Genetic Programming and Evolvable Machines (2023) 24:1
https://doi.org/10.1007/s10710-023-09450-6

A new hybrid method of Evolutionary-Numerical algorithms to solve ODEs arising in physics and engineering

S. R. Mirshafaei[1] · H. Saberi Najafi[1] · E. khaleghi[2] · A. H. Refahi Sheikhani[1]

Received: 8 April 2022 / Revised: 7 December 2022 / Accepted: 11 January 2023 /
Published online: 14 February 2023

Abstract

The present study aimed to use artificial intelligence to obtain a mathematical model to approximate the exact solution for linear and nonlinear ordinary differential equations with initial conditions arising in physics and engineering. To this end, genetic programming has been implemented, along with its combination with the Runge–Kutta fourth order method (RK4). Regarding formulation, the produced mathematical models by this new hybrid method (GPN) are flexible (in terms of functions used in the model structure and the number of them) and have acceptable accuracy compared to other existing traditional powerful methods now in use. Numerical experiments have been adequately conducted to indicate the sufficient accuracy and productive power of GPN to generate human-competitive results.

Keywords Artificial intelligence · Evolutionary algorithms · Genetic programming · Linear and nonlinear ODEs

Mathematics Subject Classification 34–04 · 68T20

✉ E. khaleghi
khaleghi@guilan.ac.ir

S. R. Mirshafaei
reza.mirshafaei@gmail.com

H. Saberi Najafi
hnajafi@guilan.ac.ir

A. H. Refahi Sheikhani
ah_refahi@liau.ac.ir

[1] Department of Applied Mathematics, Lahijan Branch, Islamic Azad University, Lahijan, Iran

[2] Department of Dynamic, Control, and Vibration, Faculty of Mechanical Engineering, University of Guilan, Rasht, Iran

1 Introduction

Ordinary differential equations (ODEs) arise in many problems related to physics and engineering (particularly complex problems arising from vibrations and dynamic systems). Generally, numerical, analytical, and approximate methods are used to solve complex linear and nonlinear ODEs [15]. Exact solutions to these equations play an essential role in adequately understanding qualitative features of many phenomena and processes. However, even if a solution exists, only for a few special ODEs, it is possible to determine this exact solution in closed form by analytical methods. It means that there is not a general analytical approach to evaluate analytical solutions [20]. Therefore, numerical and approximate solutions are used. In most numerical methods, the rule of solution function is not specified, but a numerical sequence (data pairs) of approximate values is produced. Methods such as Euler, RK4, Adams-Bashforth, Etc. are in this category [4]. One of the strongest and most practical approximation methods for solving nonlinear ODEs is perturbation-based methods [25]. These methods are used to solve ODEs arising from vibrations and dynamics. Since these ODEs are abundantly observed in various branches of science and have also recently been focused on solving them, they can be used to evaluate the effectiveness of new methods. Methods such as Homotopy Perturbation Method (HPM) [13, 16, 19], Optimal Homotopy asymptotic method (OHAM) [35, 38], Variational Iteration Method (VIM) [17], Energy Balance Method (EBM) [10, 18, 26], Differential transform method (DTM) [41], Najjar-Ismail perturbation technique (NIPT) [12], are in perturbation-based category. Each method is used for a specific type of ODEs; the mentioned studies have shown that they offer an acceptable approximation of the ODE's solution. On the other hand, with the impressive advent of the virtual world and the expansion of powerful processors, mathematicians are trying to use these hardware technologies and artificial intelligence (especially evolutionary algorithms) as automated computing systems to manage complex and time-consuming problems. A process of mathematical modeling with evolutionary algorithms involves the model generation, interpretation of numerical results, and development and control of numerical algorithms by computers. These activities are not entirely independent, and researchers must provide some preliminaries as inputs. Also, they are not purely user-centric. Genetic Programming (GP) is widely regarded as one of the most useful evolutionary algorithms. It is a branch of artificial intelligence research that involves the evolution of computer codes, with the term "evolution" referring to Darwin's concept of natural evolution [2, 7]. Despite the fact that there is a vast amount of literature on both GP and numerical methods for solving ODEs, the majority of research uses both techniques separately [43]. Additionally, studies have demonstrated that GP can solve some ODEs when combined with automatic differentiation and least mean square [21, 36]. However, no combination of a numerical method with GP has been reported to solve ODEs. Numerical methods have a strong mathematical foundation, whereas GP has characteristics like educability, processing power, and relative independence from the user. Thus, the fundamental objective of this

study is to introduce a new hybrid method (GPN) and determine the potential of evolutionary algorithms constructed by combining GP and a numerical method for getting a good approximation of analytical solutions to ODEs such that GPN can compete with existing powerful methods in terms of flexibility and accuracy. GPN does not depend on the numerical method used for the combination. It requires a numerical method that produces very accurate input data. The RK4 technique is a more often used numerical method for solving ODEs than other similar methods [5]. RK4 is quite flexible. Recent studies have demonstrated that it is capable of generating high-precision data pairs from the solutions of complicated differential equations, including chaotic equations, delayed equations, fractional order equations, stochastic equations, and nonlinear equations [3, 6, 27, 49, 52]. Therefore, we use the RK4 method for combination operations and develop algorithms to solve various problems caused by vibrations and dynamic systems problems using the Matlab programming environment. Among these efforts, we emphasize [10, 12, 18, 26, 37, 45] which provided a benchmark for comparing our algorithms' performance. In general, our findings are positive because we could obtain a good approximation of exact solutions to the problems, indicating that the proposed method could be a viable alternative for solving ODEs. In the following sections, we establish the methodology, and structure of GPN, error function and accuracy of GPN, test problems (the method's good performance through various practical examples), discussion and conclusion respectively.

2 Methodology and structure of GPN

This section discusses the methodology used to construct the GPN algorithm to solve ODEs. GPN has the following overview structure: A set of models (individuals) is produced using a random combination of mathematical equations and relations, consisting of operators, mathematical functions, random numbers, and design variables as an initial population. Each individual in the population is assessed using a fit criterion established by the type of problem and given a fitness index depending on how well it fits into the desired aim or objectives. A new population of models is generated using GP operators (selection, crossover, mutation, reproduction and alteration operators) [24, 51]. The next generation is created based on more appropriate with higher quality models. Individuals with better quality have an increased possibility of being picked as the parents to produce offspring. This process is continued until a specified number of generations is reached or the objective is achieved. In the last generation, a genetic algorithm-based procedure [9] among the created models delivers an optimal solution in terms of the number of nodes and the fit criterion which enters the error analysis step. The details of GPN are discussed below. GPN is designed in the Matlab programming environment and uses two parallel algorithms: The first is in charge of GP, whereas the latter is in charge of producing RK4 data pairs (x_i, y_i). RK4 has three primary roles. First, a duplicate of all the data is used to examine the approximate behavior of the solution function, leading to an initial guess for determining the set of GP functions and terminals. Second, 90% of data pairs are selected randomly and used to train models across all

generations. Third, 10% of the data pairs are utilized to validate the final model. The testing data pairs determine the precision of the final model's match with RK4 data pairs that did not participate in the generations training process. GP is recognized by its tree structure, where each tree is a combination of some leafs (nodes) and branches, and these nodes involve some functions and terminals [30, 33]. For example, if $F = \{\times, \div, +, -, sin, cos\}$ and $T = \{x_1, x_2, 1\}$ one simple arithmetic expression can be generated as $cos(x_1 \times x_2) + (x_1 - x_2)$. Equivalently, Fig. 1 displays this type of tree structure. Generally, based on the problem quiddity, these functions (F) and terminals (T) should be selected. By considering the special problem, a function set may be a set of arithmetic operations, boolean operations, or mathematical functions. The terminal set could be constant values or design variables. Additionally, if the problem is an ODE, then RK4 is an appropriate algorithm for determining the function and terminal set. The fundamental principle of GPN is that an initial population (trees) is generated randomly from a set of initial solutions to an ODE problem. Due to the vast search space, the effectiveness of GPN is heavily dependent on the diversity and structure of the initial population. The ODE's solution function's linear, nonlinear, oscillating, and exponential states can be determined by conceptually examining the ODE structure [36] and analyzing the behavior of RK4 data pairs (in graph state). This is a good way to guess the initial population's base set (functions, mathematical operators, number of variables, and terminals) and determine the initial GP configurations. This process starts with a set of base elements and adds or changes them, and it is repeated until the desired result is achieved.

