused by the presence of lumped or distributed masses subject to some kinematic constraints. Coriolis and centripetal or centrifugal accelerations can arise in members deforming in rotating/moving frames [13,242]. Moreover, the interactions between bodies often give rise to nonconservative forces such as inelastic impacts, friction, dead zone, relay with dead zone, backlash, saturation. Finally, also the body and surface forces can be strongly nonlinear as in the case of magnetic and electric forces (e.g., electrostatic forces which exhibit singularities).

White box modeling also allows to introduce in a suitable way the nonlinearities in the balance laws for linear and angular momentum as well as in other evolutive equations governing some states or restoring forces (e.g., internal variables describing memory-dependent forces), or in the boundary conditions, or in both.

Discretization of the governing partial differential equations carried out via one of the many viable reduction methods (e.g., Faedo–Galerkin technique or other weighted residual method, finite element method, etc.) yields a system of ordinary differential equations which, in vector-valued form, can be cast as:

$$\boldsymbol{M}\ddot{\boldsymbol{x}} + \boldsymbol{C}\dot{\boldsymbol{x}} + \boldsymbol{K}\boldsymbol{x} + \boldsymbol{n}_2\left(\boldsymbol{x},\boldsymbol{x}\right) + \boldsymbol{n}_3\left(\boldsymbol{x},\boldsymbol{x},\boldsymbol{x}\right) + \boldsymbol{n}\left(\boldsymbol{z}\right) + \boldsymbol{i}_3\left(\boldsymbol{x},\dot{\boldsymbol{x}},\dot{\boldsymbol{x}}\right) + \boldsymbol{i}_3^*\left(\boldsymbol{x},\boldsymbol{x},\ddot{\boldsymbol{x}}\right) = \boldsymbol{p}(t), \tag{1a}$$

$$\dot{\boldsymbol{z}} = \boldsymbol{f}\left(\boldsymbol{x},\dot{\boldsymbol{x}},\boldsymbol{z}\right), \tag{1b}$$

where the dot denotes differentiation with respect to time t; $\boldsymbol{x}$ is the $N \times 1$ vector of the generalized coordinates (N is the number of shape functions or nodal coordinates in the discretization scheme); $\boldsymbol{p}(t)$ is the vector of time-varying generalized excitation forces (input); $\boldsymbol{M}$ and $\boldsymbol{K}$ are $N \times N$ symmetric and positive definite matrices representing the linear mass and stiffness, respectively; $\boldsymbol{C}$ is a positive definite viscous damping matrix; $\boldsymbol{n}_2$, $\boldsymbol{n}_3$, $\boldsymbol{i}_3$, and $\boldsymbol{i}_3^*$ are $N \times 1$ non-commutative and multilinear operators which represent the nonlinear quadratic and cubic geometric forces ($\boldsymbol{n}_2$, $\boldsymbol{n}_3$) and nonlinear cubic inertia terms ($\boldsymbol{i}_3$, $\boldsymbol{i}_3^*$), respectively; $\boldsymbol{n}\left(\boldsymbol{z}\right)$ is the vector collecting the contributions from the generalized hysteretic forces $\boldsymbol{z}$ whose evolution in time is described by the second set of ordinary differential equations ruled by the nonsmooth vector field $\boldsymbol{f}$ (e.g., Bouc–Wen/Jiles–Atherton/LuGre/Prandtl–Ishlinskii/Preisach models, etc.). Depending on the considered application, the mathematical structure of nonlinear geometric, material, or inertia forces ($\boldsymbol{n}_2, \boldsymbol{n}_3, \boldsymbol{z}, \boldsymbol{i}_3, \boldsymbol{i}_3^*$) can be established from physical insights or conceptualization, but their parameters have to be identified from experimental data. In other circumstances, pure data-driven models can be conveniently adopted for them.

Often a linear modal projection of Eq. (1) is sought to decouple the linear inertia, damping and elastic forces and obtain nonlinear reduced-order models. To this end, let $\boldsymbol{B}_0$ represent the matrix collecting the eigenvectors $\boldsymbol{x}_0$ in its columns (i.e., eigensolutions of the linear undamped, unforced eigenvalue problem). By introducing the transformation $\boldsymbol{x} = \boldsymbol{B}_0\boldsymbol{q}$, the set of equations in Eq. (1) can be rewritten in the normal coordinates $\boldsymbol{q}$ as:

$$\ddot{\boldsymbol{q}} + \boldsymbol{D}\dot{\boldsymbol{q}} + \boldsymbol{\Lambda}\boldsymbol{q} + \boldsymbol{G}_2\left(\boldsymbol{q},\boldsymbol{q}\right) + \boldsymbol{G}_3\left(\boldsymbol{q},\boldsymbol{q},\boldsymbol{q}\right) + \boldsymbol{N}\left(z\right) + \boldsymbol{I}_3\left(\boldsymbol{q},\dot{\boldsymbol{q}},\dot{\boldsymbol{q}}\right) + \boldsymbol{I}_3^*\left(\boldsymbol{q},\boldsymbol{q},\ddot{\boldsymbol{q}}\right) = \boldsymbol{u}(t), \tag{2a}$$

$$\dot{z} = \hat{\boldsymbol{f}}\left(\boldsymbol{q},\dot{\boldsymbol{q}},z\right), \tag{2b}$$

where $\boldsymbol{\Lambda} = \text{diag}\left\{\omega_1^2,\ldots,\omega_j^2,\ldots,\omega_N^2\right\}$ is the diagonal matrix having the squared frequencies ω_j^2 as entries, $\boldsymbol{D} = \text{diag}\left\{2\zeta_1\omega_1,\ldots,2\zeta_j\omega_j,\ldots,2\zeta_N\omega_N\right\}$ is the diagonal matrix whose entries are the damping ratios ζ_j multiplied by the frequencies ω_j, and $\boldsymbol{u}$ is the vector of modal input. Truncation of the number of retained eigenvectors in the modal representation $\boldsymbol{x} = \sum_{j=1}^{n} q_j(t)\boldsymbol{x}_{0,j}$ with $n < N$ gives rise to reduced-order models suitable to describe the system dynamics within selected frequency ranges of interest. The use of reduced-order models is also a common practice in the identification of nonlinear dynamical systems.

Nowadays, the need of a trade-off among complexity, efficiency, and reliability in analytical or numerical studies as well as the difficulty of obtaining fully physics-based models makes gray and black box modeling more and more common in engineering sciences and practice. Note that, albeit such classification of nonlinear dynamical systems modeling is neither exhaustive nor the only exclusive one, it is especially useful to highlight the fact that white box modeling does not need collecting data (besides the knowledge of a few system parameters supposed to be available at the design stage, e.g., mass, stiffness, etc.), whereas gray box and black box modeling requires to cope with a system identification problem at different levels. In gray box modeling, the identification problem is concerned with the use of experimental data to estimate some parameters within a given phenomenological model (i.e., parametric identification). In black box modeling, the identification problem deals with the definition of a suitable model structure capable of describing the observed phenomena (i.e., nonparametric identification).

Whatever the identification problem is, it requires the implementation of the following tasks: (i) design of the experimental campaign; (ii) testing and data collection; (iii) data preprocessing; (iv) system identification; (v) validation of the final model (and comparison with alternative models, if applicable). Each step is critical for a proper modeling of nonlinear dynamical systems. However, for the sake of focus, the present work is only devoted to the broad problem of (parametric and nonparametric) nonlinear dynamical systems identification. Within this framework, the impact of theories and methodologies belonging to the area of artificial intelligence is growing tremendously. Since this trend is expected to be increasing at an even higher rate in the future, a rational survey of well-established techniques, results, and open problems can be a useful support for beginners as well as for experts interested in contributing with new ideas and novel methods to this exciting research area.

The present overview is also motivated by the fact that it is common—and somewhat natural—to apply a specific strategy in dynamic systems identification as if it is a universal tool. This is the well-known “hammer principle”: if the only known tool is a hammer, then everything looks like a nail. By reviewing several techniques based on principles and methodologies belonging to the area of artificial intelligence, this work also aims to provide a detailed presentation of a class of tools available for those who seek a suitable approach to deal with a nonlinear dynamic identification problem.

1.2 Definitions and taxonomy

The applications of evolutionary computing, swarm intelligence and neurocomputing in nonlinear dynamic systems identification are the main topics of the present state-of-the-art review. Evolutionary computation is grounded on the simulation of natural evolution processes, where the main concept is the survival of the fittest individuals. Swarm intelligence exploits the collective behavior of decentralized, self-organized natural systems of agents for problem-solving. The swarm consists of a population of simple agents whose global behavior is not governed by a centralized control structure that dictates how individual agents should behave. Conversely, the interactions between such agents lead to the emergence of an intelligent global behavior. Finally, neurocomputing is based on a parallel and distributed information processing system that attempts to mimic how the human brain works.

Following the classifications proposed in [98], the present state-of-the-art review about evolutionary computing deals with genetic algorithm (GA), differential evolution (DE), and genetic programming (GP), whereas the appraisal about swarm intelligence and neurocomputing is restricted to particle swarm optimization (PSO) and artificial neural network (ANN), respectively. According to [98], evolutionary computing, swarm intelligence, and neurocomputing are among the constituents of the computational intelligence, which is defined as a branch of artificial intelligence dealing with "the study of adaptive mechanisms to enable or facilitate intelligent behavior in complex and changing environments." The interested reader can refer, for instance, to [286] for a general overview about artificial intelligence. Such relationship between computational intelligence and artificial intelligence agrees with that suggested previously by Bezdek [26]. In his foreword to [302], Prof. Lotfi Zadeh also states that the concepts of computational intelligence and soft computing are closely related to each other, the latter being the union of computational intelligence paradigms and probabilistic computing. For the sake of completeness, it is worth noting that there is no general consensus within the scientific community about the existing definitions and classifications, and many other proposals (sometimes conflicting with each other) can be found in the literature.

1.3 Related works

The need of organizing the existing literature about nonlinear dynamic systems identification for structural mechanics and engineering applications has emerged only in recent years. For instance, the former editor in chief of *Nonlinear Dynamics*, Prof. Ali H. Nayfeh, promoted a Special Issue about the identification of nonlinear dynamical systems that was published in 2005 [92]. Two recent review papers published in *Mechanical Systems and Signal Processing* [166,246] are specifically devoted to the review of linearization methods, time-domain methods, modal methods, time-frequency approaches, black box modeling, and model updating in nonlinear dynamic system identification. None of these works was specifically conceived to cover topics addressing the use of computational intelligence methods.

A first tutorial about soft computing-based applications for mechanical systems (including systems identification) has been published in *Mechanical Systems and Signal Processing* [336]. Such a work has the merit of encompassing a huge number of topics but, as a consequence, the review of nonlinear dynamic systems identification is somewhat limited. One of the most recent state-of-the-art reviews is the preprint by Aguirre [3], where gray and black box modeling is addressed without a detailed presentation of methodologies and applications belonging to the area of artificial intelligence. To date, therefore, a comprehensive and up-to-date survey about computational intelligence methods for nonlinear dynamic systems identification is still lacking.

1.4 Outline

The remaining part of the present work is organized as follows: Sects. 2 and 3 tackle the state of the art on parametric and nonparametric identification of nonlinear dynamic systems via computational intelligence techniques, respectively. Each of these sections starts with a short presentation of the system identification problem: the parametric identification problem is reviewed in subsection 2.1 while the nonparametric identification is formulated in subsection 3.1.

In what concerns parametric identification techniques, GA, PSO, and DE are analyzed in Sects. 2.2, 2.3 and 2.4, respectively. Parametric identification of the following nonlinear dynamic models is examined: (i) the Bouc–Wen model; (ii) chaotic systems; (iii) the Jiles–Atherton model; (iv) the LuGre model; (v) the Prandtl–Ishlinskii model; (vi) the Preisach model; (vii) the Wiener–Hammerstein model. On the other hand, Sects. 3.2 and 3.3 are meant to provide an overview on nonparametric identification by means of GP and ANN, respectively.

Each of the subsections in Sects. 2 and 3 is divided into two parts: The first part contains some details about the main algorithm and its principal variants, together with some useful information and/or references for code implementation, fine tuning of control parameters, numerical issues, and theoretical foundations. The second part discusses the applications in the fields of structural mechanics and engineering (and related fields).

Finally, the goal of the closure in Sect. 4 is twofold: First, some final remarks are provided. Next, possible directions for future research are tackled.

Overall, a total of 368 literature sources are included in the references list of the present review article.

2 Parametric identification

2.1 Parametric identification problem

Computational intelligence-based parametric identification techniques herein reviewed are GA, PSO, and DE. In so doing, the parametric identification problem is formulated as a minimization problem. Therefore, let $\boldsymbol{x} = \left\{x_1 \ldots x_j \ldots x_N\right\}$ be the set of N nonlinear model parameters to be identified (with $\boldsymbol{x} \in \mathbb{R}^N$). Under the assumption of a single-objective minimization problem, let $f(\cdot)$ be a suitable real scalar measure of the similarity between the experimental response of the nonlinear system $\hat{\boldsymbol{y}}$ and that predicted using an appropriate nonlinear model ruled by $\boldsymbol{x}$, i.e., $\boldsymbol{y}(\boldsymbol{x})$. Formally, the best estimate of the model parameters can thus be obtained through the solution of the following optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{x}}\{f(\boldsymbol{x})\} \\ &\quad \text{s.t.} \\ &\quad \boldsymbol{x}^{\ell} \leq \boldsymbol{x} \leq \boldsymbol{x}^{u}, \end{aligned} \tag{3}$$

where $\boldsymbol{x}^{\ell} = \left\{x_1^{\ell} \ldots x_j^{\ell} \ldots x_N^{\ell}\right\}$ and $\boldsymbol{x}^{u} = \left\{x_1^{u} \ldots x_j^{u} \ldots x_N^{u}\right\}$ denote the lower bounds and the upper bounds of the model parameters, respectively. They define the feasible model parameters search domain for the identification problem. A simple formulation for $f(\boldsymbol{x})$ is, for instance, the normalized mean square error between the experimental response $\hat{\boldsymbol{y}}$ and that predicted by using the selected nonlinear model giving the parameter estimates $\boldsymbol{y}(\boldsymbol{x})$. It reads:

$$f(\boldsymbol{x}) = \frac{100}{\sigma_y^2 P}\left(\hat{\boldsymbol{y}} - \boldsymbol{y}(\boldsymbol{x})\right)^{\top}\left(\hat{\boldsymbol{y}} - \boldsymbol{y}(\boldsymbol{x})\right), \tag{4}$$

where P is the number of data points, σ_y^2 is the variance of the measured response, and $\top$ indicates the transpose. By using such objective function definition, according to [334], $f(\boldsymbol{x}) \leq 5.0$ indicates good model and parameter estimates and $f(\boldsymbol{x}) \leq 1.0$ can be considered an excellent outcome of the identification problem.

When the model employed to compute $\boldsymbol{y}(\boldsymbol{x})$ is based on sets of governing differential equations (as is often times the case in nonlinear dynamics), it is understood that initial conditions have to be also assigned. Initial conditions are assumed as deterministic known data of the identification problem. However, it should be kept in mind that if initial conditions are affected even by small uncertainties, then a similarity measure in terms of time histories can lead to an incorrect identification of chaotic systems [154,155] because of their well-known inherent, distinct sensitivity to initial conditions. An alternative definition of the objective function has been proposed in [156,251] in an attempt to alleviate this issue.

2.2 Genetic algorithm

2.2.1 Algorithms

GA is an optimization technique based on Darwin's theory of evolution. The method was developed in the 1960s–1970s by Holland [141], and its early implementations were mostly based on the use of the binary codification, but GAs based on floating numbers (i.e., real-coded GAs) appeared almost simultaneously [36]. Herein, the real-valued-based implementation only is reviewed since it is deemed more appropriate to handle continuous-type model parameters. The most appreciated advantages of a real-coded GA are a simpler implementation of the main operators as well as the fact that the conversion of binary strings into real values and vice versa is not required. The standard GA architecture is designed to manipulate iteratively a collection of L candidate solutions (often named individuals or chromosomes) denoted by ${}^k\boldsymbol{x}_i = \left\{{}^k x_{i1} \ldots {}^k x_{ij} \ldots {}^k x_{iN}\right\}$ (with $i = 1, \ldots, L$) at each iteration k through three important operators, namely selection, crossover, and mutation. The set of individuals ${}^k\mathcal{P}$ including all the candidate solutions at the iteration k is also named population and L is the population size.

Many variants of the classical GA have been proposed so far. Briefly speaking, a typical GA usually evolves through the following steps: (i) the objective function is first evaluated for each chromosome; (ii) chromosomes are then selected by means of the designed selection operator according to their fitness (which, for minimization problems, is inversely proportional to the objective function, from which it is derived using a suitable scaling function); (iii) selected chromosomes (parents) are combined by means of the crossover operator to produce offspring according to a given probability (i.e., probability of crossover); (iv) mutation operator is applied according to a given probability (i.e., probability of mutation); (v) the procedure is repeated on the currently new population until an arrest criterion is achieved (e.g., when the maximum number of iterations is reached or the current best objective function value does not improve after a given number of iterations). Note that, in order to enhance the convergence, a few best solutions at the current iteration are usually preserved for the next one following the so-called elitist strategy.

The selection operator aims at detecting the candidate solutions (parents) from the current population that will be combined to produce new chromosomes (offspring) by means of the crossover operator. The underlying idea of any selection operator is that the combination of good parents is likely to produce better candidate solutions through the crossover phase, but worst individuals should not be discarded at all because they bring information about potentially promising regions of the search space. The roulette wheel selection is among the most popular selection operators in GAs. It chooses parents by simulating a roulette wheel, where the area of each section of the wheel corresponding to an individual is proportional to its selection probability (i.e., the larger is the selection probability, the larger is the area). This, in turn, depends on chromosome's fitness (i.e., the larger is the fitness, the larger is the area of the wheel corresponding to that chromosome). A uniform random number between 0 and 1 is then generated, and the individual corresponding to the section of the roulette wheel that contains the extracted number is selected.

Tournament selection is another common approach in GAs. It closely mimics mating competition in nature by picking randomly a small subset of chromosomes (this number is called tournament size): in this case, the chromosome with the highest fitness in this subset becomes a parent. The typical tournament size lies between 2 and 4.

An important issue for all selection operators is the selection pressure, i.e., the degree to which the better individuals are favored (the higher the selection pressure, the more the better individuals are favored). In case of the roulette wheel selection, the larger is the difference among the areas of the sections, the higher is the selection pressure, as the winner will, on average, have a lower objective function than the winner of a roulette wheel selection in which the sectors area are similar to each other. For the tournament selection operator, a higher selection pressure can be attained by increasing the tournament size, as the winner from a larger tournament will, on average, have a higher fitness than the winner of a smaller tournament. Haupt and Haupt [131] concluded that it is very difficult to give advice on which selection scheme works best, but Zhong et al. [368] found that GA with tournament selection strategy usually exhibits better performance than roulette wheel selection.

For reducing bias and inefficiency in the selection algorithm, the stochastic universal sampling has been proposed by Baker [17]. To figure out how this works, it is possible to imagine the individuals arranged into adjacent segments of a line, such that each individual's segment is equal in size to its fitness (exactly as in roulette wheel selection). Then, as many equally spaced pointers as the number of required selections are generated along this line starting from a randomly chosen position, and the individuals corresponding to the pointer positions will be selected.

The crossover phase aims at combining usually two but also even more parents in such a way to produce the offspring. Several crossover (also called recombination) operators have been proposed so far. One of the most common crossover operators is the heuristic crossover. Let ${}^{k}\boldsymbol{x}_1$ and ${}^{k}\boldsymbol{x}_2$ be two selected parents obtained from the previous iteration, then one offspring ${}^{(k+1)}\boldsymbol{o}_1$ can be generated in the current iteration according to the heuristic crossover as follows:

$$ {}^{(k+1)}\boldsymbol{o}_1 = {}^{(k+1)}r \left({}^{k}\boldsymbol{x}_1^* - {}^{k}\boldsymbol{x}_2 \right) + {}^{k}\boldsymbol{x}_1^*, \tag{5} $$

where ${}^{(k+1)}r$ is a uniform random number between 0 and 1 freshly generated at each iteration and $(\cdot)^*$ identifies the best parent (i.e., $f({}^{k}\boldsymbol{x}_1^*) \leq f({}^{k}\boldsymbol{x}_2)$). Deep and Thakur [77] proposed the Laplace Crossover, which generates two offsprings from two parents as follows:

$$ {}^{(k+1)}\boldsymbol{o}_1 = {}^{k}\boldsymbol{x}_1 + {}^{(k+1)}\delta |{}^{k}\boldsymbol{x}_1 - {}^{k}\boldsymbol{x}_2|, \tag{6a} $$

$$ {}^{(k+1)}\boldsymbol{o}_2 = {}^{k}\boldsymbol{x}_2 + {}^{(k+1)}\delta |{}^{k}\boldsymbol{x}_1 - {}^{k}\boldsymbol{x}_2|, \tag{6b} $$

where ${}^{(k+1)}\delta = \xi_1 - \xi_2 \log {}^{(k+1)}r$ if ${}^{(k+1)}r \leq 0.5$ and ${}^{(k+1)}\delta = \xi_1 + \xi_2 \log {}^{(k+1)}r$ otherwise, with ξ_1 and ξ_2 two user-selected control parameters. Ling and Leung [198] proposed the average-bound crossover operator. It generates four offsprings (of which the best two only are retained) as follows:

$$ {}^{(k+1)}\boldsymbol{o}_1 = \frac{{}^{k}\boldsymbol{x}_1 + {}^{k}\boldsymbol{x}_2}{2}, \tag{7a} $$

$$^{(k+1)}\boldsymbol{o}_2 = \frac{(\boldsymbol{x}^u + \boldsymbol{x}^\ell)(1-\omega_a) + ({}^k\boldsymbol{x}_1 + {}^k\boldsymbol{x}_2)\omega_a}{2}, \tag{7b}$$

$$^{(k+1)}\boldsymbol{o}_3 = \boldsymbol{x}^u(1-\omega_b) + \max\{{}^k\boldsymbol{x}_1, {}^k\boldsymbol{x}_2\}\omega_b, \tag{7c}$$

$$^{(k+1)}\boldsymbol{o}_4 = \boldsymbol{x}^\ell(1-\omega_b) + \min\{{}^k\boldsymbol{x}_1, {}^k\boldsymbol{x}_2\}\omega_b, \tag{7d}$$

where ω_a and ω_b are two control parameters between 0 and 1, whereas $\max\{{}^k\boldsymbol{x}_1, {}^k\boldsymbol{x}_2\}$ and $\min\{{}^k\boldsymbol{x}_1, {}^k\boldsymbol{x}_2\}$ denote the vectors whose elements are obtained by taking the maximum and the minimum between the corresponding elements of ${}^k\boldsymbol{x}_1$ and ${}^k\boldsymbol{x}_2$, respectively. Depending on the adopted crossover operator, if it fails to produce a feasible offspring, then this is replaced with a random point in the feasible search space. The likelihood of crossover being applied depends on a probability defined by a crossover rate p_c (or crossover probability), which is a user-defined value between 0 and 1 (usually assumed as constant control parameter).

On the other hand, the main goal of the mutation operator is to ensure a proper level of exploration of the search space by introducing a suitable level of random perturbations in the current population, thus preventing the premature convergence toward a local optimal solution. The likelihood of mutation being applied depends on the mutation rate p_m (or mutation probability), which is a user-selected value between 0 and 1. For instance, in combination with the average-bound crossover operator, Ling and Leung [198] proposed the wavelet mutation, which generates a mutated chromosome as follows:

$$^{(k+1)}\tilde{\boldsymbol{o}}_i = \begin{cases} {}^k\boldsymbol{o}_i + {}^{(k+1)}\Lambda(\boldsymbol{x}^u - {}^k\boldsymbol{o}_i) & \text{if } {}^{(k+1)}\Lambda > 0 \\ {}^k\boldsymbol{o}_i + {}^{(k+1)}\Lambda(\boldsymbol{o}_i - \boldsymbol{x}^\ell) & \text{otherwise} \end{cases}, \tag{8}$$

where

$$^{(k+1)}\Lambda = \frac{1}{\sqrt{^{(k+1)}\alpha}} e^{-\frac{\left({}^{(k+1)}\zeta / {}^{(k+1)}\alpha\right)^2}{2}} \cos\left(5\frac{^{(k+1)}\zeta}{^{(k+1)}\alpha}\right), \tag{9}$$

in which ${}^{(k+1)}\alpha$ is a dilatation parameter depending on the ratio between the current iteration number and the maximum number of generations, whereas ${}^{(k+1)}\zeta$ is randomly generated between −2.5 and 2.5. Depending on the adopted mutation operator, if a mutated component of the chromosome (also named gene) lies outside the feasible search space, then this is replaced with a random number in the corresponding feasible range of values.

The interested reader can refer, for instance, to [61,69,131,232,304] for more information about GAs and their programming (to this end, the interested reader can also refer to the pseudocode provided in Algorithm 1).

Algorithm 1 Pseudocode of the standard GA algorithm

```
Require: Problem data
Require: Control parameters of the optimizer
  Set $k = 1$
  Initialize the population ${}^k\mathcal{P} = \{{}^k\boldsymbol{x}_1, \ldots, {}^k\boldsymbol{x}_i, \ldots, {}^k\boldsymbol{x}_L\}$
  for $i \leftarrow 1, L$ do
    Calculate $f({}^k\boldsymbol{x}_i)$ and the associated fitness value
  end for
  Set to "false" the status of the stopping criterion
  while stopping criterion = "false" do
    Apply the elitist strategy by copying candidate solutions from ${}^k\mathcal{P}$ into ${}^{(k+1)}\mathcal{P}$
    Apply the selection operator on ${}^k\mathcal{P}$
    Apply the crossover operator to produce offspring, and insert them into ${}^{(k+1)}\mathcal{P}$
    Apply the mutation operator on ${}^{(k+1)}\mathcal{P}$
    for $i \leftarrow 1, L$ do
      Calculate $f({}^{(k+1)}\boldsymbol{x}_i)$ and the associated fitness value
    end for
    Move the population ${}^{(k+1)}\mathcal{P}$ to the next iteration
    Set $k = k + 1$
    Decide whether the status of the stopping criterion switches from "false" to "true"
  end while
  return Carry out best solution and corresponding objective function value
```

It is widely recognized that the crossover operator and, to a lesser extent, the mutation operator, plays a fundamental role in the overall performance of GAs. Particularly, as this short review has highlighted, crossover operators might be largely different each other in terms of number of involved parents, number of generated offspring, number of control parameters, and the discrimination of the parents according to their fitness.

As regards the mutation operators available in the literature, the main difference among the existing proposals is the way by which the magnitude of the perturbation is settled. Most of them (as the mutation operator considered in the present review) also includes one or more control parameters that reduce the magnitude of the mutation when the GA is converging, in such a way to privilege the exploitation of the best solutions rather than the exploration of the search space in the conclusive phase of the iterative procedure. Since a huge number of different crossover and mutation operators are available (even not all of them are validated in a robust manner), it is not possible to provide general guidelines about the numerical values of the involved control parameters. In this sense, a more complete list of well-established crossover and mutation operators for GAs is given, for instance, by Herrera et al. [137].

Other important parameters in GAs are the population size, the probability of crossover, and the probability of mutation. As regards the population size L, it is somewhat dependent on the number of involved parameters N, that is, the larger the number of involved parameters N, the larger the population size L should be. This is quite common for all population-based computational intelligence techniques.

Generally, the population size should not be too small to jeopardize the exploration of the search space, but it should be not too large to reduce the risk of stagnation and to avoid an excessive computational time. As an example, the population size in [77] is taken to be ten times the number of variables (i.e., $L = 10N$). However, a well-known variant of the standard GA able to manage a very small population size has been proposed by Krishnakumar [172] (the so-called micro-GA, μGA), whereas other existing variants employ a dynamically adjusted population size [170]. In some variants of the standard GA, the whole population is divided into multiple subpopulations (the so-called multi-species GA, MGA), see for instance [218,262]. In such a case, different operators and/or numerical values of the control parameters apply to different subpopulations, and the exchange of information between them is ensured by the so-called migration operator. The adopted values for the crossover probability p_c are not so different from each other and typically lie between 0.5 and 1.0. For example, Herrera et al. [137] adopted $p_c = 0.60$, Deep and Thakur [77] assumed $p_c = 0.50$ or $p_c = 0.70$ (depending on the crossover operator), whereas Chelouah and Siarry [58] considered $p_c = 0.85$. The mutation probability value, on the other hand, can be quite different because it should be related to the exploration ability of the GA, i.e., population size, impact of the mutation operator, and the possible dynamic adjustment of the perturbation during the iterative procedure. As an example, Herrera et al. [137] assumed $p_m = 0.005$, whereas Deep and Thakur [77] adopted p_m between 0.005 and 0.020 (depending on which combination of crossover and mutation operator is employed). Chelouah and Siarry [58] adopted $p_m = 0.90$, but this value decades exponentially as the GA progresses.

In conclusion, it is also useful to highlight that important results have been achieved to explain theoretically how a GA works. A seminal contribution about the convergence of real-coded GA is due to Goldberg [119]. Other important results are also presented in [100,292,293].

2.2.2 Applications

The GA-based identification of the Bouc–Wen model and its variants have been addressed in many researches. Ha et al. [128] applied a standard real-coded GA to the parametric identification of a piezoelectric hysteretic actuator. To this end, the authors employed a modified Bouc–Wen model and compared three alternative fitness function formulations. The experimental identification of a piezoelectric actuator demonstrates that the minimization of the error in terms of hysteretic state parameter or system displacement leads to voltage–displacement cycles in good agreement with the measurements. A modified Bouc–Wen model was proposed in [173] to account for nonsymmetric hysteresis in magneto-rheological fluid dampers, and the corresponding parameters were identified using a GA that aims at minimizing the averaged sum of normalized square error between the simulated and experimental force data. A hybrid evolutionary strategy based on the sawtooth GA [170] has been implemented by Charalampakis and Koumousis [55] to identify the parameters of the Bouc–Wen model. The authors performed a numerical study in which the influence of the dynamic loading is investigated by taking into account harmonic dynamic forces with different amplitudes and a seismic base acceleration. The algorithm therein proposed is also compared to standard GA and μGA using experimental data related to the hysteretic response of a bolted-welded steel connection.

