[1]Guoxing Si

Research on the Data-Driven Differential Equation-Solving Algorithm Based on Artificial Intelligence

Abstract: - Data-driven DEs has gained popularity in the past few years. This work proposes the new framework, named Adam Gannet Optimization Algorithm (AdamGOA), that combines Adam Optimization and Gannet Optimization Algorithm (GOA) to improve a stability, solve higher order Differential Equations (DE) and accuracy of DE. Adam is a first-order gradient-based methods, optimizes stochastic objectives using adaptive lower-order moments. In contrast, GOA represents a different distinct action of a gannets mathematically during foraging and is employed to facilitate exploitation and exploration. In addition, a Shepard Convolutional Neural Network (ShCNN) processed data to construct meta-data and estimate derivatives. After that, the unified integral form is established to determine optimal structure. Heterogeneous parameters are used to estimate and are labeled as constants or variables. Furthermore, the experimental findings showed that the AdamGOA_ ShCNN beat leading models in Accuracy, Convergence, and Mean Square Error (MSE), with values of 0.989, 4, and 0.539, respectively.

Keywords: Differential Equation, Gannet Optimization Algorithm, Shepard Convolutional Neural Network, Adam Optimization, Heterogeneous Parameters.

1. Introduction

Unlike traditional physical or mathematical methodologies, creating Data-driven Models (DDM) does not necessitate a thorough grasp of underlying mechanisms. This advantage is critical when modeling systems dynamics complex that have yet to be thoroughly investigated sufficiently to provide high-quality analytical models [10]. Generally, modern DDMs are based on statistical data analysis to describe the researched event, and machine learning algorithms to forecast system conditions [15]. Data-driven modeling has advanced significantly nowadays. The improvement of the machine learning algorithms, increased computer resources, and increased data collection from the investigated system have all contributed to this improvement. As a result, data-driven modeling has proliferated rapidly in recent years. This improvement was aided by general growth of machine learning methods, more computer resources, and growing data available from the examined system [8].

Partial and ordinary and Differential Equations (DE) are widely used in research and engineering. Except in the most straightforward circumstances, explicit solution formulas for DE are unavailable. As a result, numerical approximations of DE play an essential role in their study [9]. Because many principles models incorporate DE, such as energy, mass, and the momentum balances along with the process constraints such as rate laws and physical properties, the system of Differential-Algebraic Equations (DAEs), is widely used in the chemical engineering systems. DAE systems are frequently found in process control, chemical processes, and reactor design [3]. As the real world, scientific theory, and technology advance, an increasing number of applications use DE to construct models in order to identify real-world rules [1][6]. By solving DE, scientific and industrial fields are investigated [6].

Various numerical approaches have been developed to accurately and efficiently approximate DE. Popular approaches for solving the initial value issue for ordinary DE including Runge-Kutta and multi-step methods, as well as their references. Finite difference [11][9], finite volume, finite element, and spectral approaches [12] are popular numerical methods for estimating PDEs. The computer business has grown considerably in recent years. The ability of computers to retain data and perform scientific computations has advanced significantly and then sparked a flurry of deep learning investigation [13][6]. In addition, numerous deep learning algorithms are utilized for solving and building DE, and also there are certain shortcomings and flaws. Some of the equations also require

[1] Basic Department of ZhengZhou Vocational College of Industrial Safety, Zhengzhou, Henan, 451192, China

*Corresponding author e-mail: zhengzhousi66@126.com

mail.126.com

analytical solution of DE as the priori information, limiting their application [14][6]. As a result, the problem of solving and creating DE requires further research [6].

The main aim is to develop a new method for identifying DEs from data using deep learning and optimization algorithms. The AdamGOA uses a ShCNN to approximate observed data and automatically calculates derivatives. The AdamGOA_ShCNN generates significant amounts of meta-data and calculates derivatives. Using meta-data, integral form of DE is determined at any of the spatial-temporal point. The AdamGOA is developed and integrates two optimization methods, Adam and GOA. The advantages of AdamGOA produce better results in solving DE problems.

Key contributions of the paper are as follows:

- The observed data is used to solve the DE problems using AdamGOA. Moreover, Adam was combined with GOA to create AdamGOA, which computes the derivatives and optimizes the meta data. In addition, the ShCNN is the deep learning that is used to train the AdamGOA, hence, the accuracy is increased.
- Heterogeneous parameters are used to model the multi-objective fitness function that handles DE and to solve the integral form problems.

The remaining sections are laid out by: Section 2 includes literature review on problem solving in DE. Section 3 examines the AdamGOA_ShCNN for DE-solving systems. Section 5 calculates the AdamGOA_ShCNN efficiency in comparison to traditional approaches. Section 6 presents the conclusion.