After determining the training and testing data, it is necessary to create the initial population. In the implementation of GP, the user can control the program execution process by changing various parameters. John Koza et al. have classified population size and the number of generations as main parameters and others as secondary parameters [32]. As a general rule, the population size should be as large as possible in order for the computer system to do computations effectively and in a reasonable amount of time [32]. This parameter's minimum value is 1000 [51]. In addition, the optional range for the number of generations is between 10 and 150. Also, the maximum depth of trees is usually considered to be 8 [32, 51]. Koza proposed three methods for creating the initial population. The full method, the grow method, and the ramped half-and-half (RHH) combines the aforementioned methods. RHH results in a more uniform and diverse range of tree sizes and shapes. RHH generates half of the

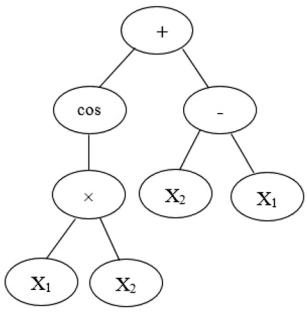


Fig. 1 Tree of the equation $cos(x_1 \times x_2) + (x_1 - x_2)$

starting population using the full procedure and the other half via the growth method [30, 33]. RHH is usually used to improve the diversity of the initial population (structural diversity) [51]. Jackson proposes modifying the RHH algorithm in which structural or behavioral duplicates are removed from the population to promote population diversity [23]. Furthermore, due to the reduction of computing costs, this process is only suggested for use in the first generation [22, 42]. Therefore, by using the modified RHH method only in the first generation (MRHH), it has been tried to achieve maximum population diversity by removing duplicate individuals. On the other hand, it has been demonstrated that recombination operators (using sub-tree crossover) have no significant effect on regression problems during the remainder of the evolutionary process (from the second generation onward) [23]. For more effectiveness, a real-value alternation operator is used in this study, which adds new structures to the population by slightly perturbing the terminals, leading to an increase in semantic diversity [24]. The performance of this operator will be described in detail in the rest of this section. Following the initial population's creation, it is the RK4 reliable input–output data's time to play a critical part in training them. A regression multi-fit program (RMF) is a user-friendly, adaptable nonlinear regression application that can fit data to various functions [48]. Using the RK4 training data pairs, the RMF assigns each member of the initial population a numeric value called the fitness criteria. Those who are more data-compatible or trainable are tested and coded. After establishing the initial population and analyzing the equivalent fitness values, the initial population is subjected to evolutionary processes (by GP operators) such as selection, reproduction, crossover (sexual recombination), mutation, and alteration. This process continues for a specific number of generations. As with other evolutionary algorithms, operators are applied to units in GP that are selected based on probability and fitness. The tournament selection is the most often used selection method [40]. The tournament determines which unit is superior to the other without specifying how much. Hence, the chances of being elected remain constant for the entire population automatically and effectively. One of the tournament's significant advantages is that units that behave inappropriately in initial generations are not immediately removed since they may represent proper attributes in future generations where this pattern is derived from nature [29, 30]. To choose individuals for genetic operations, tournament selection is employed with a max size of seven [31]. After the selection stage, other genetic operations are applied to the selected individuals. The crossover operation used in GP introduces variety into the population by generating new offspring composed of components from both parents. The crossover process begins with two parental expressions and terminates with the generation of two offspring expressions. The operation starts by selecting a random point within each parent as the parent's crossover point. After that, the crossover operator swaps the copies of two sub-trees rooted at these points, resulting in the formation of two new individuals, and they will be put into the new population. Copies are used to preserve the original trees. Repeating this process can lead to multiple offspring [24, 30]. The mutation operation introduces random changes in structures in the population. Mutation can be beneficial in reintroducing diversity in a population that may be tending to converge prematurely. Mutation begins by selecting a point at random within the tree. This mutation point can be an internal (i.e., function) point or an external (i.e., terminal) point of the tree. The mutation operation then removes whatever is currently at the

selected position and below the selected point. It uses the grow method to insert a randomly-generated sub-tree at that point. This operation is controlled by a parameter that determines the newly generated sub-tree's maximum size (measured by depth). This parameter also controls the crossover operator. The reproduction operator creates offspring by directly copying the selected individuals into the next generation. The selection procedure will be used to select an individual, who will then be directly copied into the new population. This operator ensures that the best individuals are not lost when new populations are generated [31]. Moreover, we apply a real-value alteration operator to promote local ability by randomly (using normal distribution) changing the values of real numbers (terminals) in tree structures. This operator could be considered a special case of the mutation operation, as it inserts a single terminal at a randomly selected terminal point of the tree. Similarly, this could be a special case of the crossover operation as exchanging two randomly selected sub-trees, which here are two selected terminals. This operator changes the selected terminals according to the Eq.(1) where *numpop* is the number of the initial population, $T_{s,i}$ and $T_{e,i}$ are selected and exchange terminals ($1 \leq i \leq numpop$), $\kappa \in$ span[0, 1], and UB_r, LB_r are upper and lower bounds of the interval in which the new terminal value is going to be produced, and rand[0,1] is a random number selected from normal distribution in the range 0–1 [24].

$$\begin{aligned} T_{e,i} &= (1-\kappa) \times \vartheta + \kappa \times T_{s,i}, \\ \vartheta &= rand[0,1] \times (UB_r - LB_r) + LB_r \end{aligned} \tag{1}$$

The objective of these GP operators is to generate a new population by passing on valuable materials from the parent population. Crossover, mutation, reproduction, and alternation operators are used at rates of 85%, 10%, 4%, and 1%, respectively [51]. When a termination criterion is met (last generation), the NSGA-II optimization algorithm (by using a non-dominated sorting procedure) [9] determines the optimal model in terms of the expressional complexity and the fit criterion, where the sum of the complexities of the tree structure and all its subtrees is the complexity metric, and the complexity is defined as the number of nodes (branch points plus leaves) [44]. This form of metric has the advantage of placing simpler solutions on the Pareto front when two different genotypes lead to the expression of the same phenotype. The obtained model enters the evaluation and error analysis step. The proposed GPN's schematic design is depicted in Fig. 2.

Now assume that the ODE with the used initial conditions is as follows:

$$\begin{aligned} u^{(n)} &= f(t, u, u', u'', ..., u^{(n-1)}), \quad t \in [a, b] \\ u(a) &= \alpha_0,\ u'(a) = \alpha_1, ...,\ u^{(n-1)}(a) = \alpha_{n-1} \end{aligned} \tag{2}$$

A mathematical model $u_{GPN}(t)$ is determined for Eq.(2) as an appropriate approximation of the solution function $y = u(t)$ by using GPN, where $u(t) \approx u_{GPN}(t)$. Regarding the GPN mechanism, $u_{GPN}(t)$ is the output of the "apply NSGA-II & set optimal model" block; therefore, it undergoes testing and error analysis. $u_{GPN}(t)$ is tested by RK4 testing data pairs and replaced in the ODE. The $\lambda(t)$ function and other criteria outlined in the next section determine the model's accuracy. The mathematical model that can provide the best precision for ODE is considered an appropriate