Fung and Lin [109] as well as Fung et al. [110] addressed the system identification of a precision positioning table, considering first a planar two-degree-of-freedom (DOFs) system and then a six-DOF system. The parameters of the Bouc–Wen model adopted to simulate the hysteresis of the piezoelectric actuator were identified by means of a standard real-coded GA. Monti et al. [235] applied a MGA [218] to identify the parameters of the Bouc–Wen model, and final results were compared to those obtained using standard GA and μGA. Xiaomin et al. [340] illustrated the parametric identification of a magneto-rheological damper using GA. Sireteanu et al. [303] modified the original Bouc–Wen model to facilitate its application to a wider range of hysteresis phenomena, including hysteresis cycles with an inflection point on the loading branches. The parameters of such a novel variant are then estimated through a GA and using experimental data from laboratory tests on some seismic protection devices (i.e., an elastomeric bearing for seismic isolation and a buckling restrained dissipative brace). Liu et al. [203] developed the so-called single-parameter adjustment method for the parametric identification of a magneto-rheological damper. Specifically, a modified Bouc–Wen model is considered and its parameters are analyzed in succession using the new method, in such a way that only one parameter is determined in each step for different applied current levels. The relationships between model parameters and applied current are then determined through curve fitting. Chen et al. [59] successfully applied a real-coded GA to identify a Scott–Russell mechanism driven by a piezoelectric actuator. The resulting identification problem involves a quite large number of parameters, namely equivalent resistance, capacitance, mass, elastic coefficient, damping coefficient, piezoelectric coefficient, and coefficients of the Bouc–Wen model.

The comparative evaluation in [54] encompasses different versions of GAs, PSO, and DE algorithms for the parameter estimates of the Bouc–Wen model. The parametric identification of the non-degrading hysteretic response of a bolted-welded steel connection based on experimental data demonstrates that GA-based strategies are less robust then PSO and DE. Lin et al. [195] made use of the GA for the parametric identification of a squeeze-mode magneto-rheological damper. Since the identified parameters of the Bouc–Wen model depend on applied current, excitation amplitude, and frequency, the authors argue that its implementation can cause some difficulties in setting up a robust vibration control apparatus. To cope with this issue, therefore, Lin et al. [195] also considered a viscosity model and concluded that it is as accurate as the Bouc–Wen model, but it is less sensitive to variations in amplitude and frequency of the input excitation.

Differently from the most common applications, Ortiz et al. [250] formulated the parametric identification of a modified version of the Bouc–Wen model as multi-criteria optimization problem, which is solved by means of a well-established multi-objective GA [76,215,216]. Specifically, the authors considered the Bouc–Wen–Baber–Noori model (which also accounts for degradation and pinching). The minimization problem involves four criteria, namely weighted and maximum error between measured and simulated displacements as well as final and maximum difference between measured and simulated total dissipated energy. The final model parameters are the non-dominated solution of the normalized Pareto front for which the Euclidean distance from the origin is minimum. The parameters of a modified version of the Bouc–Wen–Baber–Noori model were identified by Sengupta and Li [295] to predict the response of reinforced concrete beam–column joints with limited transversal reinforcement.

Further applications about parametric identification of the hysteretic response of dampers using the Bouc–Wen model are given in [15,16,301]. Lin et al. [196] observed that, after a parametric identification carried out using a GA, the Bouc–Wen model and the Prandtl–Ishlinskii model can both predict accurately the hysteresis of a pneumatic artificial muscle, and both perform better than the Maxwell slip model. Zaman and Sikder [358] addressed the parametric identification of the magnetization–magnetic field loop by amorphous materials considering the Bouc–Wen model and using different soft computing methods, including a standard GA.

The calibration of cyclic models for hollow cross-sectional steel members is addressed in [62] and [63]. The phenomenological model herein considered is a variant of the original Bouc–Wen model proposed by Sivaselvan and Reinhorn [305], which was here adopted to describe the cyclic behavior of the plastic region at the end of a steel member as a zero-length element (plastic hinge). A useful conclusion drawn by [62] is that, if available, it is better to take into account the monotonic response together with the cyclic response within a multi-objective formulation of the parametric identification problem. The ill conditioning of the identification problem was specifically investigated in [63]. In such a work, the authors highlight the fact that the a priori local/global sensitivity analyses are useful to assess the importance (and thus the identifiability) of the model parameters, whereas a posteriori uncertainty analyses provide a useful support to measure the fidelity of the final parameter estimates. The authors also showed that a multi-objective approach in which a certain tolerance is admitted in the definition of dominant solutions (tolerance-based Pareto optimality) leads to similar results to those achieved after sensitivity analyses, but at lower computational cost.

Recently, the GA-based parametric identification of further novel variants of the Bouc–Wen model has been reported by Pelliciari et al. [259] and Bartkowski et al. [21]. In detail, Pelliciari et al. [259] proposed a new variant that accounts for degradation and pinching by involving a number of parameters smaller than that involved in the Bouc–Wen–Baber–Noori model. To deal with the strong asymmetric shape of the cyclic loading loops observed in cylindrical samples made of vacuum packed particles, Bartkowski et al. [21] assumed that different values of the Bouc–Wen model parameters apply for the elongation and compression phases.

Within the framework of the GA-based identification of chaotic dynamics, Marano et al. [217] applied a MGA to identify the parameters of the van der Pol oscillator. Modares et al. [233] addressed the simultaneous identification of two or three model parameters of the Lorenz system using a standard GA and two versions of the PSO. Herein, the accuracy of the GA has been found to be several orders of magnitude less than that achieved by standard and improved PSO. This result is independently confirmed by Li et al. [186], who compared the performance of GA, PSO, and two variants of the gravitational search algorithm. The work of Tang et al. [315] deals with the identification of fractional-order chaotic systems via GA and DE. This study especially highlights that the accuracy of the parameter estimates gets significantly worse when the fractional order of the system is not known and is itself a parameter to be estimated. Li and Yin [191] as well as Zhang et al. [360] illustrated new hybridized algorithms for the parametric identification of chaotic systems. In their works, the performance of such novel techniques is subjected to a comparative evaluation that also involves the standard GA. It is worth noting that these recent studies provide further evidences of the fact that the standard GA is less effective than PSO and DE in parametric identification of chaotic systems.

Some of the first applications of GA in the parametric identification problem of the Jiles–Atherton model are reported in [8] and [333]. Particularly, the use of different weights is proposed in [333] to cope with both major and minor hysteresis loops in the parametric identification of magnetic materials. The authors also implemented the simulated annealing (SA) algorithm [25] in order to refine the best parameter estimates achieved by means of the GA. The combined use of GA and SA for parametric identification problems has been also found effective by Fulginei and Salvini [106] in case of magnetic hysteresis characterized by symmetric or asymmetric loops. Note that the first attempts in using GA for the parametric identification of the Jiles–Atherton model exploit a binary codification [8,333], but the applications based on real codification soon became more popular [66,184].

On the other hand, a hybrid encoding strategy was implemented in [365,366] to identify the parameters of a modified Jiles–Atherton model. In detail, while a binary codification with a large population size is adopted at the beginning of the iterative procedure, floating numbers and reduced population size are adopted soon as a few number of iterations is completed. Jia et al. [157] developed a novel force sensor that includes a giant magnetostrictive material rod. They implemented a hybrid optimizer based on the combination of GA and SA in order to identify the unknown magnetic and magnetostrictive parameters within a new phenomenological model based, in turn, on the Jiles–Atherton model.

A self-adaptive GA was elaborated in [206] to estimate the parameters of the Jiles–Atherton model for ferromagnetic materials. The self-adaptive behavior of the proposed algorithm is mainly due to the fact that selection pressure and mutation rate are adjusted based on the current iteration number. A more recent application of the GA for parametric identification of the hysteretic behavior exhibited by ferromagnetic materials can found in [136], in which the effects due to plastic deformations are particularly investigated.

Do et al. [82] used the GA to identify the parameters of a modified version of the LuGre model, which is formulated to cope with the nonlinear friction in the tendon-sheath mechanism. Further recent applications of GA in the parametric identification of the LuGre model are reported in [152,252] while parameters estimation of the Prandtl–Ishlinskii model is addressed in [196,347]. In detail, Yang et al. [347] performed the parametric identification of the Prandtl–Ishlinskii model through a modified coral reef optimization algorithm whose accuracy is almost identical to that they obtained by using a standard GA. Preisach model identification applications via GA are reported in [70] and [135] using experimental data for Si–Fe non-oriented grain steels and Fe–3 wt%Si steel sheet, respectively.

The identification of linear systems with static nonlinearities in the input (Hammerstein model) or the output (Wiener model) was also successfully performed by means of GA, see, for instance, [7,91,187,227,228,276]. Particularly, Dotoli et al. [91] also encoded the structure of the model (type of nonlinearity, number of zeros, and poles) and considered different nonlinearities (Coloumb friction, dead zone, saturation, relay with dead zone, fifth nonlinearity). These authors showed that the zero position is the most difficult parameter to identify in the Hammerstein models and found a similar accuracy level but a lower robustness in the GA-based identification of Wiener models.

For a quick overview on the application of GAs in nonlinear parametric identification problems, the reader can also refer to Table 1.

2.3 Particle swarm optimization

2.3.1 Algorithms

The PSO algorithm proposed by Kennedy and Eberhart [165] is likely the most common swarm intelligence-based optimizer. Other popular swarm intelligence-based algorithms are the ant colony optimization (ACO) algorithm [90], the bacterial foraging optimization (BFO) algorithm [236], the artificial bee colony (ABC) algorithm [163], and the Jaya algorithm [60,277], just to cite a few among many other proposals in the field. For the application of less common swarm intelligence-based algorithms in nonlinear dynamic systems identification, the interested reader can refer, for instance, to [350]. The present review is basically restricted to the PSO algorithm, its variants and hybridizations. Within this framework, two alternative hypotheses can be adopted to regulate the dynamic behavior of the swarm, thereby obtaining:

- a classical PSO algorithm, in which it is assumed that the Newtonian dynamics determines the movement of the agents (particles). Therefore, position and velocity of each agent can be determined simultaneously;
- a quantum-behaved PSO (QPSO) algorithm, in which the Newtonian dynamics is replaced with the quantum mechanics, where the term "trajectory" is meaningless.

The classical PSO algorithm considers a swarm consisting of L particles which is manipulated through a series of iterations. For each iteration k, the particles modify their main attributes, namely position ${}^k\boldsymbol{x}_i = \left\{{}^k x_{i1} \ldots {}^k x_{ij} \ldots {}^k x_{iN}\right\}$ and velocity ${}^k\boldsymbol{v}_i = \left\{{}^k v_{i1} \ldots {}^k v_{ij} \ldots {}^k v_{iN}\right\}$ (with $i = 1, \ldots, L$). The swarm and its velocity field at iteration k are denoted by ${}^k\mathcal{S}_p$ and ${}^k\mathcal{S}_v$, respectively.

In order to ensure the cohesion of the swarm, the absolute particle velocity $|{}^k v_{ij}|$ is forced to be less than a maximum velocity $v_j^{\max}$, with $\boldsymbol{v}^{\max} = \left\{v_1^{\max} \ldots v_j^{\max} \ldots v_N^{\max}\right\}$. It is commonly assumed $\boldsymbol{v}^{\max} = \nu(\boldsymbol{x}^u - \boldsymbol{x}^\ell)/\tau$, where the time factor $\tau = 1$ is introduced to preserve a physical meaning of the formula, whereas ν defines how far a particle can move starting from its current position. Although no uniform consent exists about its value, it is generally assumed $\nu = 0.5$. At the beginning (i.e., $k = 1$), the initial set of candidate solutions (i.e., the initial position ${}^1\boldsymbol{x}_i$ of each particle) is obtained by generating L samples within the assigned search space according to a uniform probability distribution function. Similarly, the initial particle velocities are generated using a uniform probability distribution function in such a way that $-v_j^{\max} \leq {}^1 v_{ij} \leq v_j^{\max}\ \forall i, j$. Therefore, all attributes of the particles are designed to be feasible at the beginning. At each iteration $(k+1)$, the particle velocity is first modified according to the following rule:

$$
\begin{aligned}
{}^{(k+1)}\boldsymbol{v}_i = w\,{}^k\boldsymbol{v}_i + \gamma_1\,{}^{(k+1)}\boldsymbol{r}_{i1} \times \left({}^k\boldsymbol{x}_i^{Pb} - {}^k\boldsymbol{x}_i\right) \\
+ \gamma_2\,{}^{(k+1)}\boldsymbol{r}_{i2} \times \left({}^k\boldsymbol{x}_i^{*b} - {}^k\boldsymbol{x}_i\right),
\end{aligned}
\qquad (10)
$$

and ${}^{(k+1)}v_{ij} = \operatorname{sign}\left({}^{(k+1)}v_{ij}\right) v_j^{\max}$ if $\left|{}^{(k+1)}v_{ij}\right| > v_j^{\max}$.

Herein, the parameter w is the inertia weight, as introduced by Shi and Eberhart [299] to enhance the balance between exploration of the search space and exploitation of the current best solutions. The parameters γ_1 and γ_2 are the so-called acceleration factors (they are also named cognitive and social parameter, respectively). In the standard version, all these parameters are constant values. The symbol × denotes the term-by-term product, whereas ${}^{(k+1)}\boldsymbol{r}_{i1}$ and ${}^{(k+1)}\boldsymbol{r}_{i2}$ are vectors whose N terms are random numbers uniformly distributed between 0 and 1 (these random vectors are freshly generated at each new iteration). Moreover, ${}^k\boldsymbol{x}_i^{Pb}$ is the best previous particle position (also known as *pbest*), that is,

Table 1 Parametric identification applications using GA

Nonlinear model	References	Applications
Bouc–Wen	Ha et al. [128]	Piezoelectric actuator
	Kwok et al. [173]	Magneto-rheological damper
	Charalampakis and Koumousis [55]	Bolted-welded steel connection
	Fung and Lin [109]	Precision positioning table
	Fung et al. [110]	Precision positioning table
	Monti et al. [235]	Numerical simulations using μGA and MGA
	Xiaomin et al. [340]	Magneto-rheological damper
	Sireteanu et al. [303]	Elastomeric base isolators, dissipative braces
	Liu et al. [203]	Magneto-rheological damper
	Chen et al. [59]	Piezoelectric actuator in Scott–Russell mechanism
	Charalampakis and Dimou [54]	Bolted-welded steel connection
	Lin et al. [195]	Magneto-rheological damper
	Ortiz et al. [250]	Ferrocement wall
	Sengupta and Li [295]	Reinforced concrete beam–column joints
	Atabay and Ozkol [15]	Magneto-rheological damper
	Bai et al. [16]	Magneto-rheological damper
	Lin et al. [196]	Pneumatic artificial muscle
	Zaman and Sikder [358]	Magnetic behavior of amorphous material
	Chisari et al. [62]	Plastic hinge in hollow cross-sectional steel members
	Shu and Li [301]	Metallic dampers
	Chisari et al. [63]	Plastic hinge in hollow cross-sectional steel members
	Pelliciari et al. [259]	Reinforced concrete bridge piers
	Bartkowski et al. [21]	Vacuum packed particles
Chaotic system	Marano et al. [217]	van der Pol system
	Modares et al. [233]	Lorenz system
	Li et al. [186]	Lorenz system
	Tang et al. [315]	Fractional-order Lü and Volta systems
	Li and Yin [191]	Chen and Lorenz systems
	Zhang et al. [360]	Lorenz system
Jiles–Atherton	Almeida et al. [8]	Magnetostrictive transducer
	Wilson et al. [333]	Ferrite toroidal core
	Leite et al. [184]	Hysteresis loops of magnetic materials
	Fulginei and Salvini [106]	Silicon-steel sheet hysteresis loop
	Chwastek and Szczyglowski [66]	Magnetic behavior of amorphous material
	Zheng et al. [365]	Magnetostrictive actuator
	Zheng et al. [366]	Magnetostrictive actuator
	Jia et al. [157]	Magnetostrictive sensor
	Lu et al. [206]	Silicon-steel sheet hysteresis loop
	Hergli et al. [136]	Silicon-steel sheet hysteresis loop
LuGre	Do et al. [82]	Tendon-sheath actuated surgical system
	Irakoze et al. [152]	Piezoelectric actuator
	Pang et al. [252]	Control of optoelectronic tracking system

Table 1 continued

Nonlinear model	References	Applications
Prandtl–Ishlinskii	Lin et al. [196]	Pneumatic artificial muscle
	Yang et al. [347]	Tracking of magnetostrictive actuator
Preisach	Consolo et al. [70]	Hysteresis loops of magnetic material
	Hergli et al. [135]	Silicon-steel sheet hysteresis loop
Wiener–Hammerstein	Li [187]	Numerical simulations
	Al-Duwaish [7]	Numerical simulations
	Dotoli et al. [91]	Numerical simulations
	Mehmood et al. [227]	Numerical simulations
	Raja et al. [276]	Numerical simulations
	Mehmood et al. [228]	Numerical simulations

$$
{}^{k}\boldsymbol{x}_i^{Pb} = \begin{cases} {}^{(k-1)}\boldsymbol{x}_i^{Pb} & \text{if } f\left({}^{k}\boldsymbol{x}_i\right) > f\left({}^{(k-1)}\boldsymbol{x}_i^{Pb}\right) \wedge k > 1 \\ {}^{k}\boldsymbol{x}_i & \text{otherwise} \end{cases} . \qquad (11)
$$

The way by which ${}^{k}\boldsymbol{x}_i^{*b}$ is defined in Eq. (10) regulates how large the influence of other particles is on the ith particle at hand. In the global PSO algorithm, it is ${}^{k}\boldsymbol{x}^{Gb}$, which is the best particle (also known as *gbest*) within the whole swarm (in such a case, the swarm is said to be fully connected or fully informed). In the local PSO algorithm, it is ${}^{k}\boldsymbol{x}_i^{Lb}$, which is the best particle among a small set of particles adjacent to the ith particle at hand (also known as *lbest*).

Afterward, the particle position is updated as follows:

$$
{}^{(k+1)}\boldsymbol{x}_i = {}^{k}\boldsymbol{x}_i + \tau\, {}^{(k+1)}\boldsymbol{v}_i . \qquad (12)
$$

Finally, the feasibility of the new particle position is ensured through the following inequalities check:

$$
\left({}^{(k+1)}x_{ij}, {}^{(k+1)}v_{ij}\right) = \begin{cases} \left({}^{(k+1)}x_{ij}, {}^{(k+1)}v_{ij}\right) & \text{if } x_j^{\ell} \le {}^{(k+1)}x_{ij} \le x_j^{u} \\ \left(x_j^{\ell}, 0\right) & \text{if } {}^{(k+1)}x_{ij} < x_j^{\ell} \\ \left(x_j^{u}, 0\right) & \text{otherwise} \end{cases} . \qquad (13)
$$

At this point, all particles of the swarm have updated their position and velocity for the next iteration, i.e., ${}^{(k+1)}\mathcal{S}_p$ and ${}^{(k+1)}\mathcal{S}_v$ have been carried out. The iterative strategy stops when a termination criterion is satisfied, e.g., when the maximum number of objective function evaluations is reached or no improvement is observed after a given number of iterations.

The pseudocode of the standard PSO algorithm is provided in Algorithm 2 in order to facilitate its implementation.

Algorithm 2 Pseudocode of the standard PSO algorithm

```
Require: Problem data
Require: Control parameters of the optimizer
  Set $k = 1$
  Initialize the swarm ${}^{k}\mathcal{S}_p = \{{}^{k}\boldsymbol{x}_1, \ldots, {}^{k}\boldsymbol{x}_i, \ldots, {}^{k}\boldsymbol{x}_L\}$
  Initialize the swarm velocity field ${}^{k}\mathcal{S}_v = \{{}^{k}\boldsymbol{v}_1, \ldots, {}^{k}\boldsymbol{v}_i, \ldots, {}^{k}\boldsymbol{v}_L\}$
  for $i \leftarrow 1, L$ do
    Calculate $f\left({}^{k}\boldsymbol{x}_i\right)$
  end for
  Set to "false" the status of the stopping criterion
  while stopping criterion = "false" do
    for $i \leftarrow 1, L$ do
      Update the particle's velocity, thereby obtaining ${}^{(k+1)}\boldsymbol{v}_i$
      Update the particle's position, thereby obtaining ${}^{(k+1)}\boldsymbol{x}_i$
      Ensure that ${}^{(k+1)}\boldsymbol{x}_i$ does not violate boundary constraints
      Calculate $f\left({}^{(k+1)}\boldsymbol{x}_i\right)$
    end for
    Move the swarm ${}^{(k+1)}\mathcal{S}_p$ and its velocity field ${}^{(k+1)}\mathcal{S}_v$ to the next iteration
    Set $k = k + 1$
    Decide whether the status of the stopping criterion switches from "false" to "true"
  end while
  return Carry out best solution and corresponding objective function value
```

It is evident that the algorithm performance depends on the numerical value of its control parameters. An important parameter is the swarm size L, which should depend on the number of unknowns N as for other population-based optimization techniques. This value should be large enough to ensure a satisfactory exploration, but a too high value of the population size increases the elaboration time as well as the risk of stagnation. In this context, Bratton and Kennedy [39] did not find significant differences when this parameter varies between 20 and 100 for a maximum number of variables N equal to 30. Huang and Mohan [147] also proposed a variant of the standard PSO able to handle a very small swarm size (so-called micro-PSO, μPSO) while other studies consider a variable swarm size [179].

It is commonly recognized that inertia weight and acceleration factors play a more important role. Initial studies in this regard state that the inertia weight should lie between 0.8 and 1.2 in order to obtain satisfactory results, whereas a typical assumption for the acceleration factors is $\gamma_1 = \gamma_2 = 2$. More recent evidences typically show that an improved performance can be obtained by varying the numerical value of these control parameters during the iterative procedure. As an example, Shi and Eberhart [300] reduced the inertia weight linearly from 0.9 to 0.4 at the end of the iterative procedure. Besides a linearly varying inertia weight, Ratnaweera et al. [278] also included iteration-varying acceleration factors in their variants of the PSO algorithm. In their study, the cognitive parameter varies linearly between 2.5 and 0.5 while the social parameter varies linearly between 0.5 and 2.0. Further proposals exist in which the numerical value of the control parameters is defined randomly or according to chaotic maps. As an example, dos Santos Coelho and Herrera [84] proposed several versions of a chaotic PSO (CPSO) algorithm in which one, two, or all the control parameters (i.e., inertia weight and acceleration factors) are adapted according to a chaotic map. They use the Zaslavskii map, whose outcome is scaled between 0.4 and 0.9 for the inertia weight, whereas it is scaled between 0.5 and 2.5 for the acceleration factors.

A well-known variant of the PSO algorithm proposed by Clerc [67] employs the so-called constriction factor, see also [95]. The constriction factor χ is alternative to the use of the static inertia weight, and it was initially proposed to replace the $\boldsymbol{v}^{\max}$ clamping. By adopting this approach, the particle velocity is estimated as follows:

$$\begin{aligned} {}^{(k+1)}\boldsymbol{v}_i = \chi \Big[{}^{k}\boldsymbol{v}_i + \gamma_1 {}^{(k+1)}\boldsymbol{r}_{i1} \times \left({}^{k}\boldsymbol{x}_i^{Pb} - {}^{k}\boldsymbol{x}_i \right) \\ + c_2 {}^{(k+1)}\boldsymbol{r}_{i2} \times \left({}^{k}\boldsymbol{x}^{Gb} - {}^{k}\boldsymbol{x}_i \right) \Big], \end{aligned} \tag{14}$$

where the constriction factor χ is

$$\chi = \frac{2}{\left| 2 - \varphi - \sqrt{\varphi^2 - 4\varphi} \right|}, \tag{15}$$

with $\varphi = \gamma_1 + \gamma_2 > 4$. When the constriction factor strategy is employed, it is typically assumed $\varphi = 4.1$ (and thus $\chi = 0.729$) with the acceleration factors both equal to 2.05 [95].

Within the QPSO algorithm, the dynamic behavior of the particle is inspired by quantum mechanics and, as a consequence, it is largely different from that of the classical PSO algorithm. In this case, the L particles of the swarm move according to the following equation:

$${}^{(k+1)}\boldsymbol{x}_i = \begin{cases} {}^{(k+1)}\boldsymbol{d} + {}^{(k+1)}\beta \left| {}^{k}\boldsymbol{x}^{Mb} - {}^{k}\boldsymbol{x}_i \right| \ln(1/{}^{(k+1)}u) \text{ if } {}^{(k+1)}\kappa \geq 0.5 \\ {}^{(k+1)}\boldsymbol{d} - {}^{(k+1)}\beta \left| {}^{k}\boldsymbol{x}^{Mb} - {}^{k}\boldsymbol{x}_i \right| \ln(1/{}^{(k+1)}u) \text{ otherwise} \end{cases}, \tag{16}$$

see, for instance, [83,201]. Herein, ${}^{(k+1)}\boldsymbol{d} = {}^{(k+1)}\psi {}^{k}\boldsymbol{x}_i^{Pb} + \left(1 - {}^{(k+1)}\psi\right) {}^{k}\boldsymbol{x}^{Gb}$, ${}^{(k+1)}\psi$, ${}^{(k+1)}u$ and ${}^{(k+1)}\kappa$ are uniform random numbers between 0 and 1 freshly generated at the iteration $(k+1)$, and ${}^{k}\boldsymbol{x}^{Mb}$ is defined as the mean of the *pbest* positions of all particles (also known as *mbest*). Note that ${}^{(k+1)}\beta$ is the only control parameter in the QPSO algorithm (it is also named contraction–expansion coefficient). Liu et al. [201] proposed a variant of the QPSO algorithm in which the mutation operator also takes place: in their final algorithm, it is assumed that the contraction–expansion coefficient varies linearly between 1.0 and 0.5 during the iterative procedure.

This is a short introduction about PSO and its main variants mostly adopted in nonlinear dynamic systems identification, but the interested reader can refer to [19,34,296,364] for a broader presentation. It is worth noting that important theoretical results have been also achieved about the analysis of the algorithm stability, convergence, and performance, see, for instance, [35,161,183,317,342].

2.3.2 Applications

The PSO and its variants or hybridizations have been employed frequently for the parametric identification of the Bouc–Wen model. For instance, numerical results by Ye and Wang [348] demonstrate that the standard PSO can efficiently identify the Bouc–Wen model parameters even in case of high noise levels. Kao and Fung [162] identified the parameters of a Bouc–Wen model employed to describe the hysteretic behavior of a piezoelectric device that actuates a Scott–Russell mechanism. A core-free dielectric electroactive polymer tubular actuator was considered by Sarban et al. [289]. Their experimental setup makes use of an actuator made of silicone-based elastomer combined with a smart compliant electrode, and its overall hysteretic behavior is described by the Bouc–Wen model whose parameters are identified by means of a standard PSO.

The modified PSO algorithm proposed by Charalampakis and Dimou [53] implements some interesting novelties, namely (i) should the objective function not decrease for a certain number of iterations, then inertia factor and maximum velocity are decreased; (ii) a craziness operator is introduced in such a way that a random velocity vector is assigned to an individual outside the search space; (iii) the individual with the worst performance is moved to the best ever position of the swarm; (iv) a different rule to update the particle position applies when its velocity results in an improvement in *gbest*. Charalampakis and Dimou [54] considered this enhanced PSO algorithm together with other computational intelligence techniques for the parametric identification of a Bouc–Wen model representing the non-degrading hysteretic behavior of a bolted-welded steel connection. In so doing, they found that its performance is satisfactory, but the accuracy is influenced by initial settings and typically lower than that of the standard DE.

Talatahari et al. [314] adopted the PSO algorithm hybridized through the Big Bang–Big Crunch algorithm [164] to identify a magneto-rheological damper using a modified Bouc–Wen model and numerically simulated data. The hybrid algorithm proposed in [180, 181] merges the features of PSO, Flock-of-Starlings Optimization [107], and Bacterial Chemotaxis Algorithm [236] in the attempt of benefiting from the good exploration ability of the first two techniques while improving the convergence speed by means of the third one. A useful conclusion drawn by Laudani et al. [180] is that the Bouc–Wen model can simulate the hysteresis loops in ferromagnetic materials in place of the Jiles–Atherton model for low-frequency excitations only, while it is unfit to predict iron losses (dynamic effects) and the minor symmetric loops even after a careful parameters estimation. Conversely, they found that the Jiles–Atherton is able to reproduce dynamic and minor loops once its parameters are properly identified. Qin et al. [271] revised the Bouc–Wen model to account for an asymmetric hysteresis. This model was then employed to mimic the response of a piezoelectric ceramic actuator, and its parameters were identified by means of the standard PSO algorithm using experimental data.

The standard PSO algorithm was adopted by Quaranta et al. [274] to identify the parameters of a Bouc–Wen model with the main goal of describing the experimental response of a high damping rubber bearing for seismic isolation under different axial load levels and horizontal displacement amplitudes. Carboni et al. [46] proposed a nonlinear rod model incorporating a hysteretic moment-curvature law described by the Bouc–Wen model for the identification of the force-displacement loops of steel wire ropes of different configurations and lengths. The parameters regulating the hysteretic moment-curvature law and a parameter regulating the initial bending stiffness of the ropes were identified using the PSO algorithm by best fitting the experimentally obtained curves for 7 × 19 and 7 × 7 wire ropes of length equal to 75 mm. The identified parameters were employed to extrapolate the hysteresis curves obtained for different lengths and displacement amplitudes of the ropes. A general close agreement with the experimental results was obtained. The work done by Yu et al. [352] is concerned with the numerical modeling and parametric identification of a magneto-rheological elastomer base isolator. To this end, the authors formulated a novel strain stiffening model that implements a smaller number of parameters as compared with the standard Bouc–Wen model. They also adopted a variant of the PSO algorithm for the parametric identification in which a self-adaptive inertia weight is implemented. Results based on experimental data reveled that this novel strain stiffening model is as accurate as the standard Bouc–Wen model despite the reduced number of model parameters, which was found useful in mitigating the overall computational effort required by parametric identification process. On the other hand, Wang et al. [324] considered a piezoelectric actuator whose output displacement is expressed by two terms function of the input voltage: while the first one is here defined by means of the standard Bouc–Wen model of hysteresis, the second term is a two-degree polynomial function introduced to account for the asymmetric hysteresis of the piezoelectric actuator. Both PSO and DE were considered to solve the parameters identification problem. Recently, Barbieri et al. [20] have employed the Bouc–Wen model to reproduce the hysteretic behavior of two devices for vibration control (i.e., wire rope isolator and Stockbridge damper) and its parameters were identified from the experimentally recorded response using the standard PSO algorithm.