2. Literature survey

Hao Xu *et al.* [1] created deep-learning with data-driven Partial Differential Equation (DL-PDE) to identify physical processes controlling PDEs. The method blends neural networks with data-driven PDE identification using sparse regressions. DL-PDE first trained a neural network, then generated a large amount of meta-data and used automatic differentiation to calculate the necessary derivatives. Finally, sparse regression is used to determine the shape of the PDE. The method produces satisfactory results when the data is limited and noisy but needs to evaluate experimental data from complex problems where the intrinsic PDEs are unknown.

Kailiang Wu and Dongbin Xiu [2] established a methodology for approximating unknown time-driven PDEs from solution data, rather than determining the terms in PDE. To restrict issue to finite dimensions, the recovery technique uses an approximation of evolution operator in the defined modal space. Next, a deep neural network e utilizing residual network (ResNet) is trained with a given set of data to provide the finite-dimensional approximation. The prediction accuracy was higher, but more complicated difficulties in the high dimensions, like electronic structure, were encountered.

Burcu Beykal *et al.* [3] developed optimization technique for systems with DAEs that are numerically infeasible. The numerical integration solution's stability constraint for DAE is expressed by SVM. After that, SVM models are incorporated with ARGONAUT framework to find the best solution. The SVM-based optimization approach efficiently solves multi-dimensional DAEs. However, the method should have examined how DAE solver options affect infeasible problems numerically and enhance fallouts by incorporating them as decision variables.

Huijuan Zhou *et al.* [4] developed an Improved Physics-Informed Neural Networks (IPINN) approach for restoring high-order solitons, data-driven solitons, and data-driven parameter discovery for VC-Hirota equation. The PINN was based on sub-neural networks for learning under varying noise intensity. The IPINN algorithm performs well in characterizing smooth solutions in narrow range. However, when a solution range and dynamic behavior increase, learning effect decreases.

Zichao Jiang *et al.* [5] developed Deep Neural Network (DNN)-based algorithm that solves linear equations with excellent accuracy. The algorithm integrates the DNN model-based corrective iteration technique with ResNet. This provides high computing efficiency as well as native hardware compatibility on the hybrid systems. Compared to traditional schemes, DNN solve many linear equations simultaneously with low processing complexity, but faces difficulties with inflexible issues.

Li Jing *et al.* [6] solved ordinary DE using polynomial function space, whereas linear combination of a simple function and their product yield's multinomial function spaces. A space function of polynomials is relatively simple, and their operation ability is more robust. As a result, by integrating genetic and deep learning algorithms, this research investigates and creates ideal model for the numerical DE solution, discovering that DNN essentially complete the data simulation. A fast and accurate technique for numerical function approximation is urgently needed.

Tamirat Temesgen Dufera [7] has introduced Deep Artificial Neural Networks to solve a system of ordinary DE. Increasing neuron size improves accuracy but requires more iterations to learn the parameters. Furthermore, arbitrary neuronal growth is not encouraged. Determine the optimal neuron size based on the situation at hand. For more significant domain values, the ANN technique outperforms the RK4 method in terms of accuracy. Additional analysis is needed to strengthen the DNN for ODE solving systems, such as delay DE and the stochastic DE. Thus, this method includes evaluating convergence, stability, and resilience for solving ODE systems.

Mikhail Maslyaev and Alexander Hvatov [8] have developed evolutionary algorithms and sparse regression to create a variety of equations that are determined meta-parameters rather than a pre-defined term set. The PDE discovery tool was used to gain metocean process equations continuously. The algorithm constructs the correct form of partial DE. However, additional tests were conducted other than sophisticated real-world and the synthetic input data from multiple sources.

The following are the issues discovered in the relevant work:

The numerical solution of DEs is complex due to the necessity to resolve spatiotemporal features across considerable lengths and timescales. Resolving the finer characteristics in a solution is often computationally challenging. The equation contains a higher-order differential term, which complicates the calculation of derivatives. The exponential increase in the computational power over last few decades has enabled solution with large-scale computational issues for DE, like Bayesian inverse problems, Uncertainty Quantification (UQ), optimal control, and the constrained optimization.

3. Problem Statement

Assume a state variable $v(y,z)$controlled by unknown autonomous time-dependent DE system.

$$\begin{cases} v_z = L(v), & (y,z) \in \delta \times \beta^+ \\ B(v) = 0, & (y,z) \in \alpha\delta \times \beta^+ \\ v(y,z) = v_0(y), y \in \bar{\delta} \end{cases} \tag{1}$$

where, z be the time, y be spatial variable, δ stands for physical domain, and the terms B and L represents the operators in the boundary and equations conditions. The underlying premise is that the operator L is constant. This work assumes known boundary conditions and focuses on learning DE within the domain.