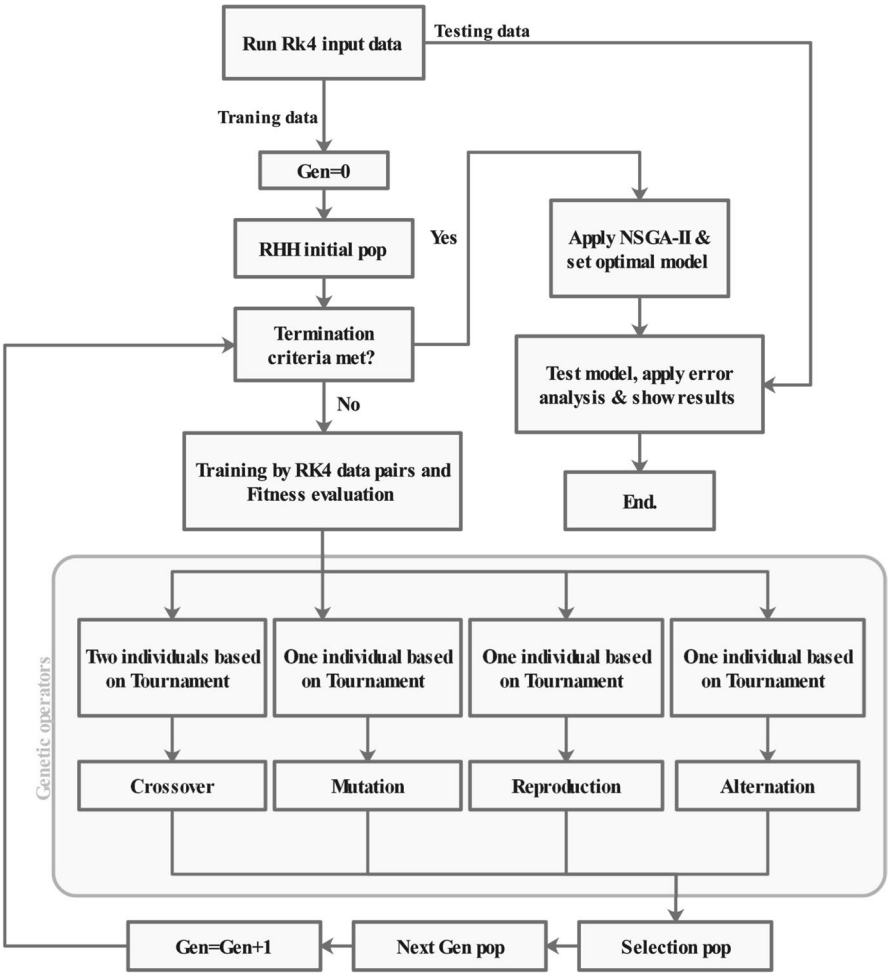


Fig. 2 Diagram of the proposed method

approximation of the solution function $u(t)$. Since GPN is a stochastic method, its average behavior must be considered (experiments should be run at least ten times independently). The number of the initial population and the depth of the trees (d_t) are two key parameters in controlling the computational times and the simplicity of the output models. we consider $numpop = 5000$ and $max(d\_t) = 6$. In Section 4, GPN will be tested on different ODEs.

3 Error function and accuracy

Two criteria ϕ, and ξ represent the model's accuracy in initial conditions, and the model's total accuracy in Eq. (2), respectively. The continuous function of $\lambda_{GPN}(t)$ indicates the error function. $\lambda_{GPN}(t)$ is obtained by inserting the model and its derivatives into Eq. (2) at all continuous points along the interval in which the ODE is defined. Further, The accuracy of predicting or fitting the obtained model ($u_{GPN}(t)$) with RK4 testing data pairs is assessed using (T_{error}) and Pearson's correlation coefficient (*PCC*) [47]. These criteria are defined to assess the accuracy of the model obtained by GPN as an appropriate approximation of the solution function of Eq. (2).

$$\lambda_{GPN}(t) = u^{(n)}_{GPN} - f(t, u'_{GPN}, u''_{GPN}, ..., u^{(n-1)}_{GPN}) \tag{3}$$

$$\phi = \frac{\sum_{k=0}^{n-1} \left| u^{(k)}_{GPN}(a) - \alpha_k \right|}{n-1} \tag{4}$$

$$\xi = \int_a^b \left| \lambda_{GPN}(t) \right| dt \tag{5}$$

$$T_{error} = \frac{1}{N} \sum_{i=1}^{N} \left(y_i - \hat{y}_i \right)^2 \tag{6}$$

Where, N, y_i and $\hat{y}_i$ are the number of RK4 testing data pairs, the value of RK4 testing data in time t_i and the value of $u_{GPN}(t)$ in time t_i respectively. If $PCC \to 1$, $\lambda_{GPN}(t) \cong zerofunction$, and other criteria are small enough, then $u_{GPN}(t)$ can be considered a reasonable approximation to the exact solution of Eq. (2).

4 Test problems

In this section, the GPN is implemented for five selected Problems of ODEs arising in physics and engineering. According to [10, 12, 18, 26, 37, 45], to achieve the desired result, problems 2–5 can only be solved in a specific method; in other words, a unique approximation method can not solve all the expressed problems. Problem 1 contains a linear ODE whose exact solution can be calculated by an analytical method. Problems 2–5 involve complex nonlinear ODEs with approximate solutions derived from the aforementioned semi-analytical methods.

Since these methods rely heavily on pure mathematics, it typically takes months (or years) to discover a formula that can solve a particular set of ODEs. Therefore, these methods cannot be compared to algorithmic methods such as GPN in execution time. However, the flexibility (in terms of functions used in the model structure and the number of them) and, more importantly, the accuracy of the obtained models are significant and comparable. GPN performance is acceptable when it achieves an approximation similar to or better than the results obtained by existing powerful methods.

4.1 Problem 1

The following equation is applied to analyze the vibration of a single-degree-of-freedom system [45, 50]:

$$mu'' + bu' + ku = 0 \quad , \; u(0) = A \; , \; u'(0) = B \tag{7}$$

General solution of the problem 1: For $m = 4$, $b = 16$, $k = 16$ and $A = 1, B = 0$ the general solution of Eq. (7) is obtained by Eq.(8) [45].

$$u_{_{GS}}(t) = e^{-2t} + 2te^{-2t} \tag{8}$$

Solving the problem 1 by the GPN method: The configuration and parameters used in GPN are summarized in Table 1. Here, RK4 with time steps $h = 0.001$ in the distance of $[0\,, 10]$ is considered to train and test in GPN and the mathematical model derived from GPN and its accuracy are shown by Eq. (9) and in Fig. 3.

$$u_{_{GPN}}(t) = e^{-2t} + 2te^{-2t} + 4.28 \times 10^{-15} \simeq e^{-2t} + 2te^{-2t} \tag{9}$$

The accuracy criteria of GPN and the comparison between GPN and other methods are demonstrated in Tables 2, 3 respectively.

4.2 Problem 2

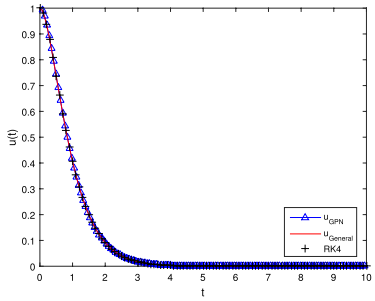
The motion of a particle on a rotating parabola is governed by the following equation:

$$\left(1 + 4q^2u^2\right)\frac{d^2u}{dt^2} + \Lambda u + 4q\left(\frac{du}{dt}\right)^2 u = 0 \tag{10}$$

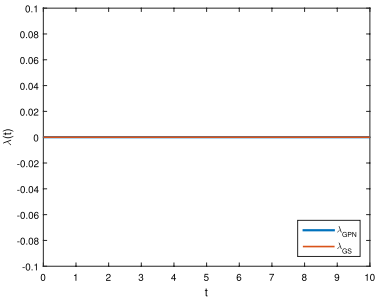
with the conditions $u(0) = A$, $u'(0) = 0$ where q and Λ are known constants and need not to be small [39]. For q=1, $\Lambda = 1$ and A=1 the results of the OHAM and GPN method are shown below.