One of the first applications of the PSO algorithm for the parametric identification of chaotic systems is due to He et al. [133]. Begambre and Laier [23] hybridized the PSO algorithm with the Nelder–Mead simplex search and applied the resulting algorithm to the identification of the Duffing oscillator. In practice, while the simplex method looks for improved values of inertia weight and acceleration factors, the PSO algorithm takes these values to minimize the objective function. Gao et al. [112] implemented a PSO for the parametric identification of chaotic systems without and with time delay in the presence of random initial noises. The version herein adopted is the one based on the constriction factor and also includes the inertia weight, which is adjusted during the iterative process.

Yang et al. [344] implemented the QPSO algorithm for the parametric identification of several chaotic systems. In their work, the authors proposed a dynamic version of the contraction–expansion coefficient, which is adapted according to the objective function value of the particles. New nonlinear dynamic models have been proposed in [233] for updating the inertia weight and acceleration factors. While the most common dynamic model for these parameters employs a simple linear variation that depends on the iteration number only, Modares et al. [233] proposed exponential functions based on the objective function of the current global optimum.

Quaranta et al. [272] compared different versions of the PSO for the parametric identification of Duffing–van der Pol oscillators, each of them based on a different search strategy, namely (i) linearly varying inertia weight and acceleration factors; (ii) constriction factor; (iii) chaotic control parameters (i.e., CPSO); iv) passive congregation [134]. The variant of the PSO algorithm proposed by Sun et al. [312] to identify chaotic systems exploits the analogy with the motion of electrons in metal conductors in an electric field. With respect to the standard velocity updating rule, the term based on the current particle velocity is here replaced with a new one that depends on *mbest* and the contraction–expansion coefficient. The standard PSO algorithm has been considered in [186] to cope with the parametric identification of the Lorenz system within a comparative assessment of a new chaotic gravitational search algorithm.

The PSO-based identification of fractional-order chaotic systems without and with time delay was addressed in [354] and [353], respectively. Banerjee and Abu-Mahfouz [18] performed a useful and interesting comprehensive study concerning the identification of several chaotic systems. As regards the identification of van der Pol–Duffing oscillator, the authors examined the search space corresponding to the related parametric identification problem, and they found that it well agrees with the results obtained previously by Quaranta et al. [272]. Further chaotic systems considered in [18] are Chen, Lorenz, Lü, and Maxwell–Blouch systems. Among the considered versions of the PSO algorithm, the study by Banerjee and Abu-Mahfouz [18] encompasses PSO algorithms with linearly and nonlinearly varying control parameters as well as the PSO algorithm including the passive congregation term.

Li and Yin [191] considered the standard PSO algorithm for a comparative evaluation of a novel hybrid technique that combines the standard DE algorithm and the ABC algorithm. Jiang et al. [159] proposed a dual PSO algorithm for the parametric identification of chaotic systems. Therein, the variant of the PSO algorithm based on the constriction factor was employed, with the exception that velocity and position of *gbest* are updated according to the so-called niche PSO algorithm (in which the inertia weight is equal to 1 while the cognitive parameter is taken equal to 1.494 and the social cooperation is removed by setting to zero the social parameter).

Prawin et al. [269] proposed a hybrid dynamic QPSO to solve parametric identification problems for nonlinear dynamic systems. The main novelties in this version of the QPSO algorithm are the following: (i) the whole swarm is divided into sub-swarms, each of them using its own agents to look for the optimum but able to exchange information with others through a periodic shuffle; (2) a periodic local search based on the Nelder–Mead method. As regards the parametric identification of the Dufffing oscillator, however, the limited comparative study presented in [269] fails to demonstrate that this novel QPSO algorithm is significantly better than the standard PSO despite its larger complexity. Another recent application of the QPSO algorithm for the identification of chaotic phenomena has been reported by Wei et al. [330], where the identification of uncertain incommensurate fractional-order chaotic systems is addressed.

Yuan et al. [355] developed a novel variant of the chaos optimization algorithm, whose performance was compared to that of some well-established computational intelligence techniques, including the standard PSO algorithm. The identification of the Lorenz system was performed in [360] by means of a hybridized version of the teaching-learning-based optimization, and its performance was compared to that of many computational intelligence techniques, including a standard version of the PSO algorithm. Moving from the positive but rather counterintuitive results obtained previously by Zheng et al. [367], Li et al. [192] considered a PSO algorithm with an increasing inertia weight. A theoretical study was also carried out to find the numerical values of the control parameters that guarantee the global convergence of the algorithm, which is applied to identify a fractional-order chaotic system with time delay. In passing, it is pointed out that there are also a few studies about the identification of hyperchaotic phenomena based on swarm intelligence: Hu et al. [144], for instance, apply the ABC algorithm for the parametric identification of fractional-order hyperchaotic systems.

A few studies exist about the parametric identification of the Jiles–Atherton model via PSO, see for instance [68,180,219]. In particular, the variant of the PSO algorithm proposed by Marion et al. [219] incorporates some significant novelties. First, the authors calculate two objective functions, namely (i) squared error between the measured values of the major hysteresis loop and the calculated ones; (ii) area error per cycle by comparing experimental data and numerical predictions. Based on these optimum criteria, the authors revised the definition of best particle into the swarm, in such a way that each particle has its own *gbest*, the latter depending on its position within the decisions space.

The parameters of the LuGre model were identified using the PSO algorithm by Wenjing [331] and Irakoze et al. [152]. Specifically, a comparison between GA and PSO was presented by Irakoze et al. [152], who found that the former is more accurate, whereas the latter exhibits a greater robustness. A significant number of studies exist about the identification of the parameters of the Prandtl–Ishlinskii model. Yang et al. [345] performed the identification of the inverse generalized Prandtl–Ishlinskii model for the compensation of the asymmetric hysteresis in piezoelectric actuators. A modified PSO was adopted to this end, namely a PSO with cyclic neighborhood topology [220]. Rahman et al. [275], Ko et al. [168], and Long et al. [205] also addressed the identification-based construction of the inverse generalized Prandtl–Ishlinskii model for hysteresis compensation problems in piezoelectric actuators. In particular, Rahman et al. [275] adopted a standard PSO algorithm, a CPSO was employed by Long et al. [205], whereas the cyclic network topology-based PSO was exploited in [168].

Gu et al. [123] developed a model of piezoelectric-actuated stages that accounts for hysteresis, linear time-invariant dynamics, and time delay. Once the linear dynamic model is identified, the parameters of a modified Prandtl–Ishlinskii hysteresis model are estimated by means of the PSO. Xie et al. [341] presented a similar study, i.e., the identification of a linear subsystem preceded by a hysteretic input based on the Prandtl–Ishlinskii model. Herein, the authors proposed new rules to update inertia weight and acceleration factors in the PSO algorithm employed for the parametric identification of the Prandtl–Ishlinskii model. In detail, they assumed that the better the objective function associated with a certain particle, the larger the corresponding inertia weight, and vice versa. On the other hand, the acceleration factors were adjusted according to the variation in the inertia weight.

A nonlinear model based on the combination of Preisach operator and fuzzy nonlinear autoregressive exogenous (NARX) model was considered in [240, 318] to mimic the hysteresis response in piezoelectric actuators, and its unknown parameters were identified through the PSO algorithm. Liu et al. [202] successfully identified the parameters of Wiener and Hammerstein models using PSO and QPSO. A hybrid version of the PSO algorithm was proposed in [189] to identify a Hammerstein model. In this case, the PSO algorithm (employing the constriction factor) is hybridized by means of the SA algorithm. Further recent applications of the PSO in the identification of Wiener–Hammerstein models are presented in [37,239,311]. Particularly, the study by Boubaker [37] deals with the identification of the Hammerstein model consisting of a memory-less polynomial block cascaded to an autoregressive with exogenous input (ARX) time series block.

To this end, the author implemented a mixed integer real-coded PSO algorithm to identify, both, the integer orders and the real coefficients of the model.

For a comprehensive overview about the use of PSO algorithms in nonlinear dynamic parametric identification problems, it is possible to refer to the list given in Table 2.

2.4 Differential evolution

2.4.1 Algorithms

The DE algorithm is an effective—yet simple and robust— nature-inspired strategy proposed by Storn and Price [310] to solve hard optimization problems. It aims at finding the global optimum through an iterative procedure in which a collection of L candidate solutions (also named individuals) ${}^{k}\boldsymbol{x}_i = \left\{{}^{k}x_{i1} \dots {}^{k}x_{ij} \dots {}^{k}x_{iN}\right\}$ (with $i = 1, \dots, L$) is manipulated at each iteration k using suitable operators, namely mutation, crossover, and selection. This set of candidate solutions ${}^{k}\mathcal{P}$ is also named population, and L is the population size.

According to the standard DE procedure, an initial population of L individuals is generated by sampling randomly the bounded search space of the model parameters with a uniform probability distribution function. At each iteration $(k+1)$ (also named generation), a tentative new solution ${}^{(k+1)}\boldsymbol{z}_i$ (also named provisional offspring) is generated through the mutation operator. It works by perturbing a candidate solution through the differences between couples of selected individuals. Some of the most common mutation operators are the following:

$$ {}^{(k+1)}\boldsymbol{z}_i = {}^{k}\boldsymbol{x}_{l_1} + F({}^{k}\boldsymbol{x}_{l_2} - {}^{k}\boldsymbol{x}_{l_3}), \tag{17a} $$

$$ {}^{(k+1)}\boldsymbol{z}_i = {}^{k}\boldsymbol{x}_{best} + F({}^{k}\boldsymbol{x}_{l_1} - {}^{k}\boldsymbol{x}_{l_2}), \tag{17b} $$

$$ {}^{(k+1)}\boldsymbol{z}_i = {}^{k}\boldsymbol{x}_i + F({}^{k}\boldsymbol{x}_{best} - {}^{k}\boldsymbol{x}_i) + F({}^{k}\boldsymbol{x}_{l_1} - {}^{k}\boldsymbol{x}_{l_2}). \tag{17c} $$

$$ \begin{aligned} {}^{(k+1)}\boldsymbol{z}_i &= {}^{k}\boldsymbol{x}_{best} + F({}^{k}\boldsymbol{x}_{l_1} - {}^{k}\boldsymbol{x}_{l_2}) \\ &\quad + F({}^{k}\boldsymbol{x}_{l_3} - {}^{k}\boldsymbol{x}_{l_4}), \end{aligned} \tag{17d} $$

$$ \begin{aligned} {}^{(k+1)}\boldsymbol{z}_i &= {}^{k}\boldsymbol{x}_{l_1} + F({}^{k}\boldsymbol{x}_{l_2} - {}^{k}\boldsymbol{x}_{l_3}) \\ &\quad + F({}^{k}\boldsymbol{x}_{l_4} - {}^{k}\boldsymbol{x}_{l_5}), \end{aligned} \tag{17e} $$

$$ \begin{aligned} {}^{(k+1)}\boldsymbol{z}_i &= {}^{k}\boldsymbol{x}_{l_1} + F({}^{k}\boldsymbol{x}_{best} - {}^{k}\boldsymbol{x}_i) \\ &\quad + F({}^{k}\boldsymbol{x}_{l_2} - {}^{k}\boldsymbol{x}_{l_3}) + F({}^{k}\boldsymbol{x}_{l_4} - {}^{k}\boldsymbol{x}_{l_5}), \end{aligned} \tag{17f} $$

see, for instance, [244,273,294]. Such versions of the mutation operator are labeled as "rand/1," "best/1," "cur-to-best/1," "best/2," "rand/2" and "rand-to-best/2," respectively. In Eq. (17), ${}^{(k+1)}\boldsymbol{z}_i = \left\{{}^{(k+1)}z_{i1} \dots {}^{(k+1)}z_{ij} \dots {}^{(k+1)}z_{iN}\right\}$ is the mutant vector, $l_1, \dots, l_5$ are integers randomly generated between 1 and L such that distinct individuals from the population at the iteration k are involved in each mutation operator, ${}^{k}\boldsymbol{x}_{best}$ is the best individual in the current population at the iteration k, F is the mutation (or scale) factor. A projection scheme is commonly implemented in order to ensure that ${}^{(k+1)}z_{ij}$ satisfies all boundary constraints. In its simplest version, it reads:

$$ {}^{(k+1)}z_{ij} = \begin{cases} x_j^{\ell} & \text{if } {}^{(k+1)}z_{ij} < x_j^{\ell} \\ x_j^{u} & \text{if } {}^{(k+1)}z_{ij} > x_j^{u} \\ {}^{(k+1)}z_{ij} & \text{otherwise} \end{cases} . \tag{18} $$

When the provisional offspring has been generated by one of the mutation operator and boundary violations are fixed, ${}^{(k+1)}z_{ij}$ is exchanged with the corresponding ${}^{k}x_{ij}$ with a uniform probability, thereby obtaining the final offspring (also named trial vector) ${}^{(k+1)}\bar{x}_{ij}$ for $j = 1, \dots, N$. To this end, the binomial crossover is typically employed. It reads:

$$ {}^{(k+1)}\bar{x}_{ij} = \begin{cases} {}^{(k+1)}z_{ij} & \text{if } {}^{(k+1)}r \le CR \vee j = {}^{(k+1)}j_{rand} \\ {}^{k}x_{ij} & \text{otherwise} \end{cases} \tag{19} $$

where ${}^{(k+1)}r$ is a uniformly distributed random number between 0 and 1, ${}^{(k+1)}j_{rand}$ is a randomly selected integer between 1 and N, CR is the crossover rate. Note that the additional condition $j = {}^{(k+1)}j_{rand}$ is introduced to ensure that ${}^{k}\boldsymbol{x}_i$ and ${}^{(k+1)}\bar{\boldsymbol{x}}_i$ differ for at least one component j. The crossover operator in Eq. (19) is labeled as "bin." For instance, a DE algorithm that implements Eq. (17a) as mutation operator and Eq. (19) as crossover operator will be labeled as "DE/rand/1/bin."

The objective function for ${}^{(k+1)}\bar{\boldsymbol{x}}_i$ is then evaluated and, according to a one-to-one spawning strategy, it replaces ${}^{k}\boldsymbol{x}_i$ (i.e., ${}^{(k+1)}\boldsymbol{x}_i = {}^{(k+1)}\bar{\boldsymbol{x}}_i$) if and only if $f\left({}^{(k+1)}\bar{\boldsymbol{x}}_i\right) \le f\left({}^{k}\boldsymbol{x}_i\right)$, otherwise no replacement occurs. By performing such procedure for all candidate solutions, a new population is available for the next iteration. Once a suitable stopping criterion is fulfilled (e.g., when the maximum number of objective function evaluations is achieved or the algorithm does

Table 2 Parametric identification applications using PSO

Nonlinear model	References	Applications
Bouc–Wen	Ye and Wang [348]	Numerical simulations using standard PSO
	Kao and Fung [162]	Piezoelectric actuator in Scott–Russell mechanism
	Sarban et al. [289]	Electroactive polymer actuator
	Charalampakis and Dimou [53]	Bolted-welded steel connection
	Charalampakis and Dimou [54]	Bolted-welded steel connection
	Talatahari et al. [314]	Magneto-rheological damper
	Laudani et al. [180]	Hysteresis loops in ferromagnetic materials
	Laudani et al. [181]	Numerical simulations using hybrid PSO
	Qin et al. [271]	Piezoelectric ceramic actuator
	Quaranta et al. [274]	High damping rubber bearing
	Carboni et al. [46]	Steel wire ropes
	Wang et al. [324]	Piezoelectric actuator
	Yu et al. [352]	Magneto-rheological elastomer base isolator
	Barbieri et al. [20]	Wire rope isolator, Stockbridge damper
Chaotic system	He et al. [133]	Lorenz system
	Begambre and Laier [23]	Duffing system
	Gao et al. [112]	Chen, Lorenz, Lü and logistic systems
	Yang et al. [344]	Chen, Duffing, Hénon, Lorenz, Nagumo–Sato, Rössler systems
	Modares et al. [233]	Lorenz system
	Quaranta et al. [272]	Duffing–van der Pol systems
	Sun et al. [312]	Chen and Lorenz systems
	Li et al. [186]	Lorenz system
	Yuan and Yang [354]	Fractional-order Chen and Lorenz systems
	Yuan et al. [353]	Fractional-order chaotic (logistic and Chen) delayed systems
	Banerjee and Abu-Mahfouz[18]	Duffing/Van der Pol, Chen, Lorenz, Lü and Maxwell–Blouch systems
	Li and Yin[191]	Chen and Lorenz systems
	Jiang et al. [159]	Lorenz system
	Prawin et al. [269]	Duffing system
	Yuan et al. [355]	Chen, Lorenz and Lü systems
	Zhang et al. [360]	Lorenz system
	Li et al. [192]	Time-delayed fractional Mackey–Glass system
	Wei et al. [330]	Fractional-order chaotic systems
Jiles–Atherton	Marion et al. [219]	Hysteresis loops of magnetic materials
	Coelho et al. [68]	Hysteresis loops of magnetic materials
	Laudani et al. [180]	Hysteresis loops in ferromagnetic materials
LuGre	Wenjing [331]	Frictional servo-systems
	Irakoze et al. [152]	Piezoelectric actuator
Prandtl–Ishlinskii	Yang et al. [345]	Piezoelectric actuator
	Rahman et al. [275]	Piezoelectric actuator
	Gu et al. [123]	Piezoelectric-actuated system
	Ko et al. [168]	Piezoelectric actuator
	Long et al. [205]	Piezoelectric actuator
	Xie et al. [341]	Piezoelectric actuator

Table 2 continued

Nonlinear model	Reference	Applications
Preisach	Nam and Ahn [240]	Ionic polymer metal composite actuator
	Truong et al. [318]	Electroactive polymer actuator
Wiener–Hammerstein	Li et al. [189]	Numerical simulations using hybrid PSO
	Liu et al. [202]	Numerical simulations using QPSO
	Sun and Liu [311]	Numerical simulations
	Naitali and Giri [239]	Simulation of electronic device including nonlinear passive circuit
	Boubaker [37]	Numerical simulations

not success in improving the objective function value after a given number of iterations), then the iterative procedure is interrupted.

For programming the standard DE algorithm, it might be useful to refer to the corresponding pseudocode provided in Algorithm 3.

Algorithm 3 Pseudocode of the standard DE algorithm

```
Require: Problem data
Require: Control parameters of the optimizer
  Set $k = 1$
  Initialize the population ${}^{k}\mathcal{P} = \left\{{}^{k}\boldsymbol{x}_1, \dots, {}^{k}\boldsymbol{x}_i, \dots, {}^{k}\boldsymbol{x}_L\right\}$
  for $i \leftarrow 1, L$ do
    Calculate $f\left({}^{k}\boldsymbol{x}_i\right)$
  end for
  Set to "false" the status of the stopping criterion
  while stopping criterion = "false" do
    for $i \leftarrow 1, L$ do
      Apply the mutation operator, thereby obtaining ${}^{(k+1)}\boldsymbol{z}_i$
      Ensure that ${}^{(k+1)}\boldsymbol{z}_i$ does not violate boundary constraints
      Apply the crossover operator, thereby obtaining ${}^{(k+1)}\bar{\boldsymbol{x}}_i$
      if $f\left({}^{(k+1)}\bar{\boldsymbol{x}}_i\right) \leq f\left({}^{k}\boldsymbol{x}_i\right)$ then
        Assume ${}^{(k+1)}\boldsymbol{x}_i = {}^{(k+1)}\bar{\boldsymbol{x}}_i$
      else
        Assume ${}^{(k+1)}\boldsymbol{x}_i = {}^{(k)}\boldsymbol{x}_i$
      end if
    end for
    Move the population ${}^{(k+1)}\mathcal{P}$ to the next iteration
    Set $k = k + 1$
    Decide whether the status of the stopping criterion switches
    from "false" to "true"
  end while
  return Carry out best solution and corresponding objective
  function value
```

The main issues related to the application of the DE algorithm in its standard version are associated with some important choices left to the user. As usual in all population-based optimization algorithms, the population size L should be not too small in order to avoid premature convergence, but it should be not too large to prevent stagnation and to save computational time. For a rule of thumb, it might be possible to refer to [310], in which the population size is assumed equal to ten times the dimensionality of the search space (i.e., $L = 10N$).

The interested reader can refer to [287,349] for a recent review about variants of the standard DE algorithm able to operate with a very small population size (so-called micro DE algorithms, μDE). Some existing variants of the standard DE algorithm also investigate the possibility of adapting dynamically the population size, see, for instance, [325]. The selection of the best numerical value for F and CR is also a decisive and critical task. In the standard DE algorithm, these are set to be constant. According to Storn and Price [310] and Liu [200], it is suggested $F \in [0.5, 1]$ and $CR \in [0.8, 1]$. Note, however, that numerical values close to the upper bound are not recommended in many studies, and thus, the possible range of values is rather narrow.

Since algorithm settings largely depend on the problem at hand and might have a significant impact on the final result, several variants have been developed in the last years in an attempt to enhance the robustness of the standard algorithm. Among the existing proposals, those that attracted most attention so far are the self-adaptive DE (SaDE) proposed by Huang et al. [148] and the adaptive DE with optional external archive (JADE) developed by Zhang and Sanderson [361]. Both have been adopted in some parametric identification problems, and thus, they are briefly examined hereafter.

SaDE implements four mutation operators (three of them undergo the classical crossover operator while the last one does not). A roulette wheel selection is employed to select the mutation strategy for each individual in the current population. At the beginning of the evolutionary search, the probability of applying the four mutation operators is the same. These probability values are then updated throughout the iterative procedure taking into account the number of trial vectors surviving the selection operator and that of discarded trail vectors for a given mutation strategy. The number of surviving and discarded trial vectors is evaluated over a given number of past generations also known as learning period, which is assumed equal to 20 iterations.

As regards the control parameters, Huang et al. [148] proposed different values of scale factor and crossover rate in the SaDE algorithm for different individuals in the current population at each iteration. Specifically, scale factor values are generated from a normal distribution with mean 0.5 and standard deviation 0.3, within the range (0, 2]. On the other hand, crossover rate values are uniformly distributed random numbers. They are generated assuming that the mean of the crossover rate is an adaptive parameter (at the beginning, it is equal to 0.5) while the standard deviation is a constant (equal to 0.1). These random values of the crossover rate are kept constant for 5 generations. Subsequently, a new set of values is generated. The mean of the crossover rate is recalculated every 20 generations as the average of the values corresponding to successful trial vectors generated during this learning period. Finally, SaDE also applies a local optimizer (Sequential Quadratic Programming technique) to some good individuals every 500 generations.

The modified DE algorithm proposed by Zhang and Sanderson [361] can employ two alternative versions of a new mutation operator named “cur-to-$p$best/1,” which are a revised version of the “cur-to-best/1” in Eq. (17c). In both versions, ${}^k\boldsymbol{x}_{best}$ in Eq. (17c) is replaced with ${}^k\boldsymbol{x}^p_{best}$, which is an individual randomly chosen as one of the top $100p\%$ individuals in the current population, with $p \in (0, 1]$. These versions differ in one aspect only, which is the use of an external repository of archived inferior individuals. This archive is initiated to be empty. After each generation, the individuals that fail in the selection process are added to the archive. If the archive size exceeds a certain threshold, then some solutions are randomly removed from the archive to keep the archive size constant. Should this archive be employed during the iterative process, “cur-to-$p$best/1” and “cur-to-best/1” in Eq. (17c) also differ for the fact that ${}^k\boldsymbol{x}_{l_2}$ is randomly chosen from the union of the current population and that archive.

As in SaDE, JADE implements different random values of scale factor and crossover rate for different individuals. Specifically, scale factor and crossover rate are randomly generated taking into account the set of all successful mutation factors and the set of all successful crossover probabilities, respectively. More recently, Islam et al. [153] proposed new strategies (i.e., a new mutation operator, a fitness-induced parent selection scheme for the binomial crossover, novel rules to update scale factor, and crossover rate) and shown that JADE can achieve superior performance if one or all of them are implemented.

On the other hand, among the most recent variants of the DE algorithm, it is worthy highlighting the recent proposal by Formica and Milicchio [103]. The main novelty in this variant is attributable to the way by which the new population is carried out, which also accounts for the ancestry of the population in the crossover phase. Additionally, differently from SaDE and JADE, chaotic maps are employed in the mutation step instead of random numbers.

For a more detailed review about recent trends and advances in DE algorithms, the interested reader can refer to [6,43,73,74,244,338]. It is also worth highlighting that some attempts have been made recently to provide a rigorous mathematical framework for DE algorithms. A useful review of the most recent efforts in this field is given by Opara and Arabas [249].

2.4.2 Applications

Kyprianou et al. [174] are among the first to propose the use of DE for parametric identification of nonlinear dynamic systems. Herein, the authors considered an experimental case study that deals with the dynamic response of the valve in a nuclear power plant obtained from an earthquake resistance test. The experimental data consist of the low-frequency seismic acceleration input and the relative response acceleration of the valve to the base, which is modeled as single-degree-of-freedom (SDOF) system with Bouc–Wen type hysteresis.

The identification of wood members joined by plywood gusset plate subjected to cyclic loading is addressed in [209] as well as in [5]. To this end, the Bouc–Wen model revised by Foliente [102] was considered in order to take into account strength degradation, stiffness degradation, and pinching. A subset of the model parameters was successfully identified by means of the standard DE, and the resulting model was shown to reproduce satisfactorily new experimental data other than the ones used for the parametric identification.

Extensive numerical simulations were presented by Worden and Manson [334] to shed light on the parametric identification of the Bouc–Wen model through the solution of an optimization problem. Specifically, the authors focused on the shape of the search space, being the objective function defined in terms of normalized mean square error between measured data and predicted values. It is therein demonstrated that the low sensitivity of the objective function with respect to some model parameters makes shallow the region that includes the optimum, thereby complicating their identification. A further issue here highlighted is associated with the fact that the search space appears noisy even if measured data are simulated without adding noise. This is because small variations in the point at which the dynamic solver meets the error criterion lead to small variations in the final predictions. As regards the numerical approach employed for the solution of the optimization problem, the standard DE is initially considered, but it is shown that the SaDE gives improved parameter estimates.

Charalampakis and Dimou [54] performed an extensive comparative assessment of different computational intelligent techniques for the parametric identification of the standard Bouc–Wen model. The authors addressed the identification of a bolted-welded steel connection under cyclic loading and also developed a variant of the DE that utilizes base vectors stochastically chosen to be either a random vector of the population or the currently best vector. This comparative evaluation in [54] indicates that DE algorithms are most effective in the parametric identification of the Bouc–Wen model.

Carboni et al. [45] proposed a modified version of the Bouc–Wen model in the attempt to describe the dynamic response of an innovative rheological device made of assemblies of NiTiNOL strands, wires and steel wire ropes. The parameters of such novel version of the Bouc–Wen model were identified through a standard DE algorithm. Quaranta et al. [274] identified the parameters of a standard Bouc–Wen model to reproduce the response of high damping rubber bearings used for seismic isolation of buildings. The experimental data adopted in the study came from code conforming full-scale laboratory tests for various amplitudes of the horizontal cyclic loads and different levels of the axial force.

A modified Bouc–Wen model is introduced in Wang et al. [324] to describe the asymmetric hysteresis of piezoelectric actuators, whose parameters were identified using experimental data and a novel variant of the DE algorithm proposed by the authors. The problem addressed by Zaman and Sikder [358] is concerned with the parametric identification of a standard Bouc–Wen model intended to replicate the observed hysteresis in the relationship between magnetization and magnetic field for amorphous materials. Among the numerical techniques adopted to calibrate the model parameters, Zaman and Sikder [358] also employed a standard version of the DE algorithm.

In the context of a wide validation study for the vibration absorber made of the rheological device presented in [45], Carboni and Lacarbonara [44] again employed the standard version of the DE algorithm to identify the parameters of the proposed variant of the Bouc–Wen model. Carboni et al. [47] further developed this modified version of the standard Bouc–Wen model in order to also account for the asymmetry occurring in the response exhibited by such novel device both in terms of stiffness and energy dissipation under severe dynamic loading. The parameters of this new hysteretic model were calibrated via standard DE algorithm.

Worden et al. [337] discussed the application of the JADE for the parametric identification of the standard Bouc–Wen model by using simulated data. Formica et al. [104] proposed a phenomenological smooth hysteresis model that mimic the Bouc–Wen law for high fidelity simulations of the hysteresis due to stick-slip energy dissipation in carbon nanotube/polymer nanocomposites. The parameters of the resulting model were identified by means of the DE algorithm and using the experimental displacement–force relationships measured for nanocomposites beam samples made of polybutylene terephthalate polymer embedding carbon nanotubes.

Positive results were also obtained from the application of the DE algorithm in the parametric identification of chaotic systems since the first studies by [52,260,344]. For instance, Ho et al. [139] applied a modified DE algorithm to identify the parameters of several chaotic systems. The main novelty in this variant of the standard DE algorithm deals with the crossover operator, which is based on the Taguchi-sliding-level method [319].

Quaranta et al. [272] performed an extensive numerical study about the parametric identification of van der Pol–Duffing oscillators, from which it was found that DE generally performs better than PSO. Another hybridized DE algorithm is proposed in [326] for parametric identification of chaotic systems. Herein, at each generation, the population undergoes classical DE operators and Nelder–Mead simplex search. The parameters of commensurate fractional-order chaotic systems were identified by Tang et al. [315] using a standard DE algorithm, and their results highlight that such a computational strategy is more robust than GA, especially when the fractional order itself is an unknown parameter to be estimated.

An extensive and rigorous statistical comparative assessment of PSO and DE algorithms in the parametric identification of chaotic systems is provided by Banerjee and Abu-Mahfouz [18]. As far as the DE algorithms are concerned, the authors considered three alternative mutation operators, and the scale factor F was assumed to vary linearly between 1 and 0.5, in such a way to facilitate the exploration of the search space at the beginning and the exploitation of the best solution at the end of the evolutionary search. Gao et al. [115] applied a standard DE for the identification of time-delayed fractional chaotic systems with and without noise.

Li and Yin [191] performed parametric identification of chaotic systems by means of a hybrid method that combines the standard DE algorithm and the ABC algorithm. However, it seems that the larger complexity of the hybridized DE algorithm adopted in [191] is not beneficial, since the final estimates of the parameters of the considered chaotic systems are the same as those that would be obtained using a standard DE algorithm. A similar conclusion can also be drawn from the study presented by Gu et al. [124], who applied again the DE algorithm hybridized by means of the ABC algorithm, in this case to estimate the parameters of fractional-order memristor-based chaotic systems. Note that an opposite strategy has been pursued in a previous study by Gao et al. [113], i.e., DE operators are introduced in the classical ABC algorithm in the attempt to enhance its performance in the parametric identification of van Der Pol–Duffing systems. Erdbrink and Krzhizhanovskaya [99] adopted a special version of DE [64] for the identification of several nonlinear dynamic systems where all candidate solutions have their own control parameters for mutation and crossover operators, which are updated taking into account the average value of successfully evolved individuals' parameter values by using the Cauchy distribution. The parameters of a nonlinear damping model as well as those in van Der Pol and Mathieu systems have been identified by Erdbrink and Krzhizhanovskaya [99] using such recent self-adaptive variant of the DE algorithm.

Yuan et al. [355] compared a novel variant of the chaos optimization algorithm with the standard DE algorithm. The identification of Chen, Lorenz, and Lü systems showed that their proposal slightly outperforms the standard DE algorithm, whose reliability in chaotic systems identification is, however, confirmed again. Recently, Liu et al. [199] have introduced a multi-selection operator in the classical DE algorithm, which employs three candidate solutions within the selection operator instead of two as in the standard version (the new candidate is generated from an alternative mutation operator and is not subjected to the crossover operator). This new version of the standard DE algorithm is then applied to identify the time-delayed fractional-order Mackey–Glass chaotic system. Although no extensive results are reported in this study, it seems that a higher convergence speed can be obtained without a significant increment of the algorithm complexity. Another application of the DE algorithm for the parametric identification of the time-delayed fractional-order Mackey–Glass chaotic system has been presented recently by Li et al. [192].

In a couple of papers, Du et al. [93,94] implemented the so-called composite DE algorithm [329] (i.e., a variant of the standard DE algorithm in which several trial vector generation strategies and some suitable control parameter settings are combined) for the parameters estimation of fractional-order chaotic systems. The final results therein reported shown that it outperforms other variants of the standard DE algorithm, including the JADE algorithm, and the accuracy of the estimated parameters is somewhat affected by the kind of noise (specifically, the Lévy noise distribution leads to larger errors than the more conventional Gauss and Cauchy noise distributions). Recently, Hu et al. [145] have implemented the DE algorithm to estimate the parameters of chaotic delayed fractional-order multiagent systems. In passing, it is pointed out that there are also a few studies about the use of DE algorithm for the identification of hyperchaotic systems, see, for instance, [114]. A different evolutionary technique—namely the oppositional backtracking search optimization algorithm—has been also employed in [197] for the identification of hyperchaotic systems.

There are also some significant results dealing with the parametric identification of the Jiles–Atherton model of hysteresis via DE algorithms. As an example, Toman et al. [316] developed a modified version of this phenomenological model, in which the pinning parameter is assumed as function of the applied magnetic field instead of a constant value as in the original formulation. The standard DE algorithm is then employed for the parametric identification of such variant of the Jiles–Atherton model, where the objective function is formulated comparing measured and simulated magnetic flux densities. The identification of a single-phase transformer shows an excellent agreement between measurements and numerical predictions.

dos Santos Coelho et al. [87] illustrated the results obtained in the context of extensive comparative evaluations of several variants of the DE algorithm in which different operators were implemented, whereas the control parameters (mutation factor and/or crossover rate) are set as constant values or uniform random variables. The numerical investigation was performed taking into account experimental data obtained from a workbench containing a rotational single sheet tester and reveals that the best variant employs the best/1/bin layout with constant mutation factor equal to 0.5 and uniformly distributed random crossover rate in the range [0.2, 0.8]. Rather than focusing on the architecture of the DE algorithm, Zhang and Fletcher [359] investigated the role of the objective function formulation in the parametric identification of the Jiles–Atherton model. The authors first observe that the classical formulations based on magnetic field or magnetic flux density at each sample point within one hysteresis loop have some limitations. Therefore, they introduced a new objective function that serves the purpose of finding the minimum of a normalized error that depends on both magnetic field and magnetic flux density (evaluated for six specified critical points). The proposed experimental application is concerned with the identification of a ferrite core via DE algorithm, and the results for excitation frequencies other than those used for the parametric identification demonstrate a good robustness of the procedure.

A modified Jiles–Atherton model was identified in [204] via DE algorithm in order to predict the displacement of giant magnetostrictive actuator working at high frequencies under different axial stress and temperature conditions. This modified model seems to be able to simulate the electrical–magnetic–thermal–mechanical coupling in giant magnetostrictive ferromagnetic materials, but it involves a quite large number of parameters.

Recently, the friction estimation problem in wind turbine blade bearings has been addressed in [309]. In detail, a modified DE algorithm specifically designed for real-time identification of time-varying systems [261] was implemented to estimate the parameters of the LuGre friction model chosen to simulate the blade bearing friction in the pitch system. Zhang and Yan [363] proposed a variant of the classical Prandtl–Ishlinskii model in order to take into account asymmetry and rate-dependent hysteresis in piezoelectric ceramic actuators. An adaptive DE algorithm was developed to identify the parameters of this novel nonlinear dynamic model, in which scale factor and crossover rate evolve according to two smooth functions. Regarding the identification of the Preisach model, an application based on the standard DE algorithm is reported by Szabó and Füzi [313].

Finally, a few applications of the DE algorithm for the parametric identification of the Wiener–Hammerstein model have been presented, see, for instance, [81,227,228,337]. Specifically, Worden et al. [337] adopted the SaDE algorithm to identify the parameters of Wiener–Hammerstein models consisting of linear blocks and nonlinear functions represented by an autoregressive exogenous (ARX) model and sum over sigmoid functions, respectively. The authors highlight a very slow convergence of the algorithm, which in part is attributable to the fact that many unstable linear blocks are generated during the evolutionary search, thereby causing a significant waste of computational time. Thus, they proposed to constrain the evolutionary search in such a way that only stable models are included within the initial population.

An overview of the examined parametric identification problems via DE is proposed in Table 3.

3 Nonparametric identification

3.1 Nonparametric identification problem

Computational intelligence-based nonparametric identification techniques here reviewed are GP and ANN. The nonparametric identification problem consists in the search for at least one mathematical model $M(\boldsymbol{u}) \subset \mathcal{M}$ such that a suitable measure of the distance (or error) $g(\cdot)$ between the predicted nonlinear dynamic system

Table 3 Parametric identification applications using DE

Nonlinear model	Reference	Applications
Bouc–Wen	Kyprianou et al. [174]	Valve of nuclear power plant
	Ma et al. [209]	Wood members joined by plywood gusset plate
	Ajavakom et al. [5]	Wood members joined by plywood gusset plate
	Worden and Manson [334]	Numerical simulations using SaDE
	Charalampakis and Dimou [54]	Bolted-welded steel connection
	Carboni et al. [45]	Shape memory tuned mass damper
	Quaranta et al. [274]	High damping rubber bearing
	Wang et al. [324]	Piezoelectric actuator
	Zaman and Sikder [358]	Magnetic behavior of amorphous material
	Carboni and Lacarbonara [44]	Nonlinear tuned mass damper
	Carboni et al. [47]	Nonlinear tuned mass damper
	Worden et al. [337]	Numerical simulations using JADE
	Formica et al. [104]	Carbon nanotube nanocomposites
Chaotic system	Chang [52]	Rössler system
	Peng et al. [260]	Lorenz system
	Yang et al. [344]	Chen, Duffing, Hénon, Lorenz, Nagumo–Sato, Rössler systems
	Ho et al. [139]	Chen, Lü and Rössler systems
	Quaranta et al. [272]	Duffing–van der Pol systems
	Wang et al. [326]	Chen, Lorenz and Lü systems
	Tang et al. [315]	Fractional-order Lü and Volta systems
	Gao et al. [113]	Duffing–van der Pol systems
	Banerjee and Abu-Mahfouz [18]	Duffing/van der Pol, Chen, Lorenz, Lü and Maxwell–Blouch systems
	Gao et al. [114]	Fractional hyperchaotic systems
	Gao et al. [115]	Fractional-order chaotic (logistic and Chen) delayed systems
	Li and Yin [191]	Chen and Lorenz systems
	Erdbrink and Krzhizhanovskaya [99]	van der Pol system
	Gu et al. [124]	Fractional memristor-based Chua and Lorenz systems
	Yuan et al. [355]	Chen, Lorenz and Lü systems
	Du et al. [93]	Fractional-order chaotic systems
	Liu et al. [199]	Time-delayed fractional Mackey–Glass system
	Du et al. [94]	Fractional-order chaotic systems
	Li et al. [192]	Time-delayed fractional Mackey–Glass system
	Hu et al. [145]	Chaotic delayed fractional-order multi-agent system
Jiles–Atherton	Toman et al. [316]	Single-phase transformer (iron core)
	dos Santos Coelho et al. [87]	Workbench with rotational single sheet tester
	Zhang and Fletcher [359]	Mn-Zn ferrite toroidal core
	Liu et al. [204]	Magnetostrictive actuator
LuGre	Stevanović et al. [309]	Wind turbine blade bearings
Prandtl–Ishlinskii	Zhang and Yan [363]	Piezoelectric nano-manipulator
Preisach	Szabó and Füzi [313]	Hysteresis loops of magnetic material
Wiener–Hammerstein	Dewhirst et al. [81]	Experimental data from the locusts leg control system
	Mehmood et al. [227]	Numerical simulations
	Worden et al. [337]	Numerical simulations using SaDE
	Mehmood et al. [228]	Numerical simulations

response $\boldsymbol{y} = M(\boldsymbol{u})$ and the experimental one $\hat{\boldsymbol{y}}$ is minimum. Herein, $\boldsymbol{u}$ denotes the candidate explanatory model variables, whereas $\mathcal{M}$ is the class of candidate models. The error function $g\left(\boldsymbol{y}|\ \boldsymbol{y} = M(\boldsymbol{u})\right)$ is evaluated over a dataset consisting of R realizations of the explanatory variables $\{\boldsymbol{u}_1 \ldots \boldsymbol{u}_R\}$ and corresponding target outputs $\left\{\hat{\boldsymbol{y}}_1 \ldots \hat{\boldsymbol{y}}_R\right\}$ (training set), which is compared with the predictions $\left\{\boldsymbol{y}_1 \ldots \boldsymbol{y}_R\right\}$ in order to evaluate the error function $g\left(\boldsymbol{y}|\ \boldsymbol{y} = M(\boldsymbol{u})\right)$.

Formally, the nonparametric identification problem can thus be formulated as follows:

$$\begin{aligned} &\min_{M(\boldsymbol{u})} \{g\left(\boldsymbol{y}|\ \boldsymbol{y} = M(\boldsymbol{u})\right)\} \\ &\quad \text{s.t.} \\ &\quad M(\boldsymbol{u}) \subset \mathcal{M}. \end{aligned} \tag{20}$$

It is premised that GP and ANN share the same goal, but they differ in the way the model $M(\boldsymbol{u})$ is built. Once the optimal model has been found, it is employed to predict the response of the dynamic system for input data other than those used for the nonparametric identification (i.e., validation set) in order to test its generalization capability.

The generalization capability of the model(s) is of paramount importance in nonparametric identification. It can be defined as the ability of the model(s) to provide satisfactory predictions for all future applications rather than performing well for a particular case and/or dataset, provided that the training set exhibits the phenomenological characteristics of interests.

To achieve a good generalization, nonparametric identification methods need to avoid both underfitting and overfitting. Specifically, underfitting is prevented by using a large enough dataset when solving the nonparametric identification problem as well as ensuring that the algorithm is properly designed to minimize the distance between the predicted nonlinear dynamic system response and the experimental one. Overfitting is a more sneaky issue in data-driven modeling. It manifests itself when solving the nonparametric identification problem as an uncontrolled growth of the model complexity without a significant reduction in the distance between the predicted nonlinear dynamic system response and the real one (as a matter of fact, the model tends to fit the noise embedded in the dataset). Reducing the complexity of the final models in nonparametric identification is also useful to facilitate their possible physical interpretation.

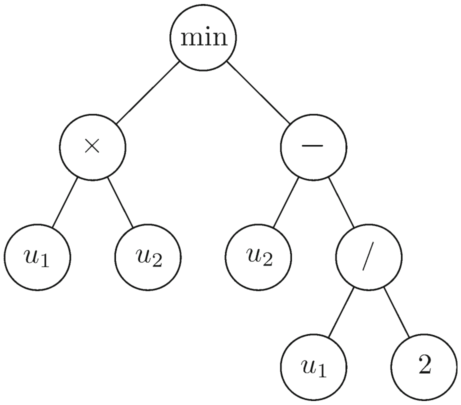


Fig. 1 An instance of tree-based candidate solution representation in GP. Such a tree represents the candidate solution $y = \min\{u_1u_2, u_2 - u_1/2\}$

3.2 Genetic programming

3.2.1 Algorithms

GP can be considered as an extension to computer programs of the Darwin theory of evolution in use in the GA [171]. In fact, GP aims to construct computer programs through an iterative procedure that, in principle, follows the same main steps (e.g., selection, crossover, and mutation).

As it happens for the GA, an important premise for the implementation of the GP is the way by which the candidate solutions (i.e., the computer programs) are codified. In the most classical representation, computer programs are formalized as trees, in which variables and constants are terminal nodes (i.e., leaves of the tree), whereas arithmetic, mathematical, logic, etc. operators are function nodes (i.e., internal nodes of the tree). A tree is a special directed graph that initiates from a root node where connections back to the immediate previous (parent) node only are allowed (Fig. 1).

The primitive set of the GP is defined as the union of the sets of allowed terminals and functions. A terminal node closes a branch of the tree while function nodes require some arguments (i.e., other function nodes or terminals) connected by means of branches. By virtue of the tree-based solutions representation, therefore, the arity of an operator (i.e., the number of arguments or operands that a function or operator takes as input) is the number of branches departing from the corresponding function node. The level of a terminal node is the number of branches back to the root node. The depth of

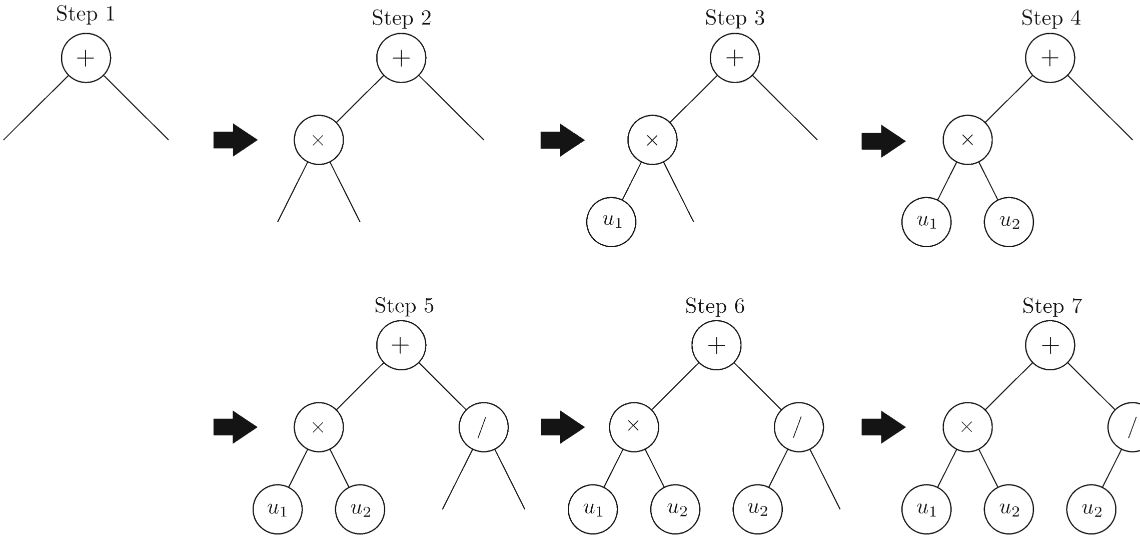


Fig. 2 A step-by-step instance of a full method application to generate a full tree having maximum depth equal to 2

the tree is the maximum level taken over the terminal nodes.

It is also a common expedient in GP to use the so-called representation based on the prefix notation in order to make the visualization of the computer program more compact with respect to the tree-based solutions representation. Although these representations are widely adopted, it should be pointed out that they might be highly inefficient from a computational standpoint. As a consequence, alternative representations have been proposed in the last years, namely graph [266] and linear [38] representations.

Similar to other evolutionary computational techniques, an initial population of candidate solutions has to be initialized randomly. To this end, the simplest approaches are the full method and the growth method, both based on the initial generation of computer programs constrained to a pre-established maximum depth.

According to the full method (Fig. 2), the nodes are taken randomly from the functions set until the maximum tree depth is reached, and beyond that depth only terminals can be chosen. On the other hand, according to the growth method, the nodes are picked up from the whole primitive set (i.e., terminals and functions) until the depth limit is achieved, below which only terminals are selected (Fig. 3). The growth method generates trees with different sizes (i.e., number of the nodes) and shapes (i.e., position of the nodes), whereas the full method produces full trees only. Therefore, the growth method is able to introduce a larger diversity in the initial population with respect to the full method, but it is highly sensitive to the cardinality of functions set and terminals set (e.g., if the terminals set cardinality is much larger than the functions set cardinality, then the growth method will mostly generate short trees). The ramped half-and-half method employs both the full method and the growth method in the attempt to fix this drawback. Accordingly, one half of the initial population is constructed using the full method while the remaining half is initialized by means of the growth method. A range of depth limits is considered in order to ensure that the initial population includes trees having different sizes and shapes. For alternative initialization methods, the interested reader can refer, for instance, to [207].

Genetic operators are then applied at each iteration to candidate solutions selected on the basis of their fitness. The fitness of the candidate solution reflects (but it is not necessarily limited to) the error between predicted and desired outputs. The tournament selection is among the most common methods for selecting individuals in GP. The tournament selection basically consists in selecting the best candidate solution from a small subset of computer programs picked up randomly from the current population. The cardinality of this subset is named tournament size and defines the selection pressure, i.e., the likelihood of better candidate solutions

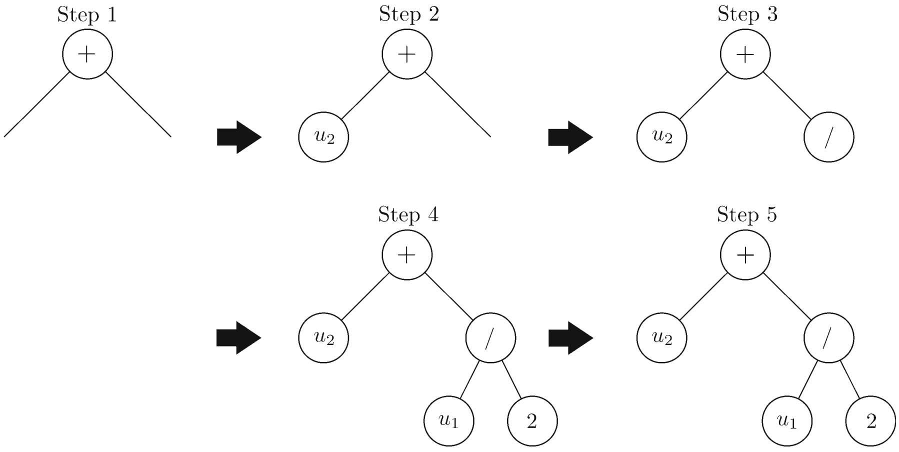


Fig. 3 A step-by-step instance of growth method application to generate a tree having maximum depth equal to 2. It can be noted that the left branch is closed even though the maximum depth has not been achieved

to be winner of the tournament (if the tournament size is large, then weak candidate computer program solutions have a smaller chance to be selected). The main operators involved in GP are crossover and mutation.

The crossover phase consists in the creation of offspring computer programs through the recombination of some parts of selected parent computer programs. A typical crossover strategy is the so-called subtree crossover. It operates by selecting randomly a crossover point in each one of the trees corresponding to two selected parent computer programs (Fig. 4). It is commonly suggested to select functions 90% of the times and leaves 10% of the times as crossover point in order to prevent swapping of small subtrees. The subtree rooted at the crossover point in a copy of the first parent is then replaced with a copy of the subtree rooted at the crossover point in the second parent in order to generate an offspring. The size-fair crossover is another popular crossover operator in GP, and it aims at switching subtrees with similar size. In this case, the first crossover point is selected randomly in the first parent computer program as in the subtree crossover. The size of the resulting subtree is then evaluated and used to constrain the choice of the crossover point in the second parent computer program.

One of the most common mutation operators is the subtree mutation (Fig. 5a). It can be conveniently implemented as the subtree crossover, but considering a freshly generated computer program in place of the second parent. An alternative mutation operator is the point mutation (Fig. 5b). In such a case, a random node is selected and the corresponding primitive is replaced with a different random primitive of the same arity extracted from the primitive set. Should a primitive of the same arity not exist, then point mutation is not performed. Constants in computer programs can be subjected to point mutation by adding random noise.

It is to be noted that subtree mutation involves the alteration of a whole subtree, whereas point mutation is applied with a given mutation rate on a per node basis (and thus, multiple nodes are allowed to undergo mutation). The uniform (or size-fair) mutation is similar to the subtree mutation, i.e., a random tree is generated to replace the one rooted at the selected point, but the mutated computer program depth is constrained to grow no more than a given percent of the original value (for example, no more than 15%).

The reproduction operator is also sometimes applied in GP together with crossover and mutation: it works by simply copying a small fraction of fit individuals from one generation to the next. This set of steps (i.e., selection, crossover, mutation) is repeated at each iteration in order to evolve the initial population toward the optimal solution. The iterative procedure is stopped once a given termination criterion is fulfilled, such as the maximum number of generations.

Besides the termination criterion, other important control parameters are the population size and the probability of performing the genetic operators. Following [171], it has become quite common in GP to assume a population size ranging from 500 to 2000 with short runlength, e.g., 50 generations. Recent research in this field indicates that splitting large populations into semi-isolated demes can be beneficial to preserve a satisfactory population diversity. There are also some proposals for handling gradually decreasing population size [208] or a small population size [14] in GP. As far as genetic operators are concerned, the crossover is com-

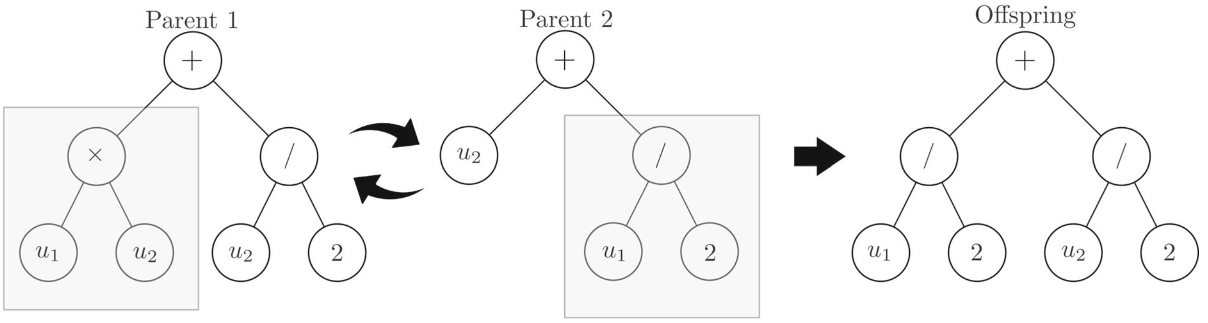


Fig. 4 An instance of subtree crossover. The tree parents represent the candidate solutions $y = u_1 u_2 + u_2/2$ (Parent 1) and $y = u_2 + u_1/2$ (Parent 2). The subtrees rooted at the crossover point are marked with a gray background region. After the subtree crossover, a new tree corresponding to the candidate solution $y = u_1/2 + u_2/2$ is obtained

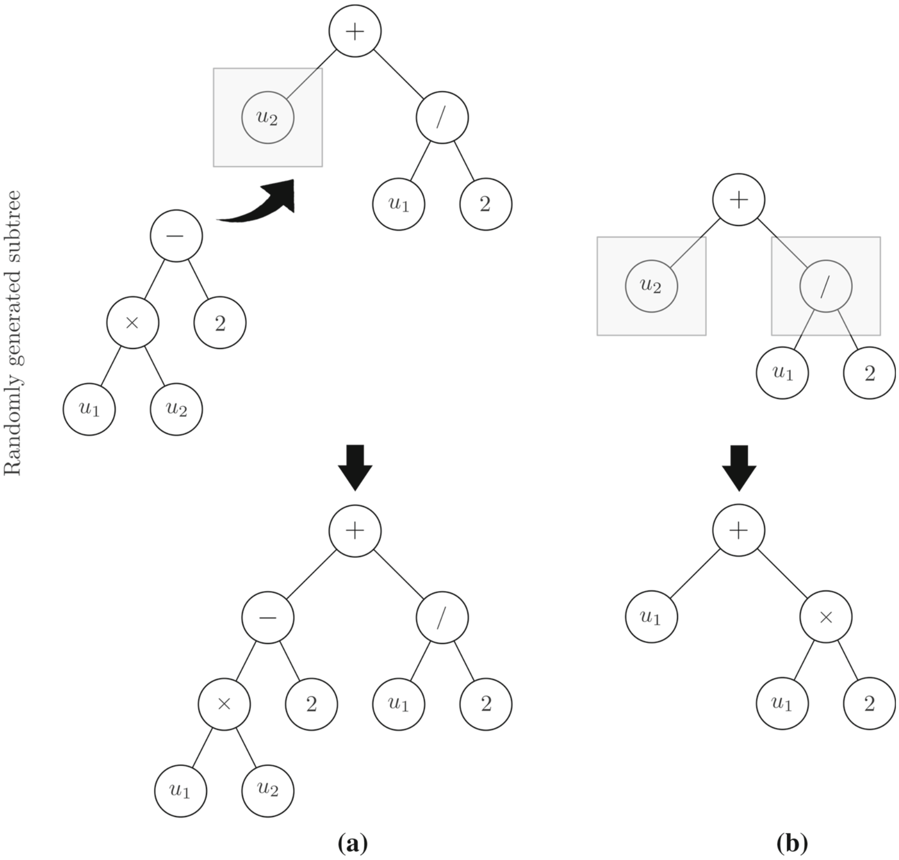


Fig. 5 An instance of some mutation operators. The candidate solution under mutation is $y = u_2 + u_1/2$. **a** (Left) illustrates the subtree mutation: the subtree rooted at the mutation point is marked with a gray background region, and it is replaced with a freshly generated computer program. **b** (Right) shows the point mutation: the nodes of the tree selected for mutation are indicated by a gray background region, and the corresponding primitive is replaced randomly with a different suitable primitive. In the first case, the candidate solution $y = u_1 u_2 - 2 + u_1/2$ is obtained. In the second case, the candidate solution $y = 3u_1$ is generated

monly employed to generate about 90% of the individuals in the population. Reproduction and mutation are typically employed to generate about 8% and 2% of the population, respectively.

It is important to highlight that the functions set is required to fulfill the so-called closure property, in such a way that GP can work properly during the iterative procedure. Such a property, in turn, can be declined into the properties of type consistency and evaluation safety. Type consistency property implies that all the functions be type consistent, i.e., they all return values of the same type, and each of their arguments also has this type. On the other hand, the evaluation safety property is intended to ensure that any possible expression that can be represented by a tree can be evaluated. These properties are required because genetic operators can virtually introduce any possible function and terminal in the tree. Type consistency and evaluation safety requirements can be weakened, provided that proper strategies are implemented for handling exceptions.

One of the main issues to be faced in GP is the uncontrolled growth of model complexity, also known as bloat. This phenomenon refers to an excessive tree growth not accompanied by a significant increment of the accuracy level, which also impedes a physical interpretation of the corresponding computer program. Different strategies have been proposed to mitigate the bloat in GP, each one with its own pros and cons. The simplest one consists in setting a threshold for the size or the depth of the computer programs [171], in such a way that offsprings exceeding such limit are discarded (keeping the parents instead). Poli [267] proposed the so-called Tarpeian method to handle the bloat problem. In this case, a fixed proportion of the individuals in the current population is selected randomly among those whose size is larger than the current average computer program size, and the corresponding fitness value is set as low as possible. Unless pathological conditions occur, this will prevent them from being selected to create the population for the next generation.

Alternatively, it is possible to formulate the fitness function taking into account the model accuracy as well as a complexity-based term [31] intended to penalize the computer programs with large size or depth (so-called parsimony pressure method). It has been also proposed to cope with bloat within a multi-objective formulation [96], where model accuracy and model complexity are conflicting criteria.

In many cases, it has been observed that bloat is attributable to the fact that the resulting expressions are not in the simplest algebraic form. In order to address such problem, computer algebra routines can been introduced to allow GP to simplify the expressions [306]. Finally, size-fair crossover and mutation operators can be implemented to control the unnecessary growth of computer program sizes [71].