Table 1 GPN configuration and Parameters for problems

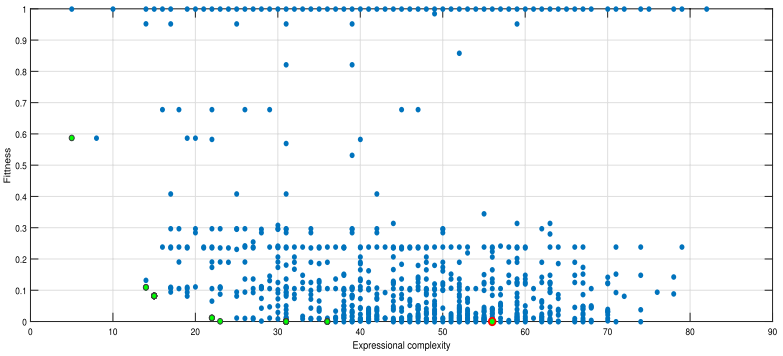
Problems	1	2	3	4	5
Population size	5000	5000	5000	5000	5000
Original population	MRHH	MRHH	MRHH	MRHH	MRHH
Number of generations	50	50	50	50	50
RK4 Output Values	10000	10000	10000	15000	23000
Type of fitness selection	Tournament 5	Tournament 5	Tournament 5	Tournament 5	Tournament 5
Max tree depth	6	6	6	6	6
Using function set	$\{\times, \div, +, -, sin, cos, exp\}$	$\{\times, \div, +, -, sin, cos\}$	$\{\times, \div, +, -, sin, cos\}$	$\{\times, \div, +, -, sin, cos\}$	$\{\times, \div, +, -, sin, cos\}$
Terminal set	$t, Rnd\ in[-10, 10]$	$t, Rnd\ in[-10, 10]$	$t, Rnd\ in[-10, 10]$	$t, Rnd\ in[-10, 10]$	$t, Rnd\ in[-10, 10]$
Stopping criteria	Max generation	Max generation	Max generation	Max generation	Max generation



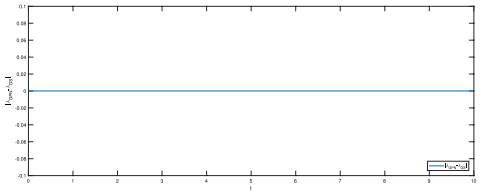
(a) Comparison between the solutions of GPN, General solution and RK4.



(b) Comparison between the $\lambda(t)$ of GPN (blue line) and General solution (red line).



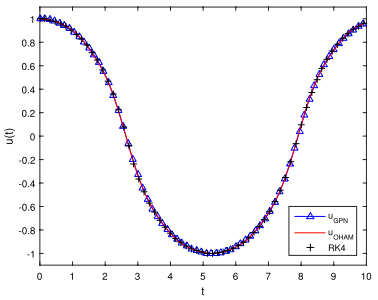
(c) Trade-off point



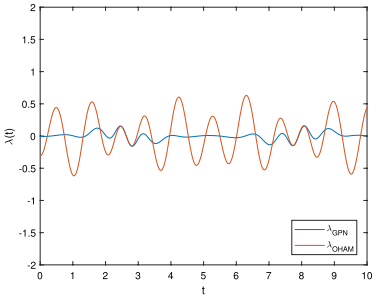
(d) $\left|\lambda_{GPN} - \lambda_{GS}\right|$.

Fig. 3 The results and accuracy of methods in Problem 1

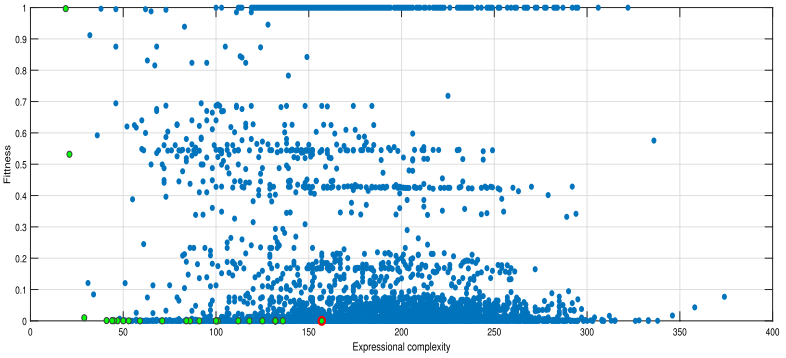
Solving the problem 2 by the OHAM: The approximate solution of Eq. (10) is obtained using the OHAM method [38] by Eq. (11).



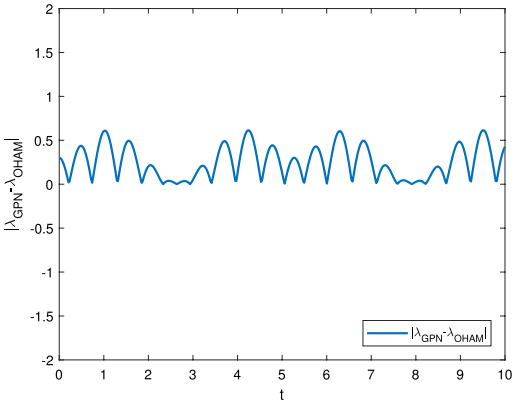
(a) Comparison between the solutions of GPN, OHAM and RK4.



(b) Comparison between the $\lambda(t)$ of GPN (blue line) and OHAM (red line).



(c) Trade-off point



(d) $\left|\lambda_{GPN} - \lambda_{OHAM}\right|$.

Fig. 4 The results and accuracy of methods in Problem 2

$$u_{_{OHAM}}(t) = 1.08140204\cos\Omega t - 0.100837908\cos 3\Omega t + 0.025163334\cos 5\Omega t - 0.008262879\cos 7\Omega t + 0.006681164\cos 9\Omega t - 0.004145751\cos 11\Omega t \tag{11}$$

where $\Omega = 0.596087918$.

Solving the problem 2 by the GPN method: The configuration and parameters used in GPN are summarized in Table 1. Here, RK4 with time steps $h = 0.001$ in the distance of $[0, 10]$ is considered to train and test in GPN. The mathematical model derived from GPN and its accuracy are shown by Eq. (12) and in Fig. 4.

$$u_{_{GPN}}(t) = 0.009481\sin(9.641\psi) - 0.003499\sin(7.856\sin(\cos(0.6039t))) - 17.65\sin(\psi) + 12.65\sin(\sin(\psi)) - 0.1475\sin(\sin(\sin(\cos(0.6068t)))) + 6.739\psi + 7.828\times 10^{-6}.\ \ where:\ \psi = \sin(\cos(0.5971t)) \tag{12}$$

4.3 Problem 3

A Duffing equation can describe various nonlinear physical phenomena such as a system consisting of the pendulum and nonlinear stiffness, Snap-through mechanism, nonlinear isolator, large deflection of a beam with nonlinear stiffness, nonlinear cable vibrations and nonlinear electrical circuit [28]. A Duffing oscillator with nonlinearity of fifth-order in the case of forced term is considered as in the following equation:

$$u'' + u + \varepsilon u^5 = p\cos\Omega t,\ \ u(0) = A,\ \ u'(0) = 0,\ \ 0 < \epsilon < \infty \tag{13}$$

where ϵ indicates the nonlinear arbitrary constants in the restoring force, which is not required to be very small in the present study. In addition, Ω is the angular frequency of the periodic driving force and p represents the amplitude of the external periodic force. hereafter, Eq. (13) is referred to briefly as the Duffing equation [12]. For $\epsilon = 100$, $P = 1$, $\Omega = 5$, $\beta = 1.00137$ and $A = 0.01$ the results of the NIPT and GPN method are shown below.