Readers who might be interested in gaining a deeper understanding of GP can refer, for instance, to [38, 171, 178] while some of the available theoretical results are summarized in [268].

3.2.2 Applications

First, applications of GP in nonlinear dynamic systems identification are reported in [121, 122]. In these studies, the numerical parameters involved in the candidate model structures discovered through GP are optimized by combining Nelder–Mead simplex search and SA technique. Gray et al. [121] applied such a hybridized GP for model-free identification using data simulated through a nonlinear system involving pure time delay, saturation, and a second-order linear block. Gray et al. [122] included some applications using experimental laboratory data from a coupled water tank system and helicopter rotor speed controller and engine.

Rodriguez-Vazquez et al. [280], Rodríguez-Vázquez and Fleming [279], Zakaria et al. [357], and La Cava et al. [176] addressed nonlinear dynamic systems identification through GP by taking into account model accuracy and complexity simultaneously within a multi-objective formulation. Rodriguez-Vazquez et al. [280] and Rodríguez-Vázquez and Fleming [279] presented some numerical simulations based on Wiener and Duffing systems, respectively. An experimental validation for a gas turbine engine is also illustrated in [280]. Based on a comparative assessment, Zakaria et al. [357] concluded that a multi-objective formulation is more effective in reducing the model complexity than a single-objective approach in which the fitness of the individuals is penalized taking into account the number of insignificant terms of the corresponding computer programs. La Cava et al. [176] implemented a linear representation-based GP where the population undergoes a new set of transformations (epigenetic hill climbing) just before the selection phase [175]. Such modified GP was implemented in order to identify wind turbines in a model-free fashion by looking for the optimal balance in terms of model accuracy and simplicity. The proposed multi-objective formulation also includes a third criterion related to the age of the candidate model [291], i.e., the number of generations since its oldest ancestor was created. The goal of introducing the age as a further decision criterion is twofold. First, it serves to ensure the diversity within the population during the iterative procedure by preserving younger models that might be otherwise dominated by older ones because they are more fit and/or less complex. By assuming that younger candidates dominate older ones that may be otherwise equivalent, an age-based decision criterion also pressures the models to improve in fitness and/or complexity during the iterative procedure, thereby reducing the risk of premature convergence.

dos Santos Coelho and Pessôa [86] focused on the influence of the likelihood of genetic operators being applied in GP. Specifically, probabilities of crossover and mutation are assigned using a large number of alternative strategies, namely (i) constant values; (ii) random values; (iii) chaotic values; (iv) linearly varying values during the iterative procedure; (v) generation-dependent values based on the current and previous best fitness values; (vi) generation-dependent values based on the current and previous best fitness values as well as the average population fitness. Applications using experimental data from a ball-and-tube prototype laboratory system showed that this latter approach provides the best results, but the others exhibit similar performances.

A model-free input/output identification approach for nonlinear SDOF systems based on GP is illustrated in [32] and [33]. Tournament selection, subtree crossover, and subtree mutation were here applied together with the reproduction operator. Parsimony and accuracy of the model structure are improved by applying periodically a combination of specialized strategies. In detail, bloat in GP is alleviated through computer algebra. Removal of trees whose depth is higher than a given threshold and a suitable formulation for the fitness evaluation of the individuals are also considered in [33]. In this case, the fitness evaluation takes into account two contributions, namely (i) the root mean square of the difference between the reference data and the corresponding model predictions; (ii) a penalty term based on the number of tree nodes. A standard GA is also employed to optimize the numerical parameters of the model after parsimony enhancement via computer algebra. The validation for dynamic loading conditions different from those employed to generate the training dataset demonstrates the satisfactory reliability of the procedure without a priori assumptions about the underlying model. The major critical issue is that such a procedure requires numerical differentiation and integration of data contaminated by noise, which is a well-known critical task. Bolourchi et al. [32] and Bolourchi et al. [33] reported successful applications using data simulated through Duffing/van der Pol systems, oscillators with Coulomb friction law, bi- and tri-linear oscillators, as well as Bouc–Wen model of hysteresis.

3.3 Artificial neural networks

3.3.1 Algorithms

The history of ANNs can be traced back to the 1940s [226,265] and basically stemmed out of the large efforts made to explain how the human brain works. Therefore, a proper understanding of an ANN requires a few preliminary physiology insights on neuronal activity in the human brain (Fig. 6a). In a very simplified way, incoming signals originated from other neurons or cells are transferred to a neuron by special connections, namely the synapses. Such connections can usually be found at the dendrites of a neuron and, sometimes, at the soma (which is the cell nucleus). Electrical and chemical synapses can be distinguished. In the electrical synapse, an electrical signal received by the synapse is transferred directly to the postsynaptic side. In case of chemical synapse, the incoming signals flow from the presynaptic side to the postsynaptic area by means of a chemical process rather than an electrical connection. A synaptic cleft and neurotransmitters make the chemical synapse a one-way signal transmission mechanism (i.e., signals in the postsynaptic side cannot flash over to the presynaptic area).

Next, dendrites receive electrical signals from different sources, which are then transferred into the nucleus of the cell. Once the soma has received activating (stimulating) and inhibiting (diminishing) signals, they are accumulated. As soon as the accumulated signal exceeds a certain value (called threshold value), the cell nucleus of the neuron produces an electrical pulse. The pulse is then transferred through the axon. It is electrically isolated through myelin sheath to better transmit the signal and leads to dendrites, which transfer the information to other neurons or other kinds of cells.

Neurocomputing is based on the implementation of an ANN, which is developed starting from the simulation of the neuronal activity in human brain. This, in turn, moves from the formulation of an artificial neuron (Fig. 6b) that mimics a real neuron and realizes the information processing unit of an ANN. The artificial neuron consists of a set of S synapses, each of them with its own weight. Specifically, any input a_q at the qth synapse (with $q = 1, \ldots, S$) of the pth artificial neuron is multiplied by the synaptic weight υ_{pq}. A combiner then aggregates the weighted inputs through a summation, whereas an activation function $\phi(\cdot)$ finally regulates the output of the pth artificial neuron c_p. The

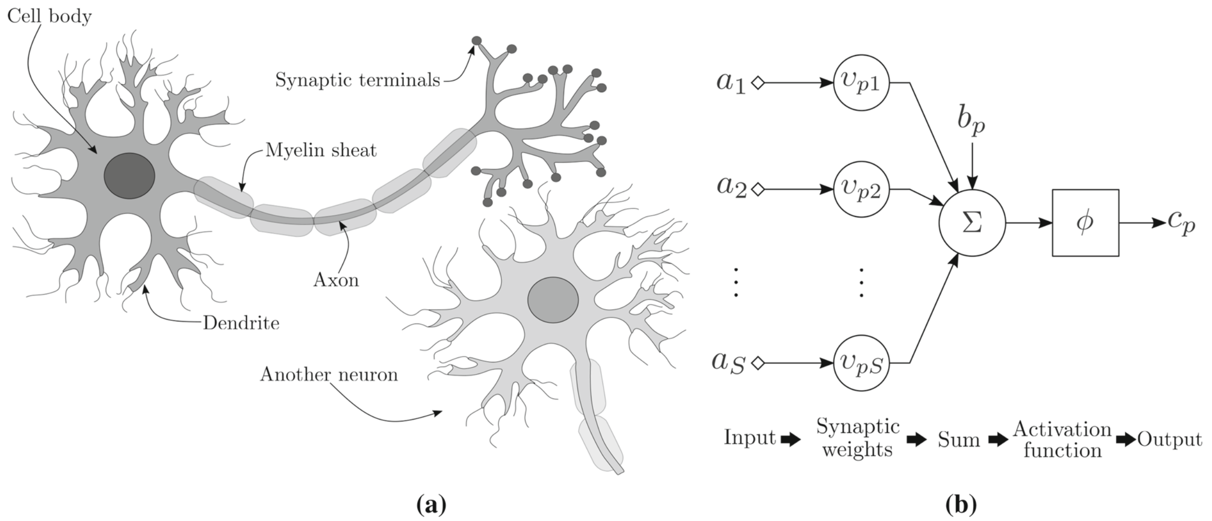


Fig. 6 Information processing unit of an ANN. **a** (Left) illustrates the essential components of a neuron in a simplified way. **b** (Right) is the nonlinear approximation of the neuron behavior (artificial neuron)

typical artificial neuron also includes an external bias b_p (Fig. 6b) that has the effect of increasing or reducing the net input of the activation function. Formally, this implies the following one-way input–output relationship for the pth artificial neuron:

$$c_p = \phi \left(\sum_{q=1}^{S} \upsilon_{pq} a_q + b_p \right) . \tag{21}$$

The bias parameters, however, can be eventually absorbed into the set of weight parameters.

Depending on which activation function is implemented, the amplitude of the output produced by an artificial neuron typically lies between 0 and 1 or, alternatively, between -1 and 1. This calls the need for scaling of the network input data in order to avoid simultaneous saturation of all nodes in the network. It is emphasized that such scaling does not alter the internal network processing. The choice of activation function is determined by the nature of the data and the assumed distribution of target variables, but the Sigmoid function is by far the most common in ANNs.

It should be pointed out that several models have been proposed for modeling the neurons and their couplings. Stochastic, chaotic, and quantum artificial neuron models have been proposed, see, for instance, [4,190,332]. Additionally, memristors have been used to emulate a biological synapse (i.e., memristor-based ANNs), see, for instance, [48,322,362]. A field coupling between neurons based on the combination of capacitor and induction coil is illustrated in [212]. Wang et al. [327] as well as Ma and Tang [210] reviewed the dynamics of network of neurons based on different artificial neuron models.

The manner in which the artificial neurons are structured defines the ANN architecture (or layout). Let the neurons be organized to form layers. Then, the simplest architecture is the single-layer feed-forward neural network (FNN), in which one layer of artificial neurons (i.e., a single layer of computation nodes) provides the output directly from the assigned input layer (Fig. 7a). Within a multi-layer FNN (Fig. 7b), one or more hidden layers take place between the input layer and the output layer of artificial neurons (accordingly, the processing units of hidden layers are called hidden artificial neurons). It is opportune to point out that there are differences in the scientific community about the way by which the layers of ANNs are counted. A multi-layer perceptron (MLP) is a FNN consisting of an input layer, a hidden layer and an output layer in which the information processing unit (the perceptron) implements a Heavised function.

The operation of a multi-layer FNN can be explained as follows: The input layer supplies the inputs for the second layer of the network (i.e., the first computation layer, namely the first hidden layer of artificial neurons). The output signals produced by this second layer are then passed as inputs to the third layer of the network (i.e., the second computation layer, namely the

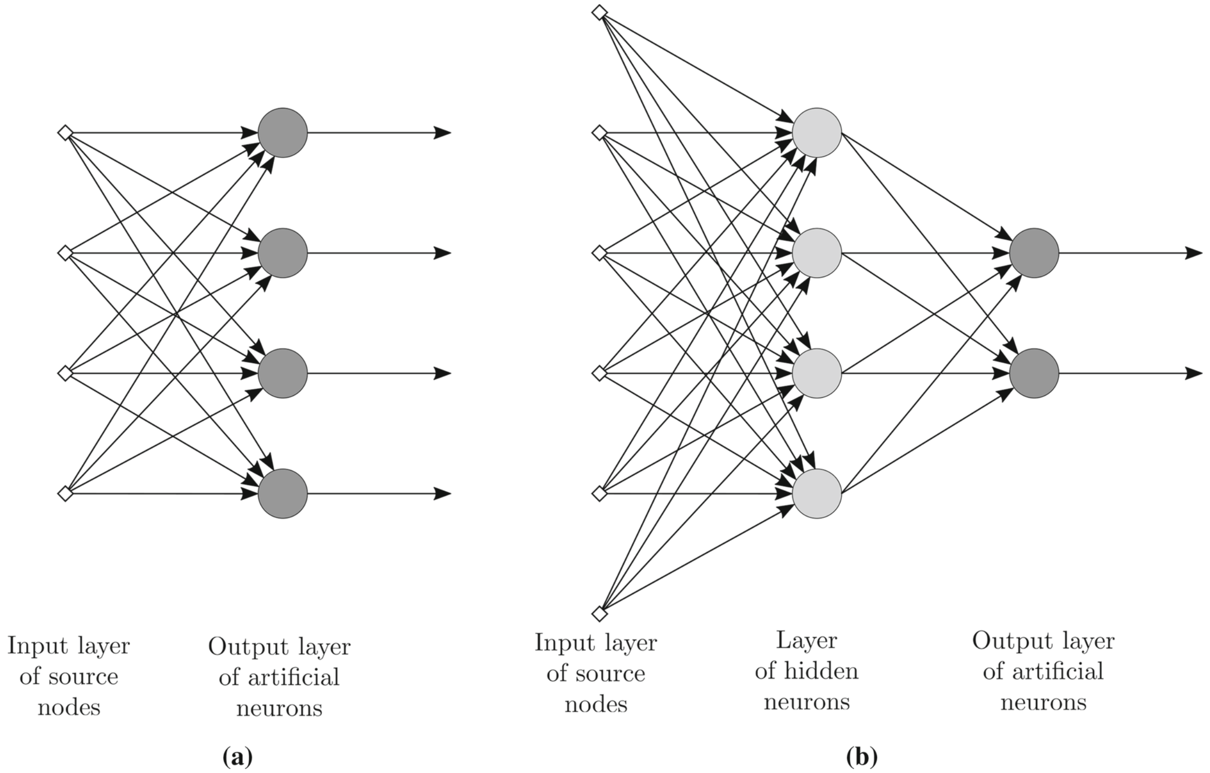

Input layer of source nodes
Output layer of artificial neurons
(a)
Input layer of source nodes
Layer of hidden neurons
Output layer of artificial neurons
(b)

Fig. 7 FNN architectures. **a** (Left) illustrates a single-layer FNN. **b** (Right) shows a multi-layer FNN consisting of a single layer of hidden artificial neurons

second hidden layer of artificial neurons), and so on, up to the last layer of the network (i.e., the layer of output neurons). This output layer then provides the response of the ANN to the activation pattern supplied by the input layer. Note that it is common expedient to employ a different activation function for artificial neurons in hidden layers and output layer, depending on the type of problem at hand.

By focusing on a FNN with a single output layer and single hidden layer (Fig. 7b), the outputs $\boldsymbol{y}$ given the inputs $\boldsymbol{u}$ are calculated according to Eq. (21) as follows:

$$\boldsymbol{y} = \phi'' \left[\boldsymbol{\Upsilon}'' \phi' \left(\boldsymbol{\Upsilon}' \boldsymbol{u} \right) \right], \tag{22}$$

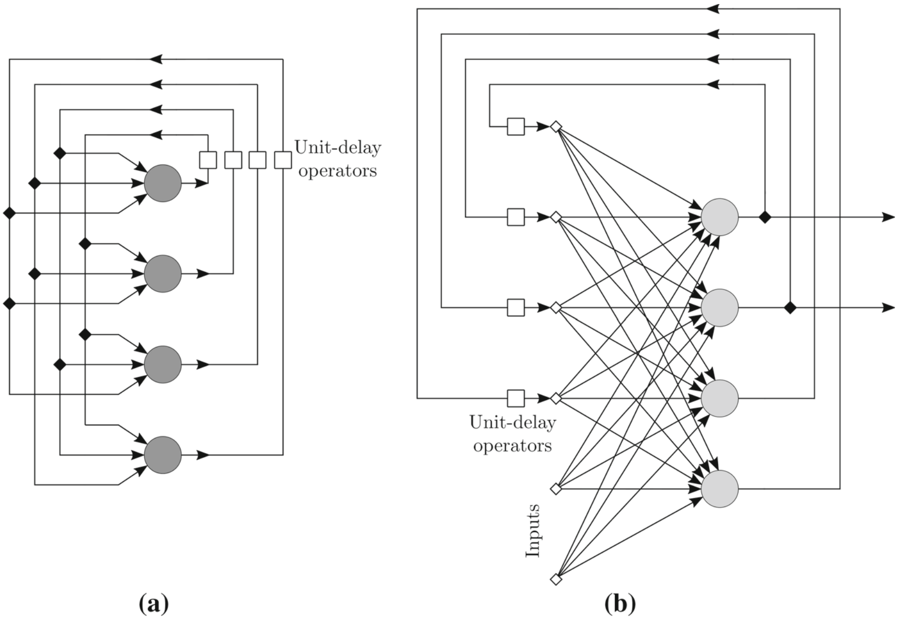
where bias terms have been incorporated into the input vector of each layer for a more concise notation. The symbols $(\cdot)'$ and $(\cdot)''$ indicate that the corresponding quantity $(\cdot)$ refers to the hidden or output layer, respectively. Therefore, $\phi'(\cdot)$ and $\phi''(\cdot)$ are the activation functions of the hidden and output layer, respectively, whereas $\boldsymbol{\Upsilon}'$ and $\boldsymbol{\Upsilon}''$ are the matrices collecting the synaptic weights υ'_{pq} and υ''_{pq} for the qth input of the pth artificial neuron in the hidden and output layer, respectively. Note that, on assuming $\phi(\cdot)$ as Sigmoid function as most common in ANNs, Eq. (22) turns out to be differentiable with respect to the network parameters, which is a useful feature that plays a central role in neurocomputing.

It can be observed in Fig. 7b that every node in each layer is connected to every other node in the adjacent forward layer. This kind of network is said to be fully connected, whereas the lack of some links leads to a partially connected network. Some FNNs also include the so-called shortcut (or skip layer) connections, i.e., connections that skip one or more layers.

Another important class of ANNs is based on a recurrent-type structure (Fig. 8). The main difference of a recurrent neural network (RNN) with respect to the FNN architecture is the presence of at least one feedback loop. Some instances of RNNs are Hopfield network [142], Boltzmann machine [2], Elman neural network (ENN) [97], and long/short-term memory (LSTM) network [140].

Depending on the number of hidden layers, the networks can be considered shallow or deep. Specifically,

Fig. 8 RNN architectures



if there is a single hidden layer, then the network is said to be shallow; otherwise, it is commonly labeled as deep network.

A radial basis function (RBF) network is a special kind of FNN architecture with no shortcuts. In its original version, it is a shallow network that includes the following three layers: (i) input layer; (ii) hidden layer (also named RBF layer, and realized by means RBF neurons); output layer (consisting of so-called RBF output neurons). The connections between input layer and RBF layer are not weighted, whereas that between RBF layer and output layer are weighted. Instead of calculating the inner product of input and synaptic weight vector, a RBF neuron calculates a norm that represents the distance between the input and the so-called position of the neuron (center). This, in turn, is employed into a radial activation function (the Gaussian function is the most common one) to generate the output of the RBF neuron. Output nodes in RBF networks are different from those in the RBF layer, since they employ the identity as activation function. For the sake of completeness, it is mentioned that there is another kind of FNN, named convolutional neural network (CNN), which is especially intended to handle two-dimensional inputs [182], such as in image and video processing applications.

The success of ANNs relies on their ability of replicating the features of a given problem (i.e., learning) through an appropriate training phase. An ANN could learn how to solve the problem at hand in several ways, but the change of synaptic weights and biases is by far the most common one. Among the existing classes of learning algorithms that have been proposed up to now, supervised learning appears to be the most popular one in nonlinear systems identification. In supervised learning, the network outputs are directly compared with the corresponding desired target solutions, and the network is adjusted iteratively in order to minimize the difference between them. The iterative procedure is then arrested once a termination criterion is satisfied. Algorithms for supervised learning of ANNs can be broadly classified into gradient-based and gradient-free (soft computing-based) techniques.

Gradient-based techniques are iterative procedures consisting of two steps, namely: (i) a first step in which the derivatives of the error function with respect to the weights are evaluated; (ii) a second step where these derivatives are used to compute the adjustments to be made to the weights. An efficient technique for the evaluation of such derivatives is the so-called error backpropagation (BP) procedure [285], which is largely implemented through the gradient descent optimization algorithm. In so doing, synaptic weights (and biases) are adjusted iteratively in the direction of the negative gradient of the error function to be minimized. According to the gradient descent optimization algorithm, each synaptic weight in Eq. (22) is updated at the current iteration $(k+1)$ as follows:

$$
\begin{aligned}
{}^{(k+1)}\upsilon'_{pq} = {}^{(k)}\upsilon'_{pq} - \eta_1 \frac{\partial g\left({}^{(k)}\boldsymbol{y}, \hat{\boldsymbol{y}}\right)}{\partial\, {}^{(k)}\upsilon'_{pq}} \\
+ \eta_2\, {}^{(k)}\Delta\upsilon'_{pq}, \qquad \text{(23a)}
\end{aligned}
$$

$$
\begin{aligned}
{}^{(k+1)}\upsilon''_{pq} = {}^{(k)}\upsilon''_{pq} - \eta_1 \frac{\partial g\left({}^{(k)}\boldsymbol{y}, \hat{\boldsymbol{y}}\right)}{\partial\, {}^{(k)}\upsilon''_{pq}} \\
+ \eta_2\, {}^{(k)}\Delta\upsilon''_{pq}, \qquad \text{(23b)}
\end{aligned}
$$

where ${}^{(k)}\boldsymbol{y}$ is the result of Eq. (22) at the iteration k. The last terms in Eq. (23) are called momentum terms, where ${}^{(k)}\Delta\upsilon'_{pq}$ and ${}^{(k)}\Delta\upsilon''_{pq}$ are the variations in the synaptic weights (in the hidden and output layer, respectively) resulting from iteration k. The momentum term provides the search process with a kind of inertia and could help to avoid excessive oscillations in the iteration process. Moreover, η_1 and η_2 are control parameters. The constant η_1 controls the size of the gradient descent step and is usually taken to be around 0.1 to 0.5. If it is too small, learning is slow. If it is too large, instability rises and the algorithm fails to converge. The parameter η_2 is assumed equal to zero in the simplest BP algorithm. A recent review about gradient descent optimization is provided within the preprint by Ruder [284].

The BP algorithm based on the gradient descent optimization algorithm is computationally efficient, but it also suffers from some limitations. The most important drawback is the slow rate of convergence in case of flat error function. Furthermore, it has also been found that learning long-term dependencies with typical gradient descent method is difficult [24]. The Levenberg–Marquardt (LM) algorithm proposed by Hagan and Menhaj [129] for training FNNs is also widely used. The LM algorithm can be considered a trust region modification of the well-known Gauss–Newton method, and the authors found that it is very effective for network training if no more than a few hundred weights have to be estimated. Note that the use of gradient-based techniques for training ANNs requires that the activation functions must be differentiable up to a certain order (depending on the maximum derivative order employed in the algorithm).

Gradient-free methodologies have also become a common alternative to train ANNs. According to Montana and Davis [234] and Sexton et al. [298], GA and SA may be a superior alternative to train ANNs. Mirjalili et al. [231] concluded that a hybrid strategy based on the heuristic combination of PSO algorithm and BP procedure makes the training of FNNs more effective. The Adaptive Random Search (ARS) technique [221] has also been successfully applied by Masri et al. [224] for ANN training.

Conversely, several studies agree with the fact that the standard DE algorithm does not seem really effective for ANN training. For instance, Ilonen et al. [151] observed that DE algorithm does not seem able to provide any distinct advantage in terms of learning rate or solution quality with respect to a gradient-based training technique. Piotrowski [264] also confirmed that DE algorithms do not perform very well in ANNs training.

Setting the best architecture of ANNs is not an easy matter, since it is somewhat problem dependent. As regards the multi-layer FNN, it has been proved that two hidden layers can be useful in many applications. However, it might be convenient to start with a shallow network (one hidden layer) and then retry with a deep network (two hidden layers or more) if needed. Apart from the input and output layers (where the number of artificial neurons is already defined by the problem statement), the goal is to have as few artificial neurons as possible but as many as necessary. This is because the larger the number of artificial neurons, the higher the dimensionality of the search space. This circumstance makes the training phase more challenging and can jeopardize the generalization capability of the ANN. Based on the current literature about FNNs, most layouts implement the hyperbolic tangent function for hidden artificial neurons and a linear activation function in the output layer, whereas synaptic weights are often initialized with small randomly chosen values. In this regard, Yam and Chow [343] proposed a way for determining the optimal bias and magnitude of initial weight vectors in FNNs.

The generalization capability is a crucial aspect in ANN and is enhanced by controlling the complexity of the net in order to avoid overfitting. A common family of methods to achieve a good generalization capability consists in controlling the complexity of the network by deleting the unnecessary connections or units in order to reduce it to an appropriate dimension. The best network size is thus the minimum one that leads to the smallest validation set error. This is the general working principle of the pruning algorithms, see, for instance, [22,49,213].

There are other ways to control the network complexity in order to avoid overfitting and enhance its generalization capability, i.e., addition of a regularization term to the error function, addition of noise to the training data, or early stopping. Details about these alternative approaches can be found in [29,116,237,270].

Finally, it is important to highlight that a significant amount of research has been performed in the last decades to provide a sound theoretical basis for the inner working principle of ANNs. The seminal contributions in this field are due to Cybenko [72] and Hornik et al. [143], who demonstrated that a finite linear sum of continuous Sigmoid functions can approximate any function to any desired degree of accuracy: actually, it implies that a FNN with one hidden layer and an arbitrary number of continuous sigmoid functions is a universal approximator, provided that no constraints exist regarding the number of nodes and the size of the weights. For more extensive discussions about the dynamics of neural nets, the interested reader can refer to [211], whereas further algorithmic and theoretical aspects as well as more general references in the field neurocomputing can be found, for instance, in [28,30,120,126,132,247,282,290].

3.3.2 Applications

First, general applications of ANNs in nonlinear dynamic systems identification can be traced back to the 1990s, see, for instance, [241,263].

The applications in the fields of structural mechanics and engineering basically originated in that period, see, for instance, [222]. A multi-layer FNN is employed in [223] to find an approximation of the unknown restoring force for a nonlinear SDOF system without resorting to a given model. The adopted network architecture has two hidden layers with 15 and 10 artificial neurons in the first and second hidden layers, respectively. The validation phase confirmed the good generalization capability of the trained ANN, provided that new inputs do not exceed the range of values adopted during the training phase. A significant contribution in [223] deals with the design of the training phase, for which two proposals are formulated. The first proposal for training the net consists of the following steps: (i) start with a random set of initial weights and then provide the inputs to the network one by one; (ii) for each training pattern, perform a forward pass using the initial set of random weights and calculate the error between the network response and the desired output; (iii) once the total error over the whole set of training patterns is calculated, apply the BP algorithm to find a new set of weights and continue to iterate until the total error is less than a selected threshold. According to the second proposal for training the net, the steps to be completed are the following: (i) starting from a random set of weights and considering a small subset of training patterns, calculate the error and then execute the BP algorithm to find an updated set of weights; (ii) using this new updated set of weights, keep iterating and updating the weights using a subset of training patterns whose size is larger than that considered in the previous step; (iii) stop the iterative procedure when the total error is less than an acceptable threshold. After a comparative evaluation of both strategies, Masri et al. [223] suggested to use the second method especially when a larger number of training patterns is available.

Worden and Tomlinson [335] illustrated a way to use FNN in classifying nonlinear SDOF dynamic systems according to the type of nonlinearity in the system. As far as the net topology is concerned, they found that a single hidden layer proved sufficient, but two hidden layers generally perform better for classification problems (for their numerical simulations, 8 artificial neurons per hidden layer were found appropriate). The nonparametric identification of polynomial-type nonlinear SDOF systems is addressed in [57], whereas nonlinear structural systems with multiple DOFs are considered in [56,194]. While Chassiakos and Masri [57] and Chassiakos and Masri [56] implemented two hidden layers (with total 25 and 8 hidden artificial neurons, respectively), a single hidden layer (including 13 hidden artificial neurons) was used in [194].

Saliah et al. [288] implemented a FNN (consisting of a single hidden layer employing 15 artificial neurons) to model the hysteretic behavior of magnetic materials: an interesting outcome of this study is that when the trained FNN is used to implement a hysteresis model for a magnetic material within a finite element-based computer program, it provides the same results as the Preisach model but in about 2% of the computation time. Further applications of the FNN in the nonparametric identification of the hysteretic response of magnetic materials were also presented in [297] and, later, in [108], whereas applications related to magnetostriction phenomena are discussed in [1,138].

Besides the applications about SDOF systems subjected to ordinary loading scenarios and dynamic characterization of materials, ANNs have also proved suitable to deal with large structures and/or complex dynamic loads. For instance, Wu and Kareem [339] applied a FNN to model bridge aerodynamic. In such a work, a cellular automata scheme is employed to determine the architecture of the FNN [127], whereas noise is intentionally introduced into the input data to enhance the robustness of the trained net against overfitting. Further applications in aerodynamic modeling have been illustrated in [214] using RNNs and, more recently, in [75]. Derkevorkian et al. [79] applied a FNN for the nonparametric identification of Yokohama Bay bridge using its response recorded under the 2011 Great East Japan Earthquake. Herein, the implemented net has a single hidden layer including 15 artificial neurons.

The ability of the ANNs in identifying changes in nonlinear dynamic systems was also evaluated in some studies for potential structural health monitoring applications [225]. For instance, based on the experimental data recorded on a five-span continuous prestressed box girder bridge, Huang and Loh [146] concluded that a FNN can be a viable tool for estimating the changes in the structural response induced by different seismic events. In a more recent study by Derkevorkian et al. [80], a FNN based on two hidden layers is also used successfully for changes identification in soil–foundation–structure interaction problems.

A larger variety of ANNs in nonlinear dynamical systems identification started to be implemented in 2000s, together with the formulation of rigorous, yet practical, frameworks for their more rational use. For instance, a variant of the BP algorithm based on fuzzy logic is employed within a FNN by Liang et al. [193] for the identification of the restoring force in nonlinear dynamic systems (the network layout employed a single hidden layer and 13 hidden artificial neurons). In the same year, the Volterra/Wiener neural network (VWNN) was introduced by Kosmatopoulos et al. [169] for the nonparametric identification of nonlinear dynamic systems. The VWNN consists of a linear multi-input/multi-output dynamic system connected in cascade with a linear-in-the-weight ANN. This computational procedure is also applied to data carried out from experimental tests on concrete and steel structures. Rojas et al. [281] develop a novel framework for constructing and training RBF networks. Their main contribution is a sequential learning algorithm able to adapt the structure of the network by creating new hidden units and also to detect and remove inactive units. The proposed approach was then employed for short-term prediction of chaotic time series generated by sing Mackey–Glass and Lorenz systems, and final results were found more accurate than the ones obtained using a standard RBF network employing a larger number of hidden artificial neurons. A step forward in the use of ANNs for nonparametric nonlinear dynamic systems identification has been made by Pei et al. [256], where the physical-driven interpretation and an improved version of the VWNN are discussed. Further significant contributions are provided in [254,255,257], in which very useful guidelines for setting up an ANN network for nonlinear dynamical systems identification are given.

Hung et al. [149] and Jiang and Adeli [158] implemented a wavelet neural network (WNN) for nonlinear identification of buildings. The same kind of network was later employed by Ong and Zainuddin [248] in the attempt to predict chaotic phenomena. The formulation of the WNN rests on the observation that the discrete form of the inverse wavelet transform basically represents a single hidden layer FNN where the hidden neuron is a dilated and translated wavelet. By adding a linear term and bias, the final one-way input–output relationship for the hidden neurons in WNNs is obtained (the so-called wavelon).

Guerra and Coelho [125] addressed the problem of defining the parameters of a RBF network (i.e., center and spread of the Gaussian activation functions as well as the network weights) to be used for nonlinear identification of the Lorenz chaotic system. Compared to the previous work on RBF network-based chaotic time series prediction by Rojas et al. [281], the main novelty here is concerned with the application of the fuzzy clustering technique to tune the centers of Gaussian functions in the hidden layer of the RBF network, whereas the PSO technique is implemented for a better tuning. Another RBF network that employs fuzzy rules has been also recently proposed by Rubio-Solis et al. [283], and it has been successfully applied in forecasting chaotic time series (numerical simulations herein discussed are based on the Mackey–Glass system). dos Santos Coelho and Pessôa [85] proposed a variant of the RBF network that implements the B-spline in place of the Gaussian function.

A useful comparative assessment among polynomial-based nonlinear estimators and several kinds of ANN for the nonparametric identification of nonlinear dynamic systems can be found in [40]. The benchmark problem herein considered for the numerical simulations is concerned with the identification of a three-DOF structure that exhibits different types of nonlinear behaviors, including bilinear hysteresis, polynomial stiffness, and Bouc–Wen model of hysteresis. Specifically, Brewick and Masri [40] considered the VWNN and two different ANNs. The first ANN has a single hidden layer (including 15 hidden neurons), and its inputs are displacements, velocities, and external excitation for all available DOFs at the current time step, whereas the nonlinear residual from the previous time step is inputted. This first architecture is then designed to estimate the nonlinear residual at the current time step. The second ANN has two hidden layers (with 10 hidden neurons per layer) and does not include the nonlinear residual in either its inputs or outputs. As it occurs in the first architecture, the inputs here are displacements, velocities, and external excitation from all available DOFs, but the outputs in this case are the accelerations of the available DOFs at the current time step. Both layouts employ the Sigmoid function and the linear transfer function in the artificial neurons of hidden layers and output layer, respectively, whereas the LM algorithm is implemented for training the networks. Final results show that the VWNN outperforms all the considered set of methods, and the identification of Bouc–Wen-type hysteresis proved to be the most challenging identification problem.

Recently, Ganjefar and Tofighi [111] have adopted an ANN that employs a quantum-type artificial neuron model in the attempt to forecast chaotic time series (so-called quantum ANN, QNN). The information processing unit herein is a qubit artificial neuron model whose states are connected to the quantum states and the transitions between the artificial neuron states are based on operations derived from quantum logic gates. Apart from the use of a quantum-type artificial neuron model, the overall architecture resembles one of the classical FNNs, and it consists of a single hidden layer. The authors also implemented a hybrid training procedure by which a GA is employed at the beginning in order to facilitate the exploration of the search space net parameters whereas the gradient descent-based optimization follows once the GA stops, in such a way to perform a more refined local search. The comparison with other conventional ANNs based on numerical data simulated by means of Mackey–Glass and Lorenz systems seems to suggest that such a QNN can lead to more accurate forecasting of chaotic behaviors.

Among the most recent applications of ANNs in nonlinear dynamic systems identification, other works deserving to be mentioned are [160,188,321,328] because they illustrate some of the first applications of deep learning. In detail, Jin et al. [160] applied deep learning for nonparametric identification of power amplifier and control system. Li et al. [188] developed a deep learning strategy for unsteady aerodynamic and aeroelastic modeling, whereas Vlachas et al. [321] as well as Wang et al. [328] addressed the chaotic time series forecasting. Particularly, the major contribution by Wang et al. [328] is the introduction of a special mechanism to enhance the long-term prediction of chaotic time series. A recurrent-type network architecture is employed in all these studies: specifically, the adopted architecture in Jin et al. [160] exploited the ENN while it consisted of LSTM cells in the work done by Vlachas et al. [321], Li et al. [188] and Wang et al. [328].

Significant research about the nonparametric identification of vibration control devices and actuators has been conducted successfully, starting in the mid-1990s. For instance, the nonparametric identification of an electro-rheological damper is performed in [42] through a FNN based on a single hidden layer. Chang and Roschke [50] implemented a FNN based on a single hidden layer and 12 hidden artificial neurons for the nonparametric identification of a magneto-rheological damper. Chang and Zhou [51] also addressed the nonparametric identification of a magneto-rheological damper but, differently from [50], a RNN was herein adopted to identify an inverse nonlinear dynamic model (e.g., to obtain the required voltage to be input to the damper so that a desirable damper force can be produced). Song et al. [307] used two nets to replicate, both, forward and inverse hysteresis relationship between applied voltage and displacement in a shape memory alloy wire actuator. In such a work, while the net for the forward analysis (whose output is the displacement) has a single hidden layer (including 5 artificial neurons), the layer devoted to the inverse analysis (in which the output is the voltage) has two hidden layers (with 4 and 5 artificial neurons in the first and the second layer, respectively). Direct and inverse identification of a magneto-rheological fluid damper is also addressed in [323]. The direct identification is performed to predict the damping force given the dynamic responses and the command voltage, whereas the inverse identification is sought to define the command voltage given the desired damping force. Wang and Liao [323] employed both FNN and RNN for the direct and inverse identification. These networks have basically the same layout (a hidden layer including 18 artificial neurons), with the only exception that the output of the RNN is delayed and fed back to the input layer.

A comparison between parametric and nonparametric techniques for the identification of nonlinear viscous dampers based on experimental data is illustrated in [356]. The parametric identification is performed using the ARS technique in order to minimize an objective function that includes a displacement-based and velocity-based normalized mean square error, where the error is defined as the difference between measurements and predictions obtained from a fractional-power damping law. Two nonparametric identification techniques are also considered, namely the restoring force method and FNN. Final results show that the force-based normalized mean square error of the identification via FNN is the smallest among the investigated methods.

Chungui et al. [65] employed an ANN for the hybrid parametric–nonparametric identification of the nonlinear hysteretic restoring force of a wire cable vibration isolation system. In detail, Chungui et al. [65] assumed that the restoring force is given by the sum of two contributions, i.e., one term reflecting the part that can be modeled on a physical basis and another part that must be identified via a nonparametric approach. On these premises, Chungui et al. [65] applied a FNN (consisting of 2 hidden layers, with 10 and 8 artificial neurons in the first and second layer, respectively) in order to develop a nonparametric model of the difference between the measured restoring force and the prediction carried out from the knowledge-based modeling part.

Most recent applications of neurocomputing in the identification of magneto-rheological devices for vibration control applications are reported in [105,167,185, 320,351]. On the other hand, Brewick et al. [41] as well as Pei et al. [258] provided further contributions about the dynamic characterization of the hysteretic device proposed by Carboni et al. [45] for vibration control. Specifically, Brewick et al. [41] implemented, both, FNN and VWNN for nonparametric identification of NiTiNOL and steel strands. A new family of mem-based models (mem-springs and mem-dashpots) is instead discussed in [258] for the identification via FNN.

For the reader's convenience, Table 4 provides an overview about the type of ANN adopted in nonlinear dynamical systems identification and their applications.

4 Conclusions

General remarks The present review has shown a large record of successful applications of computational intelligence techniques to a variety of nonlinear dynamic system identification problems. One of their most appreciated features is their versatility, i.e., the ability of dealing with the parametric or nonparametric identification of different nonlinear dynamical systems. Existing applications have also shown that these techniques are able to deal with missing data and rather large noise levels. However, such satisfactory results largely depend on the correct design and implementation of the corresponding algorithms, as well as the proper tuning of the involved control parameters. This is likely the most critical task, especially for new users and those who attempt to adopt these methods without a solid grasping of the working mechanisms.

One of the most common criticisms toward the use of such techniques is the lack of sound mathematical proofs about their stability, convergence, and performance. This belief should be partially demystified: although the theoretical foundations of computational intelligence techniques are still not as solid as those of "classical" methods, several theories (although based on relaxed hypotheses) are already available and continue to be developed (see, e.g., [346]). This review—together with the extensive list of referenced literature sources—is meant to facilitate the rational use of computational intelligence techniques in nonlinear dynamic system identification.

The overall review demonstrated that GA, PSO, and DE are all viable tools for the parametric identification of nonlinear dynamic systems. Although the largest number of existing applications is based on the use of GA, evolving trends indicate that PSO and DE are becoming more and more popular. This can be explained by noting that most of the comparative evaluations seem to agree with the fact that PSO and DE are often more effective than GA in parametric identification of nonlinear dynamic systems (in general, it seems that DE outperforms PSO and GA). Additionally, the existing proposals for the key operators in DE or PSO

Table 4 Nonparametric identification using ANN

Reference	Type	Applications
Masri et al. [222]	FNN	Chaotic behavior, robotic manipulator
Masri et al. [223]	FNN	Chaotic behavior
Worden and Tomlinson [335]	FNN	Chaotic behavior
Burton et al. [42]	FNN	Electro-rheological damper
Chassiakos and Masri [56]	FNN	Chain-like nonlinear system
Chassiakos and Masri [57]	FNN	Chaotic behavior
Liang et al. [194]	FNN	Chain-like system with hardening springs
Saliah et al. [288]	FNN	Hysteresis loops of magnetic material
Chang and Roschke [50]	FNN	Magneto-rheological damper
Serpico and Visone [297]	FNN	Hysteresis loops of magnetic material
Masri et al. [225]	FNN	Changes identification in nonlinear systems
Huang and Loh [146]	FNN	Hysteretic nonlinearity, prestressed bridge
Kosmatopoulos et al. [169]	VWNN	Reinforced concrete and steel structures
Liang et al. [193]	FNN	Chain-like system with hardening springs
Chang and Zhou [51]	RNN	Magneto-rheological damper
Rojas et al. [281]	RBF	Chaotic behavior
Hung et al. [149]	WNN	Building structure under ground motion
Song et al. [307]	FNN	Shape memory alloy wire actuator
Pei et al. [256]	VWNN	Tripod structural system
Wang and Liao [323]	FNN, RNN	Magneto-rheological damper
Jiang and Adeli [158]	WNN	Highrise building under ground motion
Pei et al. [257]	FNN	Polynomial nonlinearity
Pei and Smyth [254]	FNN	Reinforced concrete and steel structures
Pei and Smyth [255]	FNN	Polynomial and hysteretic nonlinearities
Hilgert et al. [138]	FNN	Magnetostrictive hysteresis
Guerra and Coelho [125]	RBF	Chaotic behavior
Yun et al. [356]	FNN	Nonlinear viscous damper
Chungui et al. [65]	FNN	Wire cable vibration isolation
dos Santos Coelho and Pessôa [85]	RBF	Ball-and-tube system
Wu and Kareem [339]	FNN	Bridge aerodynamic modeling
Fulginei and Salvini [108]	FNN	Hysteretic magnetic materials
Tudón-Martínez et al. [320]	FNN	Magneto-rheological fluid damper
Abdelmadjid et al. [1]	FNN	Magnetostrictive hysteresis
Derkevorkian et al. [79]	FNN	Bridge under earthquake
Khalid et al. [167]	RNN	Magneto-rheological damper
Mannarino and Mantegazza [214]	RNN	Airfoil aerodynamic modeling
Derkevorkian et al. [80]	FNN	Soil–foundation–structure interaction
Yu et al. [351]	FNN	Magneto-rheological elastomer isolator
Brewick and Masri [40]	FNN, VWNN	Polynomial and hysteretic nonlinearity
Brewick et al. [41]	FNN, VWNN	Assemblies of NiTiNOL and steel wire strands
Fu et al. [105]	RNN	Magneto-rheological elastomer isolator
Jin et al. [160]	ENN	Power amplifier, control system

Table 4 continued

Reference	Type	Applications
Ganjefar and Tofighi [111]	QNN	Chaotic behavior
Leng et al. [185]	FNN	Magneto-rheological elastomer isolator
Rubio-Solis et al. [283]	RBF	Chaotic behavior
Vlachas et al. [321]	LSTM	Chaotic behavior
de Paula and Marques [75]	FNN	Airfoil aerodynamic modeling
Li et al. [188]	LSTM	Airfoil aerodynamic modeling
Ong and Zainuddin [248]	WNN	Chaotic behavior
Pei et al. [258]	FNN	Assemblies of NiTiNOL and steel wire strands
Wang et al. [328]	LSTM	Chaotic behavior

basically share the same working principle and their optimal control parameter values typically lie within a narrow range. Conversely, the existing proposals for the key operators in GA can be very different from each other. Another point of merit is that the implementation of PSO and DE (including their most advanced variants) typically results in a more concise code using the most typical programming languages. Overall, these features make PSO and DE more robust in parametric identification problems than GA.

On the other hand, it does not seem to be fully reasonable to carry out a comparison between the performances of GP and ANN in nonparametric identification problems because their outcomes are different. Regardless of this, it is clear that ANN is much more common than GP, and it has more potential in this field. A positive feature of GP with respect to ANN is that, in some cases, the mathematical blocks in the final data-driven model structure can facilitate a physics-based interpretation. This, in turn, can drive the efforts undertaken in white or gray box modeling.

The development of new computational intelligence methods deserves further final remarks. Success and popularity of computational intelligence are in some way also the cause of a deplorable trend whereby a huge number of new algorithms or variants based on a metaphor of some natural or man-made process are proposed continuously (sometimes with limited or inappropriate validations), although their effectiveness and scientific rigor are often marginal at best, or highly questionable [308]. Albeit innovative research of high quality is being performed [78], there are also several works that do not really contribute to scientific advances (especially outside the scientific community involved in computer sciences). Because of this, some archival journals are adopting specific policies to cope with such a trend. In the field of structural mechanics and engineering, for instance, the journal *Mechanical Systems and Signal Processing* prescribes specific requirements for submissions that lie in the area of soft computing [238]. In this perspective, it is pointed out that the present state-of-the-art review has deliberately made the choice to be focused on the most common and consolidated methods only, in such a way to provide effective guidelines for nonlinear dynamic systems identification. For a broad overview on new generation computational intelligence methods, the reader can refer to the recent survey made by Dokeroglu et al. [89].

Open issues and future developments Although the computational intelligence-based nonlinear dynamic system identification has witnessed a vigorous growth in the last decades, there is still large room for improvements and breakthroughs. Without any intended limitation, some possible directions for future research are drawn next.

- Within the framework of parametric identification, for instance, the feasibility of compact algorithms can be eventually explored in the attempt to tackle some specific open challenges, i.e., hardware limitations for direct running on wireless sensor boards and near real-time applications. Compact algorithms do not manage a population of candidate solutions, but use a probabilistic representation of it. Existing studies on benchmark optimization problems have shown that compact GA [130,229], compact PSO [245], and compact DE [150,230] are quite promising in comparison with the corresponding noncompact versions, and thus, their feasibility for nonlinear dynamical systems identification might deserve dedicated studies. Another broad family of computational intelligence methods that seems potentially suitable for more extensive studies in nonlinear dynamical systems identification is memetic algorithms. They are a class of optimization algorithms whose structure is characterized by an evolutionary framework and a list of local search components [243].

- As far as nonparametric identification is concerned, the use of the evolutionary polynomial regression (EPR) might be worthy of considerations in place of, or together with, GP. This technique has been originally proposed by Giustolisi and Savic [118] and leverages on the combined use of single-objective GA and least squares method, in order to construct general nonlinear models. The original proposal was later revised in [117] by replacing the single-objective GA with a multi-objective GA in order to cope with bloat. As an example, Doglioni et al. [88] employed the EPR for hydrological time series forecasting while it was adopted in [101] to predict maximum inter-storey drift and peak floor accelerations in reinforced concrete frames under earthquakes, given one or more seismic intensity measures and minimum data about the structure (i.e., the fundamental period). Several opportunities for new results in neurocomputing-based nonparametric identification of nonlinear dynamic systems also exist. The application of more advanced ANNs and deep learning needs extensive research and comparative evaluations.
- The way by which different and huge databases can be effectively managed for improved nonlinear systems identification is basically an open issue. Therefore, significant breakthroughs can arise from the simultaneous use of different sources of information (e.g., dynamic response, images, videos, static measurements) in nonlinear dynamical systems identification. The benefit of big data and available high-capacity computing infrastructures (e.g., parallel or cloud computing) should also be quantified through the analysis of very large and complex case studies that capitalize on evolving research in data fusion approaches. Hardware innovations during the past decade about the still visionary concept of quantum machines (including the recent announcement about the possible achievement of the quantum supremacy, e.g., [12]) can disclose new long-term outlooks for unprecedented applications in nonlinear dynamical systems identification. The opportunities (and related challenges) that such a visionary technology can offer in this field, therefore, might also be worthy of investigations.

Acknowledgements GQ acknowledges the support from Sapienza University of Rome through the project n. RM116154C95-F7DD0 titled “Smart solutions for the assessment of structures in seismic areas.” WL acknowledges the partial financial support from the PRIN Grant n. 2017L7X3CS titled “3D Printing: A bridge to the future. Computational methods, innovative applications, experimental validations of new materials and technologies.”

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Abdelmadjid, N., Elamine, N., Mouloud, F.: Neural network-DFT based model for magnetostrictive hysteresis. Int. J. Appl. Electromag. Mech. **42**(3), 343–348 (2013)
2. Ackley, D.H., Hinton, G.E., Sejnowski, T.J.: A learning algorithm for Boltzmann machines. Cognit. Sci. **9**(1), 147–169 (1985)
3. Aguirre, L.A.: A bird’s eye view of nonlinear system identification. arXiv preprint. arXiv:1907.06803 (2019)
4. Aihara, K., Takabe, T., Toyoda, M.: Chaotic neural networks. Phys. Lett. A **144**(6–7), 333–340 (1990)
5. Ajavakom, N., Ng, C., Ma, F.: Performance of nonlinear degrading structures: identification, validation, and prediction. Comput. Struct. **86**(7–8), 652–662 (2008)
6. Al-Dabbagh, R.D., Neri, F., Idris, N., Baba, M.S.: Algorithmic design issues in adaptive differential evolution schemes: review and taxonomy. Swarm Evolut. Comput. **43**, 284–311 (2018)
7. Al-Duwaish, H.N.: A genetic approach to the identification of linear dynamical systems with static nonlinearities. Int. J. Syst. Sci. **31**(3), 307–313 (2000)
8. Almeida, L.A.L., Deep, G.S., Lima, A., Neff, H.: Modeling a magnetostrictive transducer using genetic algorithm. J. Magn. Magn. Mater. **226**, 1262–1264 (2001)
9. Antman, S.: Nonlinear Problems of Elasticity. Springer, New York (2005)
10. Arena, A., Lacarbonara, W.: Flutter of an arch bridge via a fully nonlinear continuum formulation. ASCE J. Aerosp. Eng. **24**, 112–123 (2011)
11. Arena, A., Lacarbonara, W.: Nonlinear parametric modeling of suspension bridges under aeroelastic forces: torsional divergence and flutter. Nonlinear Dyn. **70**, 2487–2510 (2012)

12. Arute, F., Arya, K., Babbush, R., Bacon, D., Bardin, J.C., Barends, R., Biswas, R., Boixo, S., Brandao, F.G., Buell, D.A., et al.: Quantum supremacy using a programmable superconducting processor. Nature **574**(7779), 505–510 (2019)
13. Arvin, H., Lacarbonara, W., Bakhtiari-Nejad, F.: A geometrically exact approach to the overall dynamics of rotating blades. Part II: nonlinear normal modes in flapping. Nonlinear Dyn. **70**, 2279–2301 (2012)
14. Ashlock, W.: Using very small population sizes in genetic programming. In: 2006 IEEE International Conference on Evolutionary Computation, pp. 319–326. IEEE (2006)
15. Atabay, E., Ozkol, I.: Application of a magnetorheological damper modeled using the current-dependent Bouc–Wen model for shimmy suppression in a torsional nose landing gear with and without freeplay. J. Vib. Control **20**(11), 1622–1644 (2014)
16. Bai, X.X., Chen, P., Qian, L.J.: Principle and validation of modified hysteretic models for magnetorheological dampers. Smart Mater. Struct. **24**(8), 085014 (2015)
17. Baker, J.E.: Reducing bias and inefficiency in the selection algorithm. In: Proceedings of the 2nd International Conference on Genetic Algorithms, pp. 14–21 (1987)
18. Banerjee, A., Abu-Mahfouz, I.: A comparative analysis of particle swarm optimization and differential evolution algorithms for parameter estimation in nonlinear dynamic systems. Chaos Solitons Fractals **58**, 65–83 (2014)
19. Banks, A., Vincent, J., Anyakoha, C.: A review of particle swarm optimization. Part I: background and development. Nat. Comput. **6**(4), 467–484 (2007)
20. Barbieri, N., Barbieri, R., da Silva, R.A., Mannala, M.J., Barbieri, L.S.V.: Nonlinear dynamic analysis of wire-rope isolator and Stockbridge damper. Nonlinear Dyn. **86**(1), 501–512 (2016)
21. Bartkowski, P., Zalewski, R., Chodkiewicz, P.: Parameter identification of Bouc–Wen model for vacuum packed particles based on genetic algorithm. Arch. Civil Mech. Eng. **19**(2), 322–333 (2019)
22. Bebis, G., Georgiopoulos, M., Kasparis, T.: Coupling weight elimination with genetic algorithms to reduce network size and preserve generalization. Neurocomputing **17**(3–4), 167–194 (1997)
23. Begambre, O., Laier, J.E.: A hybrid particle swarm optimization-simplex algorithm (PSOS) for structural damage identification. Adv. Eng. Softw. **40**(9), 883–891 (2009)
24. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. **5**(2), 157–166 (1994)
25. Bertsimas, D., Tsitsiklis, J., et al.: Simulated annealing. Stat. Sci. **8**(1), 10–15 (1993)
26. Bezdek, J.C.: Computational intelligence defined by everyone! In: Computational Intelligence: Soft Computing and Fuzzy-Neuro Integration with Applications, pp. 10–37. Springer, New York (1998)
27. Bigoni, D.: Nonlinear Solid Mechanics. Bifurcation Theory and Material Instability. Cambridge University Press, Cambridge (2012)
28. Bishop, C.M.: Neural networks and their applications. Rev. Sci. Instrum. **65**(6), 1803–1832 (1994)
29. Bishop, C.M.: Training with noise is equivalent to Tikhonov regularization. Neural Comput. **7**(1), 108–116 (1995)
30. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer, New York (2006)
31. Blickle, T.: Evolving compact solutions in genetic programming: a case study. In: International Conference on Parallel Problem Solving from Nature, pp. 564–573. Springer, New York (1996)
32. Bolourchi, A., Masri, S.F., Aldraihem, O.J.: Development and application of computational intelligence approaches for the identification of complex nonlinear systems. Nonlinear Dyn. **79**(2), 765–786 (2015a)
33. Bolourchi, A., Masri, S.F., Aldraihem, O.J.: Studies into computational intelligence and evolutionary approaches for model-free identification of hysteretic systems. Comput. Aided Civil Infrastruct. Eng. **30**(5), 330–346 (2015b)
34. Bonyadi, M., Michalewicz, Z.: Particle swarm optimization for single objective continuous space problems: a review. Evol. Comput. **25**(1), 1–54 (2017)
35. Bonyadi, M.R., Michalewicz, Z.: Analysis of stability, local convergence, and transformation sensitivity of a variant of the particle swarm optimization algorithm. IEEE Trans. Evol. Comput. **20**(3), 370–385 (2016)
36. Bosworth, J., Foo, N.Y., Zeigler, B.P.: Comparison of genetic algorithms with conjugate gradient methods. National Aeronautics and Space Administration (NASA CR-2093) (1972)
37. Boubaker, S.: Identification of nonlinear Hammerstein system using mixed integer-real coded particle swarm optimization: application to the electric daily peak-load forecasting. Nonlinear Dyn. **90**(2), 797–814 (2017)
38. Brameier, M.F., Banzhaf, W.: Linear Genetic Programming. Springer, New York (2007)
39. Bratton, D., Kennedy, J.: Defining a standard for particle swarm optimization. In: 2007 IEEE Swarm Intelligence Symposium, pp. 120–127. IEEE (2007)
40. Brewick, P.T., Masri, S.F.: An evaluation of data-driven identification strategies for complex nonlinear dynamic systems. Nonlinear Dyn. **85**(2), 1297–1318 (2016)
41. Brewick, P.T., Masri, S.F., Carboni, B., Lacarbonara, W.: Data-based nonlinear identification and constitutive modeling of hysteresis in NiTiNOL and steel strands. J. Eng. Mech. **142**(12), 04016107 (2016)
42. Burton, S.A., Makris, N., Konstantopoulos, I., Antsaklis, P.: Modeling the response of ER damper: phenomenology and emulation. J. Eng. Mech. **122**(9), 897–906 (1996)
43. Caraffini, F., Neri, F.: A study on rotation invariance in differential evolution. Swarm Evolut. Comput. **50**, 100436 (2018)
44. Carboni, B., Lacarbonara, W.: Nonlinear vibration absorber with pinched hysteresis: theory and experiments. J. Eng. Mech. **142**(5), 04016023 (2016)
45. Carboni, B., Lacarbonara, W., Auricchio, F.: Hysteresis of multiconfiguration assemblies of nitinol and steel strands: experiments and phenomenological identification. J. Eng. Mech. **141**(3), 04014135 (2014)
46. Carboni, B., Mancini, C., Lacarbonara, W.: Hysteretic beam model for steel wire ropes hysteresis identification. In: Structural Nonlinear Dynamics and Diagnosis. Springer

Proceedings in Physics, vol. 168, pp. 261–282. Springer (2015)

47. Carboni, B., Lacarbonara, W., Brewick, P.T., Masri, S.F.: Dynamical response identification of a class of nonlinear hysteretic systems. J. Intell. Mater. Syst. Struct. **29**(13), 2795–2810 (2018)
48. Carro-Pérez, I., Sánchez-López, C., González-Hernández, H.: Experimental verification of a memristive neural network. Nonlinear Dyn. **93**(4), 1823–1840 (2018)
49. Chaber, P., Ławryńczuk, M.: Pruning of recurrent neural models: an optimal brain damage approach. Nonlinear Dyn. **92**(2), 763–780 (2018)
50. Chang, C.C., Roschke, P.: Neural network modeling of a magnetorheological damper. J. Intell. Mater. Syst. Struct. **9**(9), 755–764 (1998)
51. Chang, C.C., Zhou, L.: Neural network emulation of inverse dynamics for a magnetorheological damper. J. Struct. Eng. **128**(2), 231–239 (2002)
52. Chang, W.D.: Parameter identification of Rossler's chaotic system by an evolutionary algorithm. Chaos Solitons Fractals **29**(5), 1047–1053 (2006)
53. Charalampakis, A., Dimou, C.: Identification of Bouc–Wen hysteretic systems using particle swarm optimization. Comput. Struct. **88**(21–22), 1197–1205 (2010)
54. Charalampakis, A., Dimou, C.: Comparison of evolutionary algorithms for the identification of Bouc–Wen hysteretic systems. J. Comput. Civil Eng. **29**(3), 04014053 (2013)
55. Charalampakis, A., Koumousis, V.: Identification of Bouc–Wen hysteretic systems by a hybrid evolutionary algorithm. J. Sound Vib. **314**(3–5), 571–585 (2008)
56. Chassiakos, A., Masri, S.: Modelling unknown structural systems through the use of neural networks. Earthq. Eng. Struct. Dyn. **25**(2), 117–128 (1996a)
57. Chassiakos, A.G., Masri, S.F.: Identification of structural systems by neural networks. Math. Comput. Simul. **40**(5–6), 637–656 (1996b)
58. Chelouah, R., Siarry, P.: A continuous genetic algorithm designed for the global optimization of multimodal functions. J. Heuristics **6**(2), 191–213 (2000)
59. Chen, C.M., Hsu, Y.C., Fung, R.F.: System identification of a Scott–Russell amplifying mechanism with offset driven by a piezoelectric actuator. Appl. Math. Model. **36**(6), 2788–2802 (2012)
60. Chen, F., Ding, Z., Lu, Z., Zeng, X.: Parameters identification for chaotic systems based on a modified Jaya algorithm. Nonlinear Dyn. **94**(4), 2307–2326 (2018)
61. Chisari, C., Amadio, C.: TOSCA: a tool for optimisation in structural and civil engineering analyses. Int. J. Adv. Struct. Eng. **10**(4), 401–419 (2018)
62. Chisari, C., Francavilla, A.B., Latour, M., Piluso, V., Rizzano, G., Amadio, C.: Critical issues in parameter calibration of cyclic models for steel members. Eng. Struct. **132**, 123–138 (2017)
63. Chisari, C., Rizzano, G., Amadio, C., Galdi, V.: Sensitivity analysis and calibration of phenomenological models for seismic analyses. Soil Dyn. Earthq. Eng. **109**, 10–22 (2018)
64. Choi, T.J., Ahn, C.W., An, J.: An adaptive Cauchy differential evolution algorithm for global numerical optimization. Sci. World J. **2013**, 969734 (2013)
65. Chungui, Z., Xinong, Z., Shilin, X., Tong, Z., Changchun, Z.: Hybrid modeling of wire cable vibration isolation system through neural network. Math. Comput. Simul. **79**(10), 3160–3173 (2009)
66. Chwastek, K., Szczyglowski, J.: Identification of a hysteresis model parameters with genetic algorithms. Math. Comput. Simul. **71**(3), 206–211 (2006)
67. Clerc, M.: The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), vol. 3, pp. 1951–1957. IEEE (1999)
68. Coelho, L.S., Guerra, F.A., Leite, J.V.: Multiobjective exponential particle swarm optimization approach applied to hysteresis parameters estimation. IEEE Trans. Magn. **48**(2), 283–286 (2012)
69. Coley, D.A.: An Introduction to Genetic Algorithms for Scientists and Engineers. World Scientific Publishing Company, Singapore (1999)
70. Consolo, G., Finocchio, G., Carpentieri, M., Azzerboni, B.: Removing numerical instabilities in the Preisach model identification using genetic algorithms. Physica B **372**(1–2), 91–96 (2006)
71. Crawford-Marks, R., Spector, L.: Size control via size fair genetic operators in the PushGP genetic programming system. In: Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation, pp. 733–739. Morgan Kaufmann Publishers Inc. (2002)
72. Cybenko, G.: Approximation by superpositions of a sigmoidal function. Math. Control Signals Syst. **2**(4), 303–314 (1989)
73. Das, S., Suganthan, P.N.: Differential evolution: a survey of the state-of-the-art. IEEE Trans. Evol. Comput. **15**(1), 4–31 (2011)
74. Das, S., Mullick, S.S., Suganthan, P.N.: Recent advances in differential evolution—an updated survey. Swarm Evolut. Comput. **27**, 1–30 (2016)
75. de Paula, N.C.G., Marques, F.D.: Multi-variable Volterra kernels identification using time-delay neural networks: application to unsteady aerodynamic loading. Nonlinear Dyn. **97**(1), 767–780 (2019)
76. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
77. Deep, K., Thakur, M.: A new crossover operator for real coded genetic algorithms. Appl. Math. Comput. **188**(1), 895–911 (2007)
78. Del Ser, J., Osaba, E., Molina, D., Yang, X.S., Salcedo-Sanz, S., Camacho, D., Das, S., Suganthan, P.N., Coello, C.A.C., Herrera, F.: Bio-inspired computation: where we stand and what's next. Swarm Evolut. Comput. **48**, 220–250 (2019)
79. Derkevorkian, A., Masri, S.F., Fujino, Y., Siringoringo, D.M.: Development and validation of nonlinear computational models of dispersed structures under strong earthquake excitation. Earthq. Eng. Struct. Dyn. **43**(7), 1089–1105 (2014)
80. Derkevorkian, A., Hernandez-Garcia, M., Yun, H.B., Masri, S.F., Li, P.: Nonlinear data-driven computational models for response prediction and change detection. Struct. Control Health Monit. **22**(2), 273–288 (2015)