Solving the problem 3 by the NIPT: The approximate solution of Eq. (13) is obtained using NIPT by Eq. (14)

$$u_{_{NIPT}}(t) = u_0 + \alpha u_1 \tag{14}$$

Where α, u_0, u_1 and other parameters are determined by Naggar and Ismail by Eqs. (15)–(20) in [12].

$$\alpha = \frac{\varepsilon}{1+\varepsilon} \tag{15}$$

$$u_0 = m\cos\beta t + n\cos\Omega t \tag{16}$$

Where β is the angular frequency, which is unknown to be future determined by Eq. (20).

$$u_1 = \frac{\left(8\eta n + 5n^5 + 30m^2n^3 + 15m^4n\right)}{8\left(\beta^2 - \Omega^2\right)}(\cos\beta t - \cos\Omega t) - \frac{\left(5m^5 + 20m^3n^2\right)}{128\beta^2}$$
$$(\cos\beta t - \cos 3\beta t) + \frac{\left(5n^5 + 20m^2n^3\right)}{16\left(\beta^2 - 9\Omega^2\right)}(\cos\beta t - \cos 3\Omega t) - \frac{m^4}{384\beta^2}$$
$$(\cos\beta t - \cos 5\beta t) + \frac{n^5}{16\left(\beta^2 - 25\Omega^2\right)}(\cos\beta t - \cos 5\Omega t) +$$
$$\frac{5m^5n}{16\left[\beta^2 - \left(4\beta + \Omega^2\right)\right]}(\cos\beta t - \cos(4\beta + \Omega)t) + \frac{20m^4n}{16\left[\beta^2 - (2\beta + \Omega)^2\right]}$$
$$(\cos\beta t - \cos(2\beta + \Omega)t) + \frac{20m^4n}{16\left[\beta^2 - (2\beta + \Omega)^2\right]}(\cos\beta t - \cos(2\beta - \Omega)t) +$$
$$\frac{5m^4n}{16\left[\beta^2 - (4\beta - \Omega)^2\right]}(\cos\beta t - \cos(4\beta - \Omega)t) + \frac{5m^3n^2}{8\left[\beta^2 - (5m3\beta + 2\Omega)^2\right]}$$
$$(\cos\beta t - \cos(3\beta + 2\Omega)t) + \frac{15m^3n^2}{8\left[\beta^2 - (\beta + 2\Omega)^2\right]}(\cos\beta t - \cos(\beta + 2\Omega)t) +$$
$$\frac{15m^3n^2}{8\left[\beta^2 - (\beta - 2\Omega)^2\right]}(\cos\beta t - \cos(\beta - 2\Omega)t) + \frac{5m^3n^2}{8\left[\beta^2 - (3\beta - 2\Omega)^2\right]}$$
$$(\cos\beta t - \cos(3\beta - 2\Omega)t) + \frac{5m^2n^3}{8\left[\beta^2 - (2\beta + 3\Omega)^2\right]}(\cos\beta t - \cos(2\beta + 3\Omega)t) +$$
$$\frac{15m^2n^3}{8\left[\beta^2 - (2\beta + \Omega)^2\right]}(\cos\beta t - \cos(2\beta + \Omega)t) + \frac{15m^2n^3}{8\left[\beta^2 - (2\beta - \Omega)^2\right]}$$
$$(\cos\beta t - \cos(2\beta - \Omega)t) + \frac{5m^2n^3}{8\left[\beta^2 - (2\beta - \Omega)^2\right]}(\cos\beta t - \cos(2\beta - 3\Omega)t) +$$
$$\frac{5mn^4}{16\left[\beta^2 - (\beta + 4\Omega)^2\right]}(\cos\beta t - \cos(\beta + 4\Omega)t) + \frac{20mn^4}{16\left[\beta^2 - (\beta + 2\Omega)^2\right]}$$
$$(\cos\beta t - \cos(\beta + 2\Omega)t) + \frac{20mn^4}{16\left[\beta^2 - (\beta - 2\Omega)^2\right]}(\cos\beta t - \cos(\beta - 2\Omega)t) +$$
$$\frac{5mn^4}{16\left[\beta^2 - (\beta - 4\Omega)^2\right]}(\cos\beta t - \cos(\beta - 4\Omega)t). \tag{17}$$

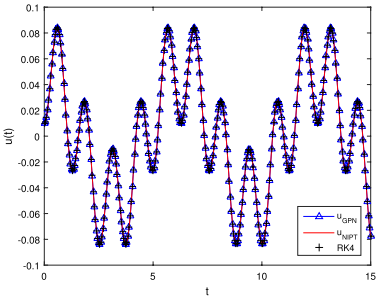
$$m = \frac{p}{\beta^2 - \Omega^2},\ n = A - \frac{p}{\beta^2 - \Omega^2},\ \alpha = \frac{\varepsilon}{1 + \varepsilon} \tag{18}$$

$$\eta = -\left(\frac{5}{8}m^4 + \frac{30}{8}m^2n^2 + \frac{15}{8}n^4\right) \tag{19}$$

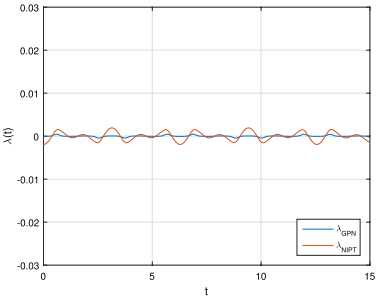
$$\beta^2 + \varepsilon\eta = 1,\ \varepsilon\eta < 1 \tag{20}$$

Solving the problem 3 by the GPN method: The configuration and parameters used in GPN are summarized in Table 1. Here, RK4 with time steps $h = 0.0015$ in the distance of $[0\,,15]$ is considered to train and test in GPN and the mathematical model derived from GPN and its accuracy are shown by Eq. (21) and in Fig. 5.

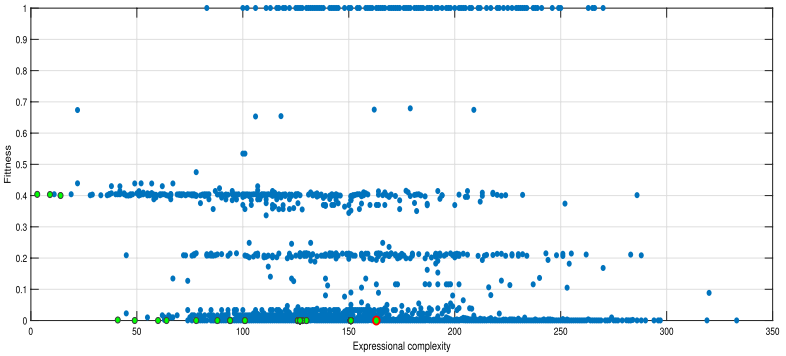
$$u_{GPN}(t) = 0.156\sin(\cos(t)) - 0.5212\sin(\sin(\cos(t))) + 0.157\sin^2(2t)\cos(t) + 0.299\cos(\sin(\cos(\cos(t))))\cos(t) - 6.63\times10^{-5}t\sin(t) - 5.79\times10^{-8} t\cos(t)(t+\sin(2t)) + 5.59\times10^{-9} \tag{21}$$



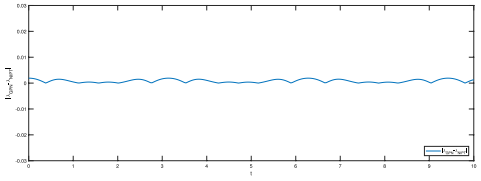
(a) Comparison between the solutions of GPN, NIPT and RK4.



(b) Comparison between the $\lambda(t)$ of GPN (blue line) and NIPT (red line).



(c) Trade-off point



(d) $|\lambda_{GPN} - \lambda_{NIPT}|$.

Fig. 5 The results and accuracy of methods in Problem 3

4.4 Problem 4

The Van der Pol oscillator is an example of self-oscillatory systems. Recently, it has been proposed as fundamental tools for control and reduction of friction [37, 46]. The Van der Pol oscillator for every nonnegative value of the parameter ϵ is described by the following equation:

$$u'' + \varepsilon\left(u^2 - 1\right)u' + u = 0 \quad , \; u(0) = A \; , \quad u'(0) = 0 \tag{22}$$

For $\epsilon = 0.1$, and $A = 1$ the results of the VIM and GPN method are shown below.

Solving the problem 4 by the VIM: The approximate solution of Eq. (22) is obtained using VIM by Eq. (23) [26].

$$\begin{aligned} u_{_{VIM}}(t) = \cos(t) + \varepsilon\left(\frac{3}{8}t\cos(t) - \frac{1}{8}\sin t\cos^2 t - \frac{1}{4}\sin t\right) + \varepsilon^2\left(\frac{-11}{256}t\sin t\right) + \\ \varepsilon^2\left(\frac{3}{128}t^2\cos t - \frac{5}{192}\cos^5 t - \frac{83}{768}\cos^3 t - \frac{9}{64}t\sin t\cos^2 t + \frac{103}{768}\cos t\right) \end{aligned} \tag{23}$$

Solving the problem 4 by the GPN method: The configuration and parameters used in GPN are summarized in Table 1. Here, RK4 with time steps $h = 0.001$ in the distance of $[0\,, 15]$ is considered to train and test in GPN and the mathematical model derived from GPN and its accuracy are shown by Eq. (24) and in Fig. 6.

$$\begin{aligned} u_{_{GPN}}(t) = 7.64 \times 10^{-5}t - 0.3321\sin(\sin(t)) - 0.001457\sin(3t) + 0.9936\cos(t) - \\ \frac{8.365 \times 10^{15}t}{2.306 \times 10^{18}t + 1.582 \times 10^{19}} + 9.16 \times 10^{-4}\sin(\cos(t))(2t + 6.86) - \\ 8.8 \times 10^{-6}t^3\sin(\sin(t)) + 0.0369t\cos(t) - 6.98 \times 10^{-7}t^4\cos(t + 6.86) - \\ 0.00203t\cos^2(t)\sin(t) + 0.00121 \end{aligned} \tag{24}$$

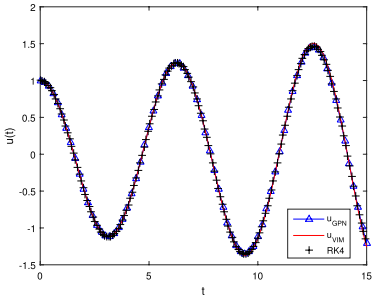
4.5 Problem 5

This problem is aimed to investigate the dynamic response of a rocking rigid rod over a circular surface, considering a pure rolling, without slipping. The governing equation on the mentioned system is given below [11].

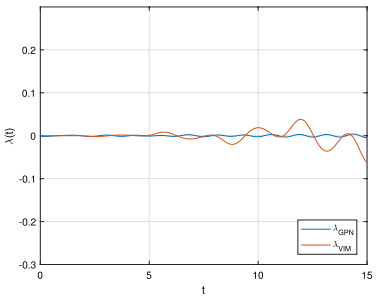
$$u'' + Mu''u^2 + Nu\left(u'\right)^2 + \alpha u - Wu^3 = 0 \tag{25}$$

with the initial conditions $u(0) = A$, $u'(0) = 0$ where $M = N = \frac{12r^2}{l^2}$, $W = \frac{6gr^2}{l^2}$ and $\alpha = \frac{12gr}{l^2}$ and r, l, g are known constants. For l=1, $\frac{r}{l} = \frac{\sqrt{6}}{6}$ and A=0.1 the results of the EBM and GPN method are shown below.

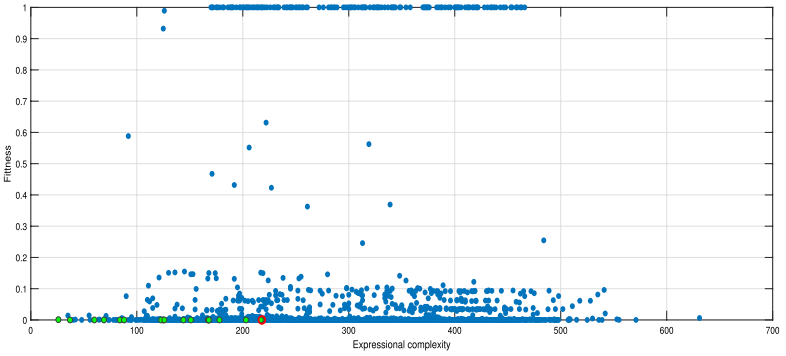
Solving the problem 5 by the EBM: The approximate solution of Eq. (25) is obtained using the EBM [10, 18] by Eq. (26).



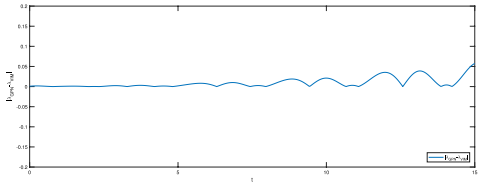
(a) Comparison between the solutions of GPN, VIM and RK4.



(b) Comparison between the $\lambda(t)$ of GPN (blue line) and VIM (red line).



(c) Trade-off point

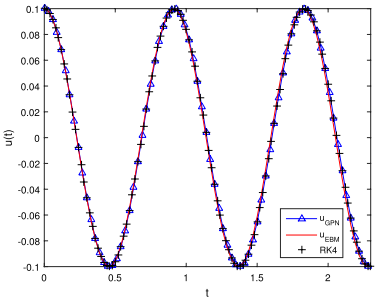


(d) $\left|\lambda_{GPN} - \lambda_{VIM}\right|$.

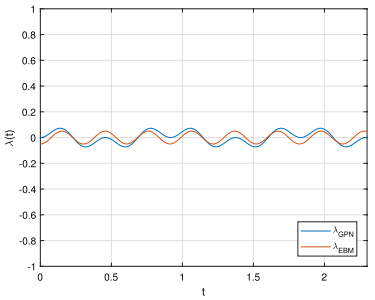
Fig. 6 The results and accuracy of methods in Problem 4

$$u_{EBM}(t) = A\cos\left(\frac{1}{2}\frac{\sqrt{-\left(1+A^2\right)\left(-4\alpha+3WA^2\right)}}{1+A^2}\ t\right) \tag{26}$$

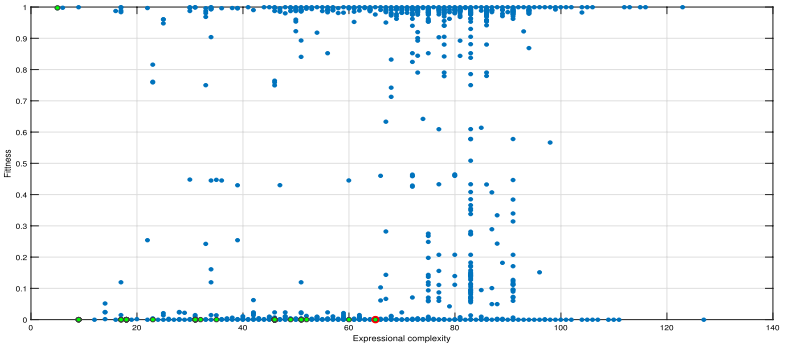
Solving the problem 5 by the GPN method: The configuration and parameters used in GPN are summarized in Table 1. Here, RK4 with time steps $h = 0.0001$ in the distance of $[0\,, 2.3]$ is considered to train and test in GPN. The mathematical model derived from GPN and its accuracy are shown by Eq. (27) and in Fig. 7.



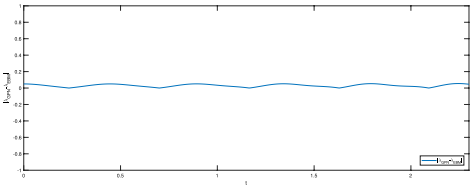
(a) Comparison between the solutions of GPN, EBM and RK4.



(b) Comparison between the $\lambda(t)$ of GPN (blue line) and EBM (red line).



(c) Trade-off point



(d) $\left|\lambda_{GPN} - \lambda_{EBM}\right|$.