81. Dewhirst, O., Simpson, D., Angarita, N., Allen, R.: Wiener-Hammerstein parameter estimation using differential evolution: application to limb reflex dynamics. In: International Conference on Bio-inspired Systems and Signal Processing, pp. 271–276 (2010)
82. Do, T., Tjahjowidodo, T., Lau, M.W.S., Phee, S.J.: A new approach of friction model for tendon-sheath actuated surgical systems: nonlinear modelling and parameter identification. Mech. Mach. Theory **85**, 14–24 (2015)
83. dos Santos, C.L.: A quantum particle swarm optimizer with chaotic mutation operator. Chaos Solitons Fractals **37**(5), 1409–1418 (2008)
84. dos Santos, C.L., Herrera, B.M.: Fuzzy identification based on a chaotic particle swarm optimization approach applied to a nonlinear yo–yo motion system. IEEE Trans. Ind. Electron. **54**(6), 3234–3245 (2007)
85. dos Santos, C.L., Pessôa, M.W.: Nonlinear identification using a B-spline neural network and chaotic immune approaches. Mech. Syst. Signal Process. **23**(8), 2418–2434 (2009a)
86. dos Santos, C.L., Pessôa, M.W.: Nonlinear model identification of an experimental ball-and-tube system using a genetic programming approach. Mech. Syst. Signal Process. **23**(5), 1434–1446 (2009b)
87. dos Santos, C.L., Mariani, V.C., Leite, J.V.: Solution of Jiles–Atherton vector hysteresis parameters estimation by modified differential evolution approaches. Expert Syst. Appl. **39**(2), 2021–2025 (2012)
88. Doglioni, A., Mancarella, D., Simeone, V., Giustolisi, O.: Inferring groundwater system dynamics from hydrological time-series data. Hydrol. Sci. J. Sci. Hydrol. **55**(4), 593–608 (2010)
89. Dokeroglu, T., Sevinc, E., Kucukyilmaz, T., Cosar, A.: A survey on new generation metaheuristic algorithms. Comput. Ind. Eng. **137**, 106040 (2019)
90. Dorigo, M., Gambardella, L.M.: Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans. Evol. Comput. **1**(1), 53–66 (1997)
91. Dotoli, M., Maione, G., Naso, D., Turchiano, B.: Genetic identification of dynamical systems with static nonlinearities. In: SMCia/01. Proceedings of the 2001 IEEE Mountain Workshop on Soft Computing in Industrial Applications (Cat. No. 01EX504), pp. 65–70. IEEE (2001)
92. Dowell, E.H., Epureanu, B.I.: Introduction. Nonlinear Dyn. **39**(1), 1–1 (2005)
93. Du, W., Miao, Q., Tong, L., Tang, Y.: Identification of fractional-order systems with unknown initial values and structure. Phys. Lett. A **381**(23), 1943–1949 (2017)
94. Du, W., Tong, L., Tang, Y.: Metaheuristic optimization-based identification of fractional-order systems under stable distribution noises. Phys. Lett. A **382**(34), 2313–2320 (2018)
95. Eberhart, R.C., Shi, Y.: Comparing inertia weights and constriction factors in particle swarm optimization. In: Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No. 00TH8512), vol. 1, pp. 84–88. IEEE (2000)
96. Ekárt, A., Nemeth, S.Z.: Selection based on the pareto nondomination criterion for controlling code growth in genetic programming. Genet. Program. Evol. Mach. **2**(1), 61–73 (2001)
97. Elman, J.L.: Finding structure in time. Cognit. Sci. **14**(2), 179–211 (1990)
98. Engelbrecht, A.P.: Computational Intelligence: An Introduction. Wiley, New York (2007)
99. Erdbrink, C.D., Krzhizhanovskaya, V.V.: Differential evolution for system identification of self-excited vibrations. J. Comput. Sci. **10**, 360–369 (2015)
100. Eshelman, L.J., Schaffer, J.D.: Real-coded genetic algorithms and interval-schemata. In: Foundations of Genetic Algorithms, vol. 2, pp. 187–202. Elsevier (1993)
101. Fiore, A., Mollaioli, F., Quaranta, G., Marano, G.C.: Seismic response prediction of reinforced concrete buildings through nonlinear combinations of intensity measures. Bull. Earthq. Eng. **16**(12), 6047–6076 (2018)
102. Foliente, G.C.: Hysteresis modeling of wood joints and structural systems. J. Struct. Eng. **121**(6), 1013–1022 (1995)
103. Formica, G., Milicchio, F.: Kinship-based differential evolution algorithm for unconstrained numerical optimization. Nonlinear Dyn. (2019). https://doi.org/10.1007/s11071-019-05358-y
104. Formica, G., Talò, M., Lanzara, G., Lacarbonara, W.: Parametric identification of carbon nanotube nanocomposites constitutive response. J. Appl. Mech. **86**(4), 041007 (2019)
105. Fu, J., Liao, G., Yu, M., Li, P., Lai, J.: NARX neural network modeling and robustness analysis of magnetorheological elastomer isolator. Smart Mater. Struct. **25**(12), 125019 (2016)
106. Fulginei, F.R., Salvini, A.: Softcomputing for the identification of the Jiles–Atherton model parameters. IEEE Trans. Magn. **41**(3), 1100–1108 (2005)
107. Fulginei, F.R., Salvini, A.: The flock of starlings optimization: influence of topological rules on the collective behavior of swarm intelligence. In: Computational Methods for the Innovative Design of Electrical Devices, pp. 129–145. Springer (2010)
108. Fulginei, F.R., Salvini, A.: Neural network approach for modelling hysteretic magnetic materials under distorted excitations. IEEE Trans. Magn. **48**(2), 307–310 (2012)
109. Fung, R.F., Lin, W.C.: System identification of a novel 6-DOF precision positioning table. Sens. Actuat. A **150**(2), 286–295 (2009)
110. Fung, R.F., Hsu, Y.L., Huang, M.S.: System identification of a dual-stage XY precision positioning table. Precis. Eng. **33**(1), 71–80 (2009)
111. Ganjefar, S., Tofighi, M.: Optimization of quantum-inspired neural network using memetic algorithm for function approximation and chaotic time series prediction. Neurocomputing **291**, 175–186 (2018)
112. Gao, F., Lee, J.J., Li, Z., Tong, H., Lü, X.: Parameter estimation for chaotic system with initial random noises by particle swarm optimization. Chaos Solitons Fractals **42**(2), 1286–1291 (2009)
113. Gao, F., Xj, L., Fx, F., Hq, T., Yb, Q., Yf, D., Balasingham, I., Hl, Z.: Parameter identification for Van der Pol-Duffing oscillator by a novel artificial bee colony algorithm with differential evolution operators. Appl. Math. Comput. **222**, 132–144 (2013)
114. Gao, F., Fei, F.X., Lee, X.J., Tong, H.Q., Deng, Y.F., Zhao, H.L.: Inversion mechanism with functional extrema model for identification incommensurate and hyper frac-

tional chaos via differential evolution. Expert Syst. Appl. **41**(4), 1915–1927 (2014a)

115. Gao, F., Xj, L., Fx, F., Hq, T., Yf, D., Hl, Z.: Identification time-delayed fractional order chaos with functional extrema model via differential evolution. Expert Syst. Appl. **41**(4), 1601–1608 (2014b)
116. Girosi, F., Jones, M., Poggio, T.: Regularization theory and neural networks architectures. Neural Comput. **7**(2), 219–269 (1995)
117. Giustolisi, O., Savic, D.: Advances in data-driven analyses and modelling using EPR-MOGA. J. Hydroinform. **11**(3–4), 225–236 (2009)
118. Giustolisi, O., Savic, D.A.: A symbolic data-driven technique based on evolutionary polynomial regression. J. Hydroinform. **8**(3), 207–222 (2006)
119. Goldberg, D.E.: Real-coded genetic algorithms, virtual alphabets, and blocking. Complex Syst. **5**(2), 139–167 (1991)
120. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
121. Gray, G.J., Li, Y., Murray-Smith, D., Sharman, K.: Structural system identification using genetic programming and a block diagram oriented simulation tool. Electron. Lett. **32**(15), 1422–1424 (1996)
122. Gray, G.J., Murray-Smith, D.J., Li, Y., Sharman, K.C., Weinbrenner, T.: Nonlinear model structure identification using genetic programming. Control Eng. Pract. **6**(11), 1341–1352 (1998)
123. Gu, G.Y., Li, C.X., Zhu, L.M., Su, C.Y.: Modeling and identification of piezoelectric-actuated stages cascading hysteresis nonlinearity with linear dynamics. IEEE/ASME Trans. Mechatron. **21**(3), 1792–1797 (2016a)
124. Gu, W., Yu, Y., Hu, W.: Parameter estimation of unknown fractional-order memristor-based chaotic systems by a hybrid artificial bee colony algorithm combined with differential evolution. Nonlinear Dyn. **84**(2), 779–795 (2016b)
125. Guerra, F.A., Coelho, L.S.: Multi-step ahead nonlinear identification of Lorenz's chaotic system using radial basis neural network with learning by clustering and particle swarm optimization. Chaos Solitons Fractals **35**(5), 967–979 (2008)
126. Gurney, K.: An Introduction to Neural Networks. UCL Press, London (1997)
127. Gutiérrez, G., Sanchis, A., Isasi, P., Molina, J.M., Galván, I.M.: Non-direct encoding method based on cellular automata to design neural network architectures. Comput. Inform. **24**(3), 225–247 (2012)
128. Ha, J.L., Kung, Y.S., Fung, R.F., Hsien, S.C.: A comparison of fitness functions for the identification of a piezoelectric hysteretic actuator based on the real-coded genetic algorithm. Sens. Actuat. A **132**(2), 643–650 (2006)
129. Hagan, M.T., Menhaj, M.B.: Training feedforward networks with the Marquardt algorithm. IEEE Trans. Neural Netw. **5**(6), 989–993 (1994)
130. Harik, G.R., Lobo, F.G., Goldberg, D.E.: The compact genetic algorithm. IEEE Trans. Evol. Comput. **3**(4), 287–297 (1999)
131. Haupt, R.L., Haupt, S.E.: Practical Genetic Algorithms. Wiley, New York (2004)
132. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice Hall, Upper Saddle River (1999)
133. He, Q., Wang, L., Liu, B.: Parameter estimation for chaotic systems by particle swarm optimization. Chaos Solitons Fractals **34**(2), 654–661 (2007)
134. He, S., Wu, Q., Wen, J., Saunders, J., Paton, R.: A particle swarm optimizer with passive congregation. Biosystems **78**(1–3), 135–147 (2004)
135. Hergli, K., Marouani, H., Zidi, M., Fouad, Y., Elshazly, M.: Identification of Preisach hysteresis model parameters using genetic algorithms. J. King Saud Univ. Sci. **31**(4), 746–752 (2017)
136. Hergli, K., Marouani, H., Zidi, M.: Numerical determination of Jiles–Atherton hysteresis parameters: magnetic behavior under mechanical deformation. Physica B **549**, 74–81 (2018)
137. Herrera, F., Lozano, M., Verdegay, J.L.: Tackling real-coded genetic algorithms: operators and tools for behavioural analysis. Artif. Intell. Rev. **12**(4), 265–319 (1998)
138. Hilgert, T., Vandevelde, L., Melkebeek, J.: Neural-network-based model for dynamic hysteresis in the magnetostriction of electrical steel under sinusoidal induction. IEEE Trans. Magn. **43**(8), 3462–3466 (2007)
139. Ho, W.H., Chou, J.H., Guo, C.Y.: Parameter identification of chaotic systems using improved differential evolution algorithm. Nonlinear Dyn. **61**(1–2), 29–41 (2010)
140. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
141. Holland, J.H.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. Michigan University Press, Michigan (1975)
142. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proc. Nat. Acad. Sci. **79**(8), 2554–2558 (1982)
143. Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Netw. **2**(5), 359–366 (1989)
144. Hu, W., Yu, Y., Gu, W.: Parameter estimation of fractional-order arbitrary dimensional hyperchaotic systems via a hybrid adaptive artificial bee colony algorithm with simulated annealing algorithm. Eng. Appl. Artif. Intell. **68**, 172–191 (2018)
145. Hu, W., Wen, G., Rahmani, A., Yu, Y.: Differential evolution-based parameter estimation and synchronization of heterogeneous uncertain nonlinear delayed fractional-order multi-agent systems with unknown leader. Nonlinear Dyn. **97**(2), 1087–1105 (2019)
146. Huang, C.C., Loh, C.H.: Nonlinear identification of dynamic systems using neural networks. Comput. Aided Civil Infrastruct. Eng. **16**(1), 28–41 (2001)
147. Huang, T., Mohan, A.S.: Micro-particle swarm optimizer for solving high dimensional optimization problems (μPSO for high dimensional optimization problems). Appl. Math. Comput. **181**(2), 1148–1154 (2006)
148. Huang, V., Qin, A., Suganthan, P.: Self-adaptive differential evolution algorithm for constrained real-parameter optimization. In: IEEE Congress on Evolutionary Computation, pp. 17–24 (2006)
149. Hung, S.L., Huang, C., Wen, C., Hsu, Y.: Nonparametric identification of a building structure from experimental

data using wavelet neural network. Comput. Aided Civil Infrastruct. Eng. **18**(5), 356–368 (2003)

150. Iacca, G., Caraffini, F., Neri, F.: Compact differential evolution light: high performance despite limited memory requirement and modest computational overhead. J. Comput. Sci. Technol. **27**(5), 1056–1076 (2012)
151. Ilonen, J., Kamarainen, J.K., Lampinen, J.: Differential evolution training algorithm for feed-forward neural networks. Neural Process. Lett. **17**(1), 93–105 (2003)
152. Irakoze, R., Yakoub, K., Kaddouri, A.: Identification of piezoelectric LuGre model based on particle swarm optimization and real-coded genetic algorithm. In: 2015 IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE), pp. 1451–1457. IEEE (2015)
153. Islam, S.M., Das, S., Ghosh, S., Roy, S., Suganthan, P.N.: An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization. IEEE Trans. Syst. Man Cybern. Part B (Cybernetics) **42**(2), 482–500 (2012)
154. Jafari, S., Golpayegani, S.M.R.H., Daliri, A.: Comment on 'Parameters identification of chaotic systems by quantum-behaved particle swarm optimization' [Int. J. Comput. Math. 86 (12)(2009), pp. 2225–2235]. Int. J. Comput. Math. **90**(5), 903–905 (2013a)
155. Jafari, S., Golpayegani, S.R.H., Darabad, M.R.: Comment on 'Parameter identification and synchronization of fractional-order chaotic systems' [Commun. Nonlinear Sci. Numer. Simulat. 2012; 17: 305–16]. Commun. Nonlinear Sci. Numer. Simul. **18**(3), 811–814 (2013b)
156. Jafari, S., Sprott, J.C., Pham, V.T., Golpayegani, S.M.R.H., Jafari, A.H.: A new cost function for parameter estimation of chaotic systems using return maps as fingerprints. Int. J. Bifurc. Chaos **24**(10), 1450134 (2014)
157. Jia, Z.Y., Liu, H.F., Wang, F.J., Ge, C.Y.: Research on a novel force sensor based on giant magnetostrictive material and its model. J. Alloys Compd. **509**(5), 1760–1767 (2011)
158. Jiang, X., Adeli, H.: Dynamic wavelet neural network for nonlinear identification of highrise buildings. Comput. Aided Civil Infrastruct. Eng. **20**(5), 316–330 (2005)
159. Jiang, Y., Lau, F.C., Wang, S., Tse, C.K.: Parameter identification of chaotic systems by a novel dual particle swarm optimization. Int. J. Bifurc. Chaos **26**(02), 1650024 (2016)
160. Jin, X., Shao, J., Zhang, X., An, W., Malekian, R.: Modeling of nonlinear system based on deep learning framework. Nonlinear Dyn. **84**(3), 1327–1340 (2016)
161. Kadirkamanathan, V., Selvarajah, K., Fleming, P.J.: Stability analysis of the particle dynamics in particle swarm optimizer. IEEE Trans. Evol. Comput. **10**(3), 245–255 (2006)
162. Kao, C.C., Fung, R.F.: Using the modified PSO method to identify a Scott–Russell mechanism actuated by a piezoelectric element. Mech. Syst. Signal Process. **23**(5), 1652–1661 (2009)
163. Karaboga, D., Basturk, B.: On the performance of artificial bee colony (ABC) algorithm. Appl. Soft Comput. **8**(1), 687–697 (2008)
164. Kaveh, A., Talatahari, S.: Size optimization of space trusses using Big Bang–Big Crunch algorithm. Comput. Struct. **87**(17–18), 1129–1140 (2009)
165. Kennedy, J., Eberhart, R.: Particle swarm optimization. Proc. IEEE Int. Conf. Neural Netw. **4**, 1942–1948 (1995)
166. Kerschen, G., Worden, K., Vakakis, A.F., Golinval, J.C.: Past, present and future of nonlinear system identification in structural dynamics. Mech. Syst. Signal Process. **20**(3), 505–592 (2006)
167. Khalid, M., Yusof, R., Joshani, M., Selamat, H., Joshani, M.: Nonlinear identification of a magneto-rheological damper based on dynamic neural networks. Comput. Aided Civil Infrastruct. Eng. **29**(3), 221–233 (2014)
168. Ko, Y.R., Hwang, Y., Chae, M., Kim, T.H.: Direct identification of generalized Prandtl–Ishlinskii model inversion for asymmetric hysteresis compensation. ISA Trans. **70**, 209–218 (2017)
169. Kosmatopoulos, E., Smyth, A., Masri, S., Chassiakos, A.: Robust adaptive neural estimation of restoring forces in nonlinear structures. J. Appl. Mech. **68**(6), 880–893 (2001)
170. Koumousis, V.K., Katsaras, C.P.: A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance. IEEE Trans. Evol. Comput. **10**(1), 19–28 (2006)
171. Koza, J.R.: Genetic Programming: on the Programming of Computers by Means of Natural Selection, vol. 1. MIT Press, Cambridge (1992)
172. Krishnakumar, K.: Micro-genetic algorithms for stationary and non-stationary function optimization. Intell. Control Adapt. Syst. Int. Soc. Opt. Photon. **1196**, 289–297 (1990)
173. Kwok, N., Ha, Q., Nguyen, M., Li, J., Samali, B.: Bouc–Wen model parameter identification for a MR fluid damper using computationally efficient GA. ISA Trans. **46**(2), 167–179 (2007)
174. Kyprianou, A., Worden, K., Panet, M.: Identification of hysteretic systems using the differential evolution algorithm. J. Sound Vib. **248**(2), 289–314 (2001)
175. La Cava, W., Helmuth, T., Spector, L., Danai, K.: Genetic programming with epigenetic local search. In: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, pp. 1055–1062, ACM (2015)
176. La Cava, W., Danai, K., Spector, L., Fleming, P., Wright, A., Lackner, M.: Automatic identification of wind turbine models using evolutionary multiobjective optimization. Renew. Energy **87**, 892–902 (2016)
177. Lacarbonara, W.: Nonlinear Structural Dynamics. Theory, Modeling, and Dynamical Phenomena. Springer, New York (2013)
178. Langdon, W.B., Poli, R., McPhee, N.F., Koza, J.R.: Genetic programming: an introduction and tutorial, with a survey of techniques and applications. In: Fulcher, J., Jain, L.C. (eds.) Computational Intelligence: A Compendium. Studies in Computational Intelligence, vol. 115. Springer, Berlin (2008)
179. Lanzarini, L., Leza, V., De Giusti, A.: Particle swarm optimization with variable population size. In: International Conference on Artificial Intelligence and Soft Computing, pp. 438–449. Springer, New York (2008)
180. Laudani, A., Fulginei, F.R., Salvini, A.: Comparative analysis of Bouc–Wen and Jiles–Atherton models under symmetric excitations. Physica B **435**, 134–137 (2014a)
181. Laudani, A., Fulginei, F.R., Salvini, A.: Bouc–Wen hysteresis model identification by the metric-topological evolutionary optimization. IEEE Trans. Magn. **50**(2), 621–624 (2014b)

182. Le Cun, Y., Jackel, L.D., Boser, B., Denker, J.S., Graf, H.P., Guyon, I., Henderson, D., Howard, R.E., Hubbard, W.: Handwritten digit recognition: applications of neural network chips and automatic learning. IEEE Commun. Mag. **27**(11), 41–46 (1989)
183. Leboucher, C., Shin, H.S., Siarry, P., Le Ménec, S., Chelouah, R., Tsourdos, A.: Convergence proof of an enhanced particle swarm optimisation method integrated with evolutionary game theory. Inf. Sci. **346**, 389–411 (2016)
184. Leite, J., Avila, S., Batistela, N., Carpes, W., Sadowski, N., Kuo-Peng, P., Bastos, J.: Real coded genetic algorithm for Jiles–Atherton model parameters identification. IEEE Trans. Magn. **40**(2), 888–891 (2004)
185. Leng, D., Xu, K., Ma, Y., Liu, G., Sun, L.: Modeling the behaviors of magnetorheological elastomer isolator in shear-compression mixed mode utilizing artificial neural network optimized by fuzzy algorithm (ANNOFA). Smart Mater. Struct. **27**(11), 115026 (2018)
186. Li, C., Zhou, J., Xiao, J., Xiao, H.: Parameters identification of chaotic system by chaotic gravitational search algorithm. Chaos Solitons Fractals **45**(4), 539–547 (2012)
187. Li, H.X.: Identification of Hammerstein models using genetic algorithms. IEE Proc. Control Theory Appl. **146**(6), 499–504 (1999)
188. Li, K., Kou, J., Zhang, W.: Deep neural network for unsteady aerodynamic and aeroelastic modeling across multiple mach numbers. Nonlinear Dyn. **96**(3), 2157–2177 (2019)
189. Li, L., Wang, L., Liu, L.: An effective hybrid PSOSA strategy for optimization and its application to parameter estimation. Appl. Math. Comput. **179**(1), 135–146 (2006)
190. Li, P., Xiao, H., Shang, F., Tong, X., Li, X., Cao, M.: A hybrid quantum-inspired neural networks with sequence inputs. Neurocomputing **117**, 81–90 (2013)
191. Li, X., Yin, M.: Parameter estimation for chaotic systems by hybrid differential evolution algorithm and artificial bee colony algorithm. Nonlinear Dyn. **77**(1–2), 61–71 (2014)
192. Li, X., Fc, L., Liu, X., Guo, Y.: Parameter identification and optimisation for a class of fractional-order chaotic system with time delay. Int. J. Model. Ident. Control **29**(2), 153–162 (2018)
193. Liang, Y., Feng, D., Cooper, J.: Identification of restoring forces in non-linear vibration systems using fuzzy adaptive neural networks. J. Sound Vib. **242**(1), 47–58 (2001)
194. Liang, Y.C., Zhou, C.G., Wang, Z.S.: Identification of restoring forces in non-linear vibration systems based on neural networks. J. Sound Vib. **206**, 103–108 (1997)
195. Lin, C.J., Yau, H.T., Lee, C.Y., Tung, K.H.: System identification and semiactive control of a squeeze-mode magnetorheological damper. IEEE/ASME Trans. Mechatron. **18**(6), 1691–1701 (2013)
196. Lin, C.J., Lin, C.R., Yu, S.K., Chen, C.T.: Hysteresis modeling and tracking control for a dual pneumatic artificial muscle system using Prandtl–Ishlinskii model. Mechatronics **28**, 35–45 (2015)
197. Lin, J.: Oppositional backtracking search optimization algorithm for parameter identification of hyperchaotic systems. Nonlinear Dyn. **80**(1–2), 209–219 (2015)
198. Ling, S.H., Leung, F.F.: An improved genetic algorithm with average-bound crossover and wavelet mutation operations. Soft. Comput. **11**(1), 7–31 (2007)
199. Liu, F., Li, X., Liu, X., Tang, Y.: Parameter identification of fractional-order chaotic system with time delay via multi-selection differential evolution. Syst. Sci. Control Eng. **5**(1), 42–48 (2017)
200. Liu, J.: On setting the control parameter of the differential evolution method. In: Proceedings of the 8th International Conference on Soft Computing, pp. 11–18 (2002)
201. Liu, J., Xu, W., Sun, J.: Quantum-behaved particle swarm optimization with mutation operator. In: 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05), p. 4. IEEE (2005)
202. Liu, J., Xu, W., Sun, J.: Nonlinear system identification of Hammerstien and Wiener model using swarm intelligence. In: 2006 IEEE International Conference on Information Acquisition, pp. 1219–1223. IEEE (2006)
203. Liu, Y., Yang, S., Liao, Y.: A quantizing method for determination of controlled damping parameters of magnetorheological damper models. J. Intell. Mater. Syst. Struct. **22**(18), 2127–2136 (2011)
204. Liu, Y., Gao, X., Li, Y.: Giant magnetostrictive actuator nonlinear dynamic Jiles–Atherton model. Sens. Actuat. A **250**, 7–14 (2016)
205. Long, Z., Wang, R., Fang, J., Dai, X., Li, Z.: Hysteresis compensation of the Prandtl–Ishlinskii model for piezoelectric actuators using modified particle swarm optimization with chaotic map. Rev. Sci. Instrum. **88**(7), 075003 (2017)
206. Lu, H., Wen, X., Lan, L., An, Y., Li, X.: A self-adaptive genetic algorithm to estimate JA model parameters considering minor loops. J. Magn. Magn. Mater. **374**, 502–507 (2015)
207. Luke, S.: Two fast tree-creation algorithms for genetic programming. IEEE Trans. Evol. Comput. **4**(3), 274–283 (2000)
208. Luke, S., Balan, G.C., Panait, L.: Population implosion in genetic programming. In: Genetic and Evolutionary Computation Conference, pp. 1729–1739. Springer (2003)
209. Ma, F., Ng, C., Ajavakom, N.: On system identification and response prediction of degrading structures. Struct. Control Health Monit. **13**(1), 347–364 (2006)
210. Ma, J., Tang, J.: A review for dynamics of collective behaviors of network of neurons. Sci. China Technol. Sci. **58**(12), 2038–2045 (2015)
211. Ma, J., Tang, J.: A review for dynamics in neuron and neuronal network. Nonlinear Dyn. **89**(3), 1569–1578 (2017)
212. Ma, J., Zq, Y., Lj, Y., Tang, J.: A physical view of computational neurodynamics. J. Zhejiang Univ. Sci. A **20**(9), 639–659 (2019)
213. Ma, L., Khorasani, K.: New training strategies for constructive neural networks with application to regression problems. Neural Netw. **17**(4), 589–609 (2004)
214. Mannarino, A., Mantegazza, P.: Nonlinear aeroelastic reduced order modeling by recurrent neural networks. J. Fluids Struct. **48**, 103–121 (2014)
215. Marano, G.C., Quaranta, G.: Fuzzy-based robust structural optimization. Int. J. Solids Struct. **45**(11–12), 3544–3557 (2008)
216. Marano, G.C., Quaranta, G., Greco, R.: Multi-objective optimization by genetic algorithm of structural systems subject to random vibrations. Struct. Multidiscip. Optim. **39**(4), 385–399 (2009a)