Fig. 7 The results and accuracy of methods in Problem 5

$$u_{GPN}(t) = 0.01552\cos(6.854t) - 6.46\times10^{-6}\cos(20.57t) - 8.05\times10^{-7}\sin(\cos(7.033t)) + 0.08449\cos(6.856t) - 1.129\times10^{-10} \tag{27}$$

5 Discussion and conclusion

In this study, GP and its combination with the RK4 method have been employed to propose a new GPN method for solving linear and complicated nonlinear ODEs in physics and engineering. Since GPN is a stochastic method, the optimal

determination of the initial parameters (especially the set of functions and terminals) significantly impacts its performance and maintains the structural diversity of the initial population. Two factors are considered to determine the set of functions and terminals in the GPN method. First, the study of the physical structure of the ODE and, more importantly, second, the behavior of RK4 data pairs. The graph obtained from Rk4 data pairs has helped to make an initial guess about the type of functions and terminals used in the structure of the ODEs solution. The graph of the RK4 data pairs in problem 1 (Fig. 3-a) clearly shows that the structure of the approximate solution consists of an exponential function, and in problems 2–5 (Figs. 4-a, 5-a, 6-a, and 7-a), there is a periodic and trigonometric structure. Additionally, the terminal set is determined based on the RK4 graph's range. Other parameters, such as the number of generations, the size of the initial population, the depth of the trees, the method of selection (and its size), the rate of applying genetic operators, and the number of GPN executions, have been determined according to the criteria described in Sect. 3. After selecting the initial parameters, the MRHH method has been employed to create the initial population, improving the diversity. On the other hand, 90% of the Rk4 data pairs have been considered for the training population in all generations. Genetic operators admit models that have shown more compatibility with these data before the evolution process. In addition to the traditional GP operators, a real-valued alternation operator (described in Sect. 3) is added to the evolution process since the input problem of the GPN is an ODE, so the terminals (function coefficients) are sensitive. Sometimes, slight changes in the terminals can lead to an unexpected solution. That is, a model that does not have good compatibility with the training data in generation i may become a compatible solution in generation $i + 1$ just by changing some coefficients (terminals). After the completion of the evolutionary processes (the end of the final generation), the individuals (produced models) have been evaluated by the NSGA-II multi-objective optimization algorithm in terms of the expressional complexity (simplicity compared to other models) and fitness values. In multi-objective optimization problems, it is not possible to find points that simultaneously maximize (minimize) all objective functions; hence, non-dominated design points (Pareto points) have been presented as a candidate for the final solution (green points in Figs. 3-c-, 4-c, 5-c, 6-c, and 7-c). Each member of the set of Pareto points is superior to the other design points (at least from the perspective of an objective function). Among the Pareto points, the trade-off point of the objective functions (green point with a red border) has been selected, and its corresponding model has gone to the error analysis and testing step. The placement and arrangement of the all points in Figs. 3-c-, 4-c, 5-c, 6-c, and 7-c demonstrate the fitness (semantic) diversity. In other words, the strategy used in the GPN method has led to the production of models that tend to solve ODEs with an acceptable approximation. The method's validity has been shown for five ODEs (one linear and four complex nonlinear equations). As mentioned in section 4, the amount of conformity of the model obtained with 10% of the RK4 testing data pairs for each problem has been measured by the T_{error} criterion and Pearson's correlation coefficient. The first and second rows of Table 2 show that the final model of each problem is entirely compatible with these data,

Table 2 Accuracy criteria of GPN for each problem

Problems	1	2	3	4	5
T_{error}	$1.66 \times 10^{-27} \simeq 0$	2.41×10^{-7}	3.09×10^{-14}	9.27×10^{-9}	4.79×10^{-17}
PCC	1	1	1	1	1
ϕ	$4.36 \times 10^{-15} \simeq 0$	5.01×10^{-5}	2.31×10^{-8}	5.04×10^{-4}	1.08×10^{-9}
GPN runs	10	10	10	10	10
Average run time (per run)	276 sec	541 sec	363 sec	426 sec	237 sec

Table 3 Comparison of ξ criterion between GPN and other methods

Problems	1	2	3	4	5
ξ_{GPN}	$1.642 \times 10^{-13} \simeq 0$	0.355	2.86×10^{-4}	0.020	0.0914
$\xi_{Other\ methods}$	$\xi_{GS} = 0$	$\xi_{OHAM} = 2.740$	$\xi_{NIPT} = 0.0117$	$\xi_{VIM} = 0.134$	$\xi_{EBM} = 0.0939$

despite the fact that the RK4 testing data pairs were not included in the training of the models. Moreover, the third row of Table 2 (ϕ criterion) shows the model's behavior in the initial conditions of ODEs. For each of the five problems, two initial conditions are given. The calculated values have shown that the final models obtained by GPN are compatible with the initial conditions of the ODEs. Also, the ϕ criterion can be considered a special case of the total accuracy of the model in initial conditions point(s). The total accuracy of the model (ξ) is a reliable criterion in evaluating the performance of methods for solving ODEs because it is obtained by putting the solution produced by the methods and its derivatives into the equation. The values of this criterion for GPN and other comparative methods are listed in Table 3. According to the theory of ODEs, the exact solution of an ODE is unique if it exists (problem 1). If the exact solution is unavailable (especially in nonlinear equations), approximate solutions can be produced with different methods (problems 2–5). Powerful methods produce solutions with sufficient accuracy and a different structure for a problem. The difference between the model errors can describe this structural difference. There is no difference in Fig. 1-d because GPN has converged to the exact solution. In addition, in Figure 6-d, this difference is more prominent in the interval [6 , 15], while in Figs. 4-d, 5-d, and 7-d, this difference is evident throughout the distance. It is possible to conclude that GPN has achieved an accuracy comparable to that of the powerful methods despite producing different structures compared to the existing solutions. The results show that GPN accuracy is equivalent to or better than existing methods in all cases. GPN can be developed in future work to solve other ODEs, such as chaotic, delayed, fractional order, stochastic, and partial equations. Also, mathematical and physical systems based on input–output data can be modeled and described by developing the idea implemented in the GPN method.

Declarations

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

References

1. P. Balasubramaniam, A. Vincent Antony Kumar, Solution of matrix Riccati differential equation for nonlinear singular system using genetic programming. Genetic Programming and Evolvable Machines. **10**, 71–89(2009). https://doi.org/10.1007/s10710-008-9072-z
2. W. Banzhaf, Artificial intelligence: Genetic programming. International Encyclopedia of the Social and Behavioral Sciences (Second Edition). 41-45(Available online 12 March)(2015). https://doi.org/10.1016/B978-0-08-097086-8.43003-5
3. A.H. Bukhari, M.A.Z. Raja, N. Rafiq, M. Shoaib, A.K. Kiani, C.M. Shu, Design of intelligent computing networks for nonlinear chaotic fractional Rossler system. Chaos, Solitons & Fractals **157**, 111–985 (2022). https://doi.org/10.1016/j.chaos.2022.111985
4. S. Chakraverty, N. Mahato, P. Karunakar, TD. Rao, Advanced numerical and semi-analytical methods for differential equations. John Wiley & Sons(2019)
5. V. Chauhan, P.K. Srivastava, Computational techniques based on Runge-Kutta method of various order and type for solving differential equations. International Journal of Mathematical, Engineering and Management Sciences. **4**, 375–386 (2019). https://doi.org/10.33889/IJMEMS.2019.4.2-030
6. Two-step Runge-Kutta methods for stochastic differential equations, DAmbrosio, R., Scalone, C. Applied Mathematics and Computation. **403**, 125–930 (2021). https://doi.org/10.1016/j.amc.2020.125930
7. Ch. Darwin, On the Origin of Species...(John Murray, London). Mentor edition, New American Library, New York City(1859)
8. W.J. de Araujo Lobão, M.A.C. Pacheco, D.M. Dias, A.C.A. Abreu, Solving stochastic differential equations through genetic programming and automatic differentiation. Engineering Applications of Artificial Intelligence. **68**, 110–120(2018). https://doi.org/10.1016/j.engappai.2017.10.021
9. K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation. **6**, 182–197 (2002). https://doi.org/10.1109/4235.996017
10. S. Durmaz, S.A. Demirbağ, M. Kaya, High order He's energy balance method based on collocation method. International Journal of Nonlinear Sciences and Numerical Simulation. **11**, 1–6 (2010). https://doi.org/10.1515/IJNSNS.2010.11.S1.1
11. Y.O. El-Dib, G.M. Moatimid, Stability Configuration of a Rocking Rigid Rod over a Circular Surface Using the Homotopy Perturbation Method and Laplace Transform. Arabian Journal for Science and Engineering. **44**, 6581–6591 (2019). https://doi.org/10.1007/s13369-018-03705-6
12. A. El-Naggar, G. Ismail, Analytical solution of strongly nonlinear Duffing oscillators. Alexandria Engineering Journal. **55**, 1581–1585 (2016). https://doi.org/10.1016/j.aej.2015.07.017
13. DD. Ganji, MM. Alipour, AH. Fereydoun, Y. Rostamian, Analytic approach to investigation of fluctuation and frequency of the oscillators with odd and even nonlinearities. International Journal of Engineering. **23**, 41–56(2010). http://www.ije.ir/article_71830.html
14. DE. Goldberg, A note on Boltzmann tournament selection for genetic algorithms and population-oriented simulated annealing. Complex Systems. **4**, 445–460(1990). https://www.complex-systems.com/abstracts/v04_i04_a05/
15. M. Hatami, D.D. Ganji, M. Sheikholeslami, *Differential Transformation method for Mechanical Engineering Problems* (Academic Press-Reed Elsevier, Netherlands, 2016)
16. J.H. He, Homotopy perturbation technique. Computer methods in applied mechanics and engineering. **178**, 257–262 (1999). https://doi.org/10.1016/S0045-7825(99)00018-3
17. J.H. He, Variational iteration method-a kind of non-linear analytical technique: some examples. International journal of non-linear mechanics. **34**, 699–708 (1999). https://doi.org/10.1016/S0020-7462(98)00048-1
18. J.H. He, Preliminary report on the energy balance for nonlinear oscillations. Mechanics Research Communications. **29**, 107–111 (2002). https://doi.org/10.1016/S0093-6413(02)00237-9