217. Marano, G.C., Quaranta, G., Monti, G.: Genetic algorithms in mechanical systems identification: state-of-the-art review. In: Topping, B., Tsompanakis, Y. (eds.) Soft Computing in Civil and Structural Engineering, vol. 2, pp. 43–72. Saxe-Coburg Publications, Stirlingshire (2009b)
218. Marano, G.C., Quaranta, G., Monti, G.: Modified genetic algorithm for the dynamic identification of structural systems using incomplete measurements. Comput. Aided Civil Infrastruct. Eng. **26**(2), 92–110 (2011)
219. Marion, R., Scorretti, R., Siauve, N., Raulet, M.A., Krahenbuhl, L.: Identification of Jiles–Atherton model parameters using particle swarm optimization. IEEE Trans. Magn. **44**(6), 894–897 (2008)
220. Maruta, I., Kim, T.H., Song, D., Sugie, T.: Synthesis of fixed-structure robust controllers using a constrained particle swarm optimizer with cyclic neighborhood topology. Expert Syst. Appl. **40**(9), 3595–3605 (2013)
221. Masri, S., Bekey, G., Safford, F.: A global optimization algorithm using adaptive random search. Appl. Math. Comput. **7**(4), 353–375 (1980)
222. Masri, S., Chassiakos, A., Caughey, T.: Structure-unknown non-linear dynamic systems: identification through neural networks. Smart Mater. Struct. **1**(1), 45 (1992)
223. Masri, S., Chassiakos, A., Caughey, T.: Identification of nonlinear dynamic systems using neural networks. J. Appl. Mech. **60**(1), 123–133 (1993)
224. Masri, S., Smyth, A., Chassiakos, A., Nakamura, M., Caughey, T.: Training neural networks by adaptive random search techniques. J. Eng. Mech. **125**(2), 123–132 (1999)
225. Masri, S., Smyth, A., Chassiakos, A., Caughey, T., Hunter, N.: Application of neural networks for detection of changes in nonlinear systems. J. Eng. Mech. **126**(7), 666–676 (2000)
226. McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. **5**(4), 115–133 (1943)
227. Mehmood, A., Aslam, M.S., Chaudhary, N.I., Zameer, A., Raja, M.A.Z.: Parameter estimation for hammerstein control autoregressive systems using differential evolution. SIViP **12**(8), 1603–1610 (2018)
228. Mehmood, A., Chaudhary, N.I., Zameer, A., Raja, M.A.Z.: Novel computing paradigms for parameter estimation in Hammerstein controlled auto regressive auto regressive moving average systems. Appl. Soft Comput. **80**, 263–284 (2019)
229. Mininno, E., Cupertino, F., Naso, D.: Real-valued compact genetic algorithms for embedded microcontroller optimization. IEEE Trans. Evol. Comput. **12**(2), 203–219 (2008)
230. Mininno, E., Neri, F., Cupertino, F., Naso, D.: Compact differential evolution. IEEE Trans. Evol. Comput. **15**(1), 32–54 (2010)
231. Mirjalili, S., Hashim, S.Z.M., Sardroudi, H.M.: Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm. Appl. Math. Comput. **218**(22), 11125–11137 (2012)
232. Mitchell, M.: An Introduction to Genetic Algorithms. MIT press (1998)
233. Modares, H., Alfi, A., Fateh, M.M.: Parameter identification of chaotic dynamic systems through an improved particle swarm optimization. Expert Syst. Appl. **37**(5), 3714–3720 (2010)
234. Montana, D.J., Davis, L.: Training feedforward neural networks using genetic algorithms. In: Proceedings of the 11th International Joint Conference on Artificial Intelligence, vol.1, pp. 762–767 (1989)
235. Monti, G., Quaranta, G., Marano, G.C.: Genetic-algorithm-based strategies for dynamic identification of nonlinear systems with noise-corrupted response. J. Comput. Civil Eng. **24**(2), 173–187 (2009)
236. Muller, S.D., Marchetto, J., Airaghi, S., Kournoutsakos, P.: Optimization based on bacterial chemotaxis. IEEE Trans. Evol. Comput. **6**(1), 16–29 (2002)
237. Murray, A.F., Edwards, P.J.: Enhanced MLP performance and fault tolerance resulting from synaptic weight noise during training. IEEE Trans. Neural Netw. **5**(5), 792–802 (1994)
238. Mechanical Systems and Signal Processing. Document guidelines for machine learning papers in MSSP. https://www.journals.elsevier.com/mechanical-systems-and-signal-processing (2019). Accessed 28 July 2019
239. Naitali, A., Giri, F.: Wiener–Hammerstein system identification—an evolutionary approach. Int. J. Syst. Sci. **47**(1), 45–61 (2016)
240. Nam, D.N.C., Ahn, K.K.: Identification of an ionic polymer metal composite actuator employing Preisach type fuzzy NARX model and particle swarm optimization. Sens. Actuat. A **183**, 105–114 (2012)
241. Narendra, K.S., Parthasarathy, K.: Identification and control of dynamical systems using neural networks. IEEE Trans. Neural Netw. **1**(1), 4–27 (1990)
242. Nayfeh, A.H., Pai, F.: Linear and Nonlinear Structural Mechanics. Wiley, New York (2004)
243. Neri, F., Cotta, C.: Memetic algorithms and memetic computing optimization: a literature review. Swarm Evolut. Comput. **2**, 1–14 (2012)
244. Neri, F., Tirronen, V.: Recent advances in differential evolution: a survey and experimental analysis. Artif. Intell. Rev. **33**(1–2), 61–106 (2010)
245. Neri, F., Mininno, E., Iacca, G.: Compact particle swarm optimization. Inf. Sci. **239**, 96–121 (2013)
246. Noël, J.P., Kerschen, G.: Nonlinear system identification in structural dynamics: 10 more years of progress. Mech. Syst. Signal Process. **83**, 2–35 (2017)
247. Ojha, V.K., Abraham, A., Snášel, V.: Metaheuristic design of feedforward neural networks: a review of two decades of research. Eng. Appl. Artif. Intell. **60**, 97–116 (2017)
248. Ong, P., Zainuddin, Z.: Optimizing wavelet neural networks using modified cuckoo search for multi-step ahead chaotic time series prediction. Appl. Soft Comput. **80**, 374–386 (2019)
249. Opara, K.R., Arabas, J.: Differential evolution: a survey of theoretical analyses. Swarm Evolut. Comput. **44**, 546–558 (2019)
250. Ortiz, G.A., Alvarez, D.A., Bedoya-RuíZ, D.: Identification of Bouc–Wen type models using multi-objective optimization algorithms. Comput. Struct. **114**, 121–132 (2013)
251. Panahi, S., Jafari, S., Pham, V.T., Kingni, S.T., Zahedi, A., Sedighy, S.H.: Parameter identification of a chaotic circuit

with a hidden attractor using Krill herd optimization. Int. J. Bifurc. Chaos **26**(13), 1650221 (2016)

252. Pang, H., Zhang, X., Xu, Z.: Adaptive backstepping-based tracking control design for nonlinear active suspension system with parameter uncertainties and safety constraints. ISA Trans. (2018)
253. Paquali, M., Lacarbonara, W.: A geometrically exact formulation for thin multi-layered laminated composite plates: theory and experiment. Compos. Struct. **93**, 1649–1663 (2011)
254. Pei, J.S., Smyth, A.W.: New approach to designing multilayer feedforward neural network architecture for modeling nonlinear restoring forces. I: Formulation. J. Eng. Mech. **132**(12), 1290–1300 (2006a)
255. Pei, J.S., Smyth, A.W.: New approach to designing multilayer feedforward neural network architecture for modeling nonlinear restoring forces. II: Applications. J. Eng. Mech. **132**(12), 1301–1312 (2006b)
256. Pei, J.S., Smyth, A., Kosmatopoulos, E.: Analysis and modification of volterra/wiener neural networks for the adaptive identification of non-linear hysteretic dynamic systems. J. Sound Vib. **275**(3–5), 693–718 (2004)
257. Pei, J.S., Wright, J.P., Smyth, A.W.: Mapping polynomial fitting into feedforward neural networks for modeling nonlinear dynamic systems and beyond. Comput. Methods Appl. Mech. Eng. **194**(42–44), 4481–4505 (2005)
258. Pei, J.S., Carboni, B., Lacarbonara, W.: Mem-models as building blocks for simulation and identification of hysteretic systems. In: Book of Abstracts of the 1st International Nonlinear Dynamics Conference, pp. 325–327 (2019)
259. Pelliciari, M., Marano, G.C., Cuoghi, T., Briseghella, B., Lavorato, D., Tarantino, A.M.: Parameter identification of degrading and pinched hysteretic systems using a modified Bouc–Wen model. Struct. Infrastruct. Eng. **14**(12), 1573–1585 (2018)
260. Peng, B., Liu, B., Zhang, F.Y., Wang, L.: Differential evolution algorithm-based parameter estimation for chaotic systems. Chaos Solitons Fractals **39**(5), 2110–2118 (2009)
261. Perisic, N., Green, P.L., Worden, K., Kirkegaard, P.H.: Identification of time-varying nonlinear systems using differential evolution algorithm. In: Topics in Modal Analysis, vol. 7, pp. 575–583. Springer (2014)
262. Perry, M., Koh, C., Choo, Y.: Modified genetic algorithm strategy for structural identification. Comput. Struct. **84**(8–9), 529–540 (2006)
263. Pham, D.T., Liu, X.: Identification of linear and nonlinear dynamic systems using recurrent neural networks. Artif. Intell. Eng. **8**(1), 67–75 (1993)
264. Piotrowski, A.P.: Differential evolution algorithms applied to neural network training suffer from stagnation. Appl. Soft Comput. **21**, 382–406 (2014)
265. Pitts, W., McCulloch, W.S.: How we know universals the perception of auditory and visual forms. Bull. Math. Biophy. **9**(3), 127–147 (1947)
266. Poli, R.: New Ideas in Optimization. Parallel Distributed Genetic Programming, pp. 403–432. McGraw-Hill Ltd., Maidenhead (1999)
267. Poli, R.: A simple but theoretically-motivated method to control bloat in genetic programming. In: European Conference on Genetic Programming, pp. 204–217. Springer, New York (2003)
268. Poli, R., Vanneschi, L., Langdon, W.B., McPhee, N.F.: Theoretical results in genetic programming: the next ten years? Genet. Program. Evol. Mach. **11**(3–4), 285–320 (2010)
269. Prawin, J., Rao, A.R.M., Lakshmi, K.: Nonlinear parametric identification strategy combining reverse path and hybrid dynamic quantum particle swarm optimization. Nonlinear Dyn. **84**(2), 797–815 (2016)
270. Prechelt, L.: Automatic early stopping using cross validation: quantifying the criteria. Neural Netw. **11**(4), 761–767 (1998)
271. Qin, H., Bu, N., Chen, W., Yin, Z.: An asymmetric hysteresis model and parameter identification method for piezoelectric actuator. Mathem. Probl. Eng. (2014)
272. Quaranta, G., Monti, G., Marano, G.C.: Parameters identification of Van der Pol–Duffing oscillators via particle swarm optimization and differential evolution. Mech. Syst. Signal Process. **24**(7), 2076–2095 (2010)
273. Quaranta, G., Fiore, A., Marano, G.C.: Optimum design of prestressed concrete beams using constrained differential evolution algorithm. Struct. Multidiscip. Optim. **49**(3), 441–453 (2014a)
274. Quaranta, G., Marano, G.C., Greco, R., Monti, G.: Parametric identification of seismic isolators using differential evolution and particle swarm optimization. Appl. Soft Comput. **22**, 458–464 (2014b)
275. Rahman, M.A., Mamun, A.A., Yao, K.: Analysis and compensation of hysteresis of PZT micro-actuator used in high precision dual-stage servo system. Int. J. Mechatron. Autom. **5**(1), 58–68 (2015)
276. Raja, M.A.Z., Shah, A.A., Mehmood, A., Chaudhary, N.I., Aslam, M.S.: Bio-inspired computational heuristics for parameter estimation of nonlinear hammerstein controlled autoregressive system. Neural Comput. Appl. **29**(12), 1455–1474 (2018)
277. Rao, R.: Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems. Int. J. Ind. Eng. Comput. **7**(1), 19–34 (2016)
278. Ratnaweera, A., Halgamuge, S.K., Watson, H.C.: Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans. Evol. Comput. **8**(3), 240–255 (2004)
279. Rodríguez-Vázquez, K., Fleming, P.J.: Evolution of mathematical models of chaotic systems based on multiobjective genetic programming. Knowl. Inf. Syst. **8**(2), 235–256 (2005)
280. Rodriguez-Vazquez, K., Fonseca, C.M., Fleming, P.J.: Identifying the structure of nonlinear dynamic systems using multiobjective genetic programming. IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. **34**(4), 531–545 (2004)
281. Rojas, I., Pomares, H., Bernier, J.L., Ortega, J., Pino, B., Pelayo, F.J., Prieto, A.: Time series analysis using normalized PG-RBF network with regression weights. Neurocomputing **42**(1–4), 267–285 (2002)
282. Rojas, R.: Neural Networks: A Systematic Introduction. Springer, Berlin (1996)
283. Rubio-Solis, A., Melin, P., Martinez-Hernandez, U., Panoutsos, G.: General type-2 radial basis function neural

network: a data-driven fuzzy model. IEEE Trans. Fuzzy Syst. **27**(2), 333–347 (2018)

284. Ruder, S.: An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747 (2016)
285. Rumelhart, D.E., Hintonf, G.E.: Learning representations by back-propagating errors. Nature **323**, 9 (1986)
286. Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach. Pearson Education Limited, Malaysia (2016)
287. Salehinejad, H., Rahnamayan, S., Tizhoosh, H.R.: Micro-differential evolution: diversity enhancement and a comparative study. Appl. Soft Comput. **52**, 812–833 (2017)
288. Saliah, H., Lowther, D., Forghani, B.: A neural network model of magnetic hysteresis for computational magnetics. IEEE Trans. Magn. **33**(5), 4146–4148 (1997)
289. Sarban, R., Oubaek, J., Kristjánsdóttir, G.R., Jones, R.W.: Hysteresis modelling of a core-free EAP tubular actuator. In: International Society for Optics and Photonics on Electroactive Polymer Actuators and Devices (EAPAD), vol. 7287, p. 728717 (2009)
290. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2015)
291. Schmidt, M., Lipson, H.: Age-fitness pareto optimization. In: Genetic Programming Theory and Practice VIII, pp. 129–146. Springer, New York (2011)
292. Schmitt, L.M.: Theory of genetic algorithms. Theoret. Comput. Sci. **259**(1–2), 1–61 (2001)
293. Schmitt, L.M.: Theory of genetic algorithms II: models for genetic operators over the string-tensor representation of populations and convergence to global optima for arbitrary fitness function under scaling. Theoret. Comput. Sci. **310**(1–3), 181–231 (2004)
294. Scodeggio, A., Quaranta, G., Marano, G.C., Monti, G., Fleischman, R.B.: Optimization of force-limiting seismic devices connecting structural subsystems. Comput. Struct. **162**, 16–27 (2016)
295. Sengupta, P., Li, B.: Modified Bouc–Wen model for hysteresis behavior of RC beam-column joints with limited transverse reinforcement. Eng. Struct. **46**, 392–406 (2013)
296. Sengupta, S., Basak, S., Peters, R.: Particle swarm optimization: a survey of historical and recent developments with hybridization perspectives. Mach. Learn. Knowl. Extract. **1**(1), 157–191 (2018)
297. Serpico, C., Visone, C.: Magnetic hysteresis modeling via feed-forward neural networks. IEEE Trans. Magn. **34**(3), 623–628 (1998)
298. Sexton, R.S., Dorsey, R.E., Johnson, J.D.: Beyond backpropagation: using simulated annealing for training neural networks. J. Organ. End User Comput. **11**(3), 3–10 (1999)
299. Shi, Y., Eberhart, R.: A modified particle swarm optimizer. In: 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360), pp. 69–73. IEEE (1998)
300. Shi, Y., Eberhart, R.C.: Empirical study of particle swarm optimization. In: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), vol. 3, pp. 1945–1950. IEEE (1999)
301. Shu, G., Li, Z.: Parametric identification of the Bouc–Wen model by a modified genetic algorithm: application to evaluation of metallic dampers. Earthq. Struct. **13**(4), 397–407 (2017)
302. Siddique, N., Adeli, H.: Computational Intelligence: Synergies of Fuzzy Logic, Neural Networks and Evolutionary Computing. Wiley, New York (2013)
303. Sireteanu, T., Giuclea, M., Mitu, A.: Identification of an extended Bouc–Wen model with application to seismic protection through hysteretic devices. Comput. Mech. **45**(5), 431–441 (2010)
304. Sivanandam, S., Deepa, S.: Introduction to Genetic Algorithms. Springer, New York (2007)
305. Sivaselvan, M.V., Reinhorn, A.M.: Hysteretic models for deteriorating inelastic structures. J. Eng. Mech. **126**(6), 633–640 (2000)
306. Smith, J.F., Nguyen, T.H.: Guiding genetic program based data mining using fuzzy rules. In: International Conference on Intelligent Data Engineering and Automated Learning, pp. 1337–1345. Springer, New York (2006)
307. Song, G., Chaudhry, V., Batur, C.: A neural network inverse model for a shape memory alloy wire actuator. J. Intell. Mater. Syst. Struct. **14**(6), 371–377 (2003)
308. Sörensen, K.: Metaheuristics—the metaphor exposed. Int. Trans. Oper. Res. **22**(1), 3–18 (2015)
309. Stevanović, N., Green, P.L., Worden, K., Kirkegaard, P.H.: Friction estimation in wind turbine blade bearings. Struct. Control Health Monit. **23**(1), 103–122 (2016)
310. Storn, R., Price, K.: Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. **11**(4), 341–359 (1997)
311. Sun, J., Liu, X.: A novel APSO-aided maximum likelihood identification method for Hammerstein systems. Nonlinear Dyn. **73**(1–2), 449–462 (2013)
312. Sun, J., Zhao, J., Wu, X., Fang, W., Cai, Y., Xu, W.: Parameter estimation for chaotic systems with a drift particle swarm optimization method. Phys. Lett. A **374**(28), 2816–2822 (2010)
313. Szabó, Z., Füzi, J.: Implementation and identification of Preisach type hysteresis models with Everett function in closed form. J. Magn. Magn. Mater. **406**, 251–258 (2016)
314. Talatahari, S., Rahbari, N.M., Kaveh, A.: A new hybrid optimization algorithm for recognition of hysteretic nonlinear systems. KSCE J. Civil Eng. **17**(5), 1099–1108 (2013)
315. Tang, Y., Zhang, X., Hua, C., Li, L., Yang, Y.: Parameter identification of commensurate fractional-order chaotic system via differential evolution. Phys. Lett. A **376**(4), 457–464 (2012)
316. Toman, M., Stumberger, G., Dolinar, D.: Parameter identification of the Jiles–Atherton hysteresis model using differential evolution. IEEE Trans. Magn. **44**(6), 1098–1101 (2008)
317. Trelea, I.C.: The particle swarm optimization algorithm: convergence analysis and parameter selection. Inf. Process. Lett. **85**(6), 317–325 (2003)
318. Truong, B.N.M., Nam, D.N.C., Ahn, K.K.: Hysteresis modeling and identification of a dielectric electro-active polymer actuator using an APSO-based nonlinear Preisach NARX fuzzy model. Smart Mater. Struct. **22**(9), 095004 (2013)
319. Tsai, J.T., Chou, J.H., Liu, T.K.: Tuning the structure and parameters of a neural network by using hybrid Taguchi-genetic algorithm. IEEE Trans. Neural Netw. **17**(1), 69–80 (2006)

320. Tudón-Martínez, J., Lozoya-Santos, J.J., Morales-Menendez, R., Ramirez-Mendoza, R.: An experimental artificial-neural-network-based modeling of magneto-rheological fluid dampers. Smart Mater. Struct. **21**(8), 085007 (2012)
321. Vlachas, P.R., Byeon, W., Wan, Z.Y., Sapsis, T.P., Koumoutsakos, P.: Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. Proc. R. Soc. A Math. Phys. Eng. Sci. **474**(2213), 20170844 (2018)
322. Wang, C., Xiong, L., Sun, J., Yao, W.: Memristor-based neural networks with weight simultaneous perturbation training. Nonlinear Dyn. **95**(4), 2893–2906 (2019a)
323. Wang, D., Liao, W.: Modeling and control of magnetorheological fluid dampers using neural networks. Smart Mater. Struct. **14**(1), 111 (2004)
324. Wang, G., Chen, G., Bai, F.: Modeling and identification of asymmetric Bouc–Wen hysteresis for piezoelectric actuator via a novel differential evolution algorithm. Sens. Actuat. A **235**, 105–118 (2015)
325. Wang, H., Rahnamayan, S., Wu, Z.: Adaptive differential evolution with variable population size for solving high-dimensional problems. In: 2011 IEEE Congress of Evolutionary Computation (CEC), pp. 2626–2632. IEEE (2011a)
326. Wang, L., Xu, Y., Li, L.: Parameter identification of chaotic systems by hybrid Nelder–Mead simplex search and differential evolution algorithm. Expert Syst. Appl. **38**(4), 3238–3245 (2011b)
327. Wang, Q., Zheng, Y., Ma, J.: Cooperative dynamics in neuronal networks. Chaos Solitons Fractals **56**, 19–27 (2013)
328. Wang, R., Kalnay, E., Balachandran, B.: Neural machine-based forecasting of chaotic dynamics. Nonlinear Dyn. (2019b). https://doi.org/10.1007/s11071-019-05127-x
329. Wang, Y., Cai, Z., Zhang, Q.: Differential evolution with composite trial vector generation strategies and control parameters. IEEE Trans. Evol. Comput. **15**(1), 55–66 (2011c)
330. Wei, J., Yu, Y., Cai, D.: Identification of uncertain incommensurate fractional-order chaotic systems using an improved quantum-behaved particle swarm optimization algorithm. J. Comput. Nonlinear Dyn. **13**(5), 051004 (2018)
331. Wenjing, Z.: Parameter identification of LuGre friction model in servo system based on improved particle swarm optimization algorithm. In: 2007 Chinese Control Conference, pp. 135–139. IEEE (2007)
332. Whittle, P.: A stochastic model of an artificial neuron. Adv. Appl. Probab. **23**(4), 809–822 (1991)
333. Wilson, P.R., Ross, J.N., Brown, A.D.: Optimizing the Jiles–Atherton model of hysteresis by a genetic algorithm. IEEE Trans. Magn. **37**(2), 989–993 (2001)
334. Worden, K., Manson, G.: On the identification of hysteretic systems. Part I: Fitness landscapes and evolutionary identification. Mech. Syst. Signal Process. **29**, 201–212 (2012)
335. Worden, K., Tomlinson, G.: Modeling and classification of non-linear systems using neural networks-I. Simul. Mech. Syst. Signal Process. **8**(3), 319–356 (1994)
336. Worden, K., Staszewski, W.J., Hensman, J.J.: Natural computing for mechanical systems research: a tutorial overview. Mech. Syst. Signal Process. **25**(1), 4–111 (2011)
337. Worden, K., Barthorpe, R., Cross, E., Dervilis, N., Holmes, G., Manson, G., Rogers, T.: On evolutionary system identification with applications to nonlinear benchmarks. Mech. Syst. Signal Process. **112**, 194–232 (2018)
338. Wu, G., Shen, X., Li, H., Chen, H., Lin, A., Suganthan, P.N.: Ensemble of differential evolution variants. Inf. Sci. **423**, 172–186 (2018)
339. Wu, T., Kareem, A.: Modeling hysteretic nonlinear behavior of bridge aerodynamics via cellular automata nested neural network. J. Wind Eng. Ind. Aerodyn. **99**(4), 378–388 (2011)
340. Xiaomin, X., Qing, S., Ling, Z., Bin, Z.: Parameter estimation and its sensitivity analysis of the MR damper hysteresis model using a modified genetic algorithm. J. Intell. Mater. Syst. Struct. **20**(17), 2089–2100 (2009)
341. Xie, Y., Fu, J.L., Chen, B.Y.: Parameter identification of hysteresis nonlinear dynamic model for piezoelectric positioning system based on the improved particle swarm optimization method. Adv. Mech. Eng. **9**(6), 1687814017702813 (2017)
342. Xu, G., Yu, G.: On convergence analysis of particle swarm optimization algorithm. J. Comput. Appl. Math. **333**, 65–73 (2018)
343. Yam, J.Y., Chow, T.W.: Feedforward networks training speed enhancement by optimal initialization of the synaptic coefficients. IEEE Trans. Neural Netw. **12**(2), 430–434 (2001)
344. Yang, K., Maginu, K., Nomura, H.: Parameters identification of chaotic systems by quantum-behaved particle swarm optimization. Int. J. Comput. Math. **86**(12), 2225–2235 (2009)
345. Yang, M.J., Gu, G.Y., Zhu, L.M.: Parameter identification of the generalized Prandtl–Ishlinskii model for piezoelectric actuators using modified particle swarm optimization. Sens. Actuat. A **189**, 254–265 (2013)
346. Yang, X.S., He, X.S.: Mathematical Foundations of Nature-Inspired Algorithms. Springer, New York (2019)
347. Yang, Y., Yang, B., Niu, M.: Adaptive trajectory tracking of magnetostrictive actuator based on preliminary hysteresis compensation and further adaptive filter controller. Nonlinear Dyn. **92**(3), 1109–1118 (2018)
348. Ye, M., Wang, X.: Parameter estimation of the Bouc–Wen hysteresis model using particle swarm optimization. Smart Mater. Struct. **16**(6), 2341 (2007)
349. Yildiz, Y.E., Topal, A.O.: Large scale continuous global optimization based on micro differential evolution with local directional search. Inf. Sci. **477**, 533–544 (2019)
350. Yousri, D., AbdelAty, A.M., Said, L.A., Elwakil, A., Maundy, B., Radwan, A.G.: Parameter identification of fractional-order chaotic systems using different meta-heuristic optimization algorithms. Nonlinear Dyn. **95**(3), 2491–2542 (2019)
351. Yu, Y., Li, Y., Li, J.: Nonparametric modeling of magnetorheological elastomer base isolator based on artificial neural network optimized by ant colony algorithm. J. Intell. Mater. Syst. Struct. **26**(14), 1789–1798 (2015a)
352. Yu, Y., Li, Y., Li, J.: Parameter identification of a novel strain stiffening model for magnetorheological elastomer base isolator utilizing enhanced particle swarm optimization. J. Intell. Mater. Syst. Struct. **26**(18), 2446–2462 (2015b)

353. Yuan, L., Yang, Q., Zeng, C.: Chaos detection and parameter identification in fractional-order chaotic systems with delay. Nonlinear Dyn. **73**(1–2), 439–448 (2013)
354. Yuan, L.G., Yang, Q.G.: Parameter identification and synchronization of fractional-order chaotic systems. Commun. Nonlinear Sci. Numer. Simul. **17**(1), 305–316 (2012)
355. Yuan, X., Zhang, T., Dai, X., Wu, L.: Master-slave model-based parallel chaos optimization algorithm for parameter identification problems. Nonlinear Dyn. **83**(3), 1727–1741 (2016)
356. Yun, H.B., Tasbighoo, F., Masri, S.F., Caffrey, J.P., Wolfe, R.W., Makris, N., Black, C.: Comparison of modeling approaches for full-scale nonlinear viscous dampers. J. Vib. Control **14**(1–2), 51–76 (2008)
357. Zakaria, M.Z., Jamaluddin, H., Ahmad, R., Loghmanian, S.M.: Comparison between multi-objective and single-objective optimization for the modeling of dynamic systems. Proc. Inst. Mech. Eng. Part I J. Syst. Control Eng. **226**(7), 994–1005 (2012)
358. Zaman, M.A., Sikder, U.: Bouc–Wen hysteresis model identification using modified firefly algorithm. J. Magn. Magn. Mater. **395**, 229–233 (2015)
359. Zhang, D., Fletcher, J.E.: Double-frequency method using differential evolution for identifying parameters in the dynamic Jiles–Atherton model of Mn–Zn ferrites. IEEE Trans. Instrum. Meas. **62**(2), 460–466 (2013)
360. Zhang, H., Li, B., Zhang, J., Qin, Y., Feng, X., Liu, B.: Parameter estimation of nonlinear chaotic system by improved TLBO strategy. Soft. Comput. **20**(12), 4965–4980 (2016)
361. Zhang, J., Sanderson, A.C.: JADE: adaptive differential evolution with optional external archive. IEEE Trans. Evol. Comput. **13**(5), 945–958 (2009)
362. Zhang, L., Yang, Y., et al.: Lag synchronization for fractional-order memristive neural networks via period intermittent control. Nonlinear Dyn. **89**(1), 367–381 (2017)
363. Zhang, Y., Yan, P.: Modeling, identification and compensation of hysteresis nonlinearity for a piezoelectric nanomanipulator. J. Intell. Mater. Syst. Struct. **28**(7), 907–922 (2017)
364. Zhang, Y., Wang, S., Ji, G.: A comprehensive survey on particle swarm optimization algorithm and its applications. Math. Probl. Eng. **2015**, 931256 (2015)
365. Zheng, J., Cao, S., Wang, H., Huang, W.: Hybrid genetic algorithms for parameter identification of a hysteresis model of magnetostrictive actuators. Neurocomputing **70**(4–6), 749–761 (2007)
366. Zheng, J., Cao, S., Wang, H.: Modeling of magnetomechanical effect behaviors in a giant magnetostrictive device under compressive stress. Sens. Actuat. A **143**(2), 204–214 (2008)
367. Zheng, Y.l., Ma, L.H., Zhang, L.Y., Qian, J.X.: Empirical study of particle swarm optimizer with an increasing inertia weight. In: The 2003 Congress on Evolutionary Computation, CEC'03, vol. 1, pp. 221–226. IEEE (2003)
368. Zhong, J., Hu, X., Zhang, J., Gu, M.: Comparison of performance between different selection strategies on simple genetic algorithms. In: International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06), vol. 2, pp. 1115–1121. IEEE (2005)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.