19. J.H. He, Homotopy perturbation method: a new nonlinear analytical technique. Applied Mathematics and computation. **135**, 73–79 (2003). https://doi.org/10.1016/S0096-3003(01)00312-5
20. M. Hermann, M. Saravi, Nonlinear ordinary differential equations. Springer (2016). https://doi.org/10.1007/978-81-322-2812-7
21. H. Iba, Inference of differential equation models by genetic programming. Information Sciences. **178**, 4453–4468 (2008). https://doi.org/10.1016/j.engappai.2017.10.021
22. D. Jackson, Phenotypic diversity in initial genetic programming populations. European Conference on Genetic Programming. Springer, Berlin, Heidelberg., 98–109(2010). https://doi.org/10.1007/978-3-642-12148-7_9
23. D. Jackson, Promoting phenotypic diversity in genetic programming. International Conference on Parallel Problem Solving from Nature. Springer, Berlin, Heidelberg., 472–481(2010). https://doi.org/10.1007/978-3-642-15871-1_48
24. A. Jamali, E. Khaleghi, I. Gholaminezhad, N. Nariman-Zadeh, Modelling and prediction of complex non-linear processes by using Pareto multi-objective genetic programming. International Journal of Systems Science. **47**, 1675–1688 (2016). https://doi.org/10.1080/00207721.2014.945983
25. R.N. Jazar, Perturbation Methods in Science and Engineering. Springer (2021). https://doi.org/10.1007/978-3-030-73462-6
26. S.H.H. Kachapi, DD. Ganji, Dynamics and vibrations. Springer(2015)
27. C.A. Kennedy, M.H. Carpenter, Higher-order additive Runge-Kutta schemes for ordinary differential equations. Applied numerical mathematics. **136**, 183–205 (2019). https://doi.org/10.1016/j.apnum.2018.10.007
28. I. Kovacic, M.J. Brennan, The Duffing equation: nonlinear oscillators and their behaviour. John Wiley & Sons(2011)
29. J.R. Koza, Genetic programming: A paradigm for genetically breeding populations of computer programs to solve problems (Vol. 34). Stanford, CA: Stanford University, Department of Computer Science.(1990)
30. J.R. Koza, Genetic programming II: Automatic discovery of reusable subprograms. MIT Press(1994)
31. J.R. Koza, Genetic programming as a means for programming computers by natural selection. Statistics and Computing. **4**, 87–112 (1994). https://doi.org/10.1007/BF00175355
32. J.R. Koza, Genetic programming In Search methodologies. Springer, Boston, MA.(2005). https://doi.org/10.1007/0-387-28356-0_5
33. J.R. Koza, M.A. Keane, M.J. Streeter, W. Mydlowec, J. Yu , G. Lanza, Genetic programming IV: Routine human-competitive machine intelligence. Springer Science & Business Media(2006)
34. J.R. Koza, Human-competitive results produced by genetic programming. Genetic programming and evolvable machines. **11**(3), 248–251 (2010). https://doi.org/10.1007/s10710-010-9112-3
35. S. Liao, Homotopy analysis method in nonlinear differential equations. Beijing Higher education press(2012). https://doi.org/10.1007/978-3-642-25132-0
36. W.JA. Lobão, D.M. Dias, M.A. Pacheco, Genetic programming and automatic differentiation algorithms applied to the solution of ordinary and partial differential equations. IEEE Congress on Evolutionary Computation (CEC)., 5286–5292(2016). https://doi.org/10.1109/CEC.2016.7748362.
37. V. Marinca, N. Herişanu, I. Laza, E. Ghita, An Application of the Optimal Homotopy Asymptotic Method to Generalized Van der Pol Oscillator. Applied Mechanics and Materials. **801**, 33–37 (2015). https://doi.org/10.4028/www.scientific.net/AMM.801.33
38. V. Marinca, N. Herişanu, Determination of periodic solutions for the motion of a particle on a rotating parabola by means of the optimal homotopy asymptotic method. Journal of Sound Vibration. **329**, 1450–1459 (2010). https://doi.org/10.1016/j.jsv.2009.11.005
39. A. Nayfeh, D.T. Mook, Nonlinear oscillations. John Wiley & Sons(2008)
40. S. Nguyen, Y. Mei, M. Zhang, Genetic programming for production scheduling: a survey with a unified framework. Complex and Intelligent Systems. **3**, 41–66 (2017). https://doi.org/10.1007/s40747-017-0036-x
41. S. Nourazar, A. Mirzabeigy, Approximate solution for nonlinear Duffing oscillator with damping effect using the modified differential transform method. Scientia Iranica. **20**, 364–368 (2013). https://doi.org/10.1016/j.scient.2013.02.023
42. D.P. Searson, GPTIPS 2: an open-source software platform for symbolic data mining. Handbook of genetic programming applications. Springer, Cham. 551-573(2015). https://doi.org/10.1007/978-3-319-20883-1_22

43. T. Seaton, G. Brown, J.F. Miller, Analytic Solutions to Differential Equations under Graph-Based Genetic Programming. Genetic Programming: 13th European Conference, EuroGP 2010, Istanbul, Turkey, April 7-9, 2010. Proceedings.Springer, Berlin, Heidelberg. **21**, 232-243(2010). https://doi.org/10.1007/978-3-642-12148-7_20
44. G.F. Smits, M. Kotanchek, Pareto-front exploitation in symbolic regression. Genetic programming theory and practice II. Springer, Boston, MA. 283-299(2005). https://doi.org/10.1007/0-387-23254-0_17
45. G. Strang, Differential equations and linear algebra. Wellesley-Cambridge Press Wellesley(2014)
46. M. Tsatsos, *Theoretical and Numerical study of the Van der Pol equation* (Aristotle University of Thessaloniki, Doctoral desertation, 2006)
47. D. Wackerly, W. Mendenhall, R.L. Scheaffer, Mathematical statistics with applications. Cengage Learning(2014)
48. A.R. Walmsley, A.G. Lowe, Multifit: a flexible non-linear least squares regression program in BASIC. Computer methods and programs in biomedicine. **21**, 113–118 (1985). https://doi.org/10.1016/0169-2607(85)90070-7
49. W. Wang, Y. Li, K. Wu, Y. Cui, Y. Song, Nonlinear Dynamics Analysis of Electric Energy Regeneration Device Based on Vibration Energy Recovery. In Advances in Nonlinear Dynamics. Springer, Cham, 241–253(2022). https://doi.org/10.1007/978-3-030-81170-9_22
50. W.C. Xie, Differential Equations for Engineers. Cambridge University Press(2010)
51. F. Zhang, S. Nguyen, Y. Mei, M. Zhang, Genetic Programming for Production Scheduling. Springer Singapore(2021). https://doi.org/10.1007/978-981-16-4859-5
52. M. Zhu, B. Chang, C. Fu, Convolutional neural networks combined with runge-kutta methods. Computer methods and programs in biomedicine. arXiv preprint arXiv:1802.08831(2018). https://doi.org/10.48550/arXiv.1802.08831