Let us assume that data regarding the solution $v(y,z)$ is present at the specific time occurrences, termed "snapshots". Thus, the data is given as,

$$\omega(y, z_i) = v(y, z_i) + \in (y, z_i), \quad i = 1, \ldots, R \tag{2}$$

where, $R \geq 1$represents the total number of solution field snapshots, while $\in (y, z_i)$represents data acquisition noise/errors.

3.1 Proposed Methodology

The coefficients in the unified integral form might be variables or constants. Nevertheless, most DE methods are limited to PDEs with a constant coefficient. As a result, for DEs with a heterogeneous parameter, a step-by-step method is used, to first determine structure of DE and then compute the changing coefficients by that structure. In most circumstances, heterogeneous parameters are believed to be connected to a . Figure 1 depicts a diagrammatic depiction of our proposed technique for solving DEs issues with heterogeneous parameter sets.

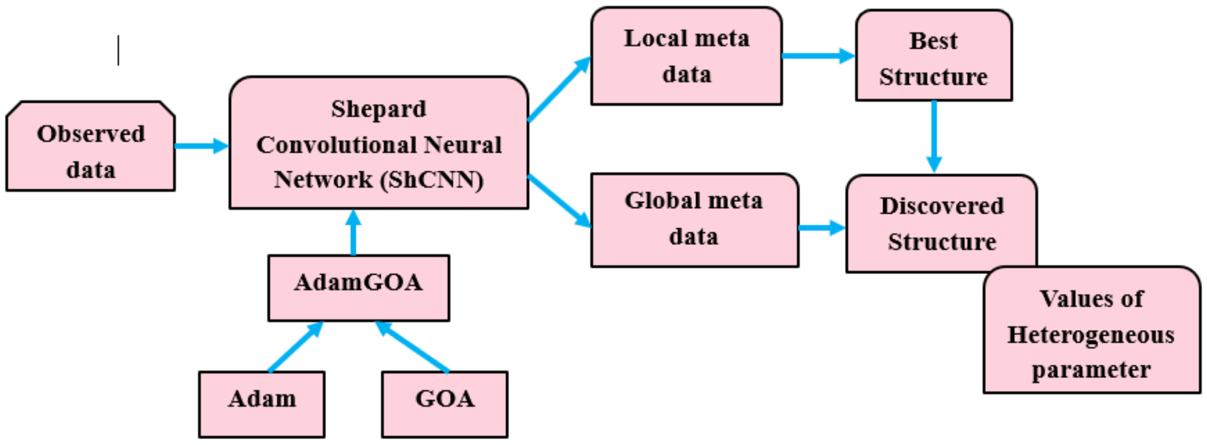


Figure 1. Detailed view of AdamGOA_ShCNN for solving DE

3.2 Differential Equation Structure

The stepwise procedure begins with the discovery of the DE structure, considering the difficulty of discovering both the structure and the parametric coefficients simultaneously. The whole domain is partitioned uniformly into M local windows in the image, with the trained ShCNN using AdamGOA generating local meta-data in each. Then, the ShCNN is used to find particular structure in every local window, yielding M potentials. The stability function is defined to measure confidence of revealed structure, expressed as,

$$K = \frac{M^{best}}{M^{local}} \tag{3}$$

where, K is stability, the optimal structure occurrences are represented as M^{best}, M^{local} is the local window. A higher K suggests a greater confidence in the observed DE structure.

3.3 Shepard Convolutional Neural Network

ShCNN [14] is a neural network identical to CNN except for the ReLU layer. In ShCNN, the Shepard interpolation layer replaces the ReLU layer. The Shepard model's convolution format is as follows:

$$H_q = \begin{cases} (F * P)_q (F * Z)_h & ; if\ Z_q = 0 \\ P^q & ; \ if\ Z_q = 1 \end{cases} \tag{4}$$

where P is input, H is output, the coordinates are h, Z be binary indicator, Z_q is anticipated values, $*$ denotes the convolution process, and F depicts the kernel function. Furthermore, the component-wise division is in charge of the information sent across the network. Moreover, the ShCNN can handle the interpolation of discontinuous-spaced data. Incorporating a Shepard interpolation layer in ShCNN results in an effective outcome.

-Shepard interpolation layer

The following represents the expression for the interpolation layer:

$$D_h^i\left(D^{i-1}, Z^i\right) = \varpi\left(\sum_p \frac{F_{hn}^i * D_n^{i-1}}{F_{hn}^i * Z_n^i} + r^i\right); i = 1,2,3,... \tag{5}$$

where q represents layer index, D_h^i denotes the feature map index, and D^{i-1} and Z^i be both the current layers input and mask, F_{hn} determines the trainable kernels. Weighting the Shepard interpolation layers together results in an exceedingly nonlinear interpolation operator. The term r be the bias, ϖ is network nonlinearity and the term D be smooth and differential function. Figure 2 depicts the ShCNN framework for solving DEs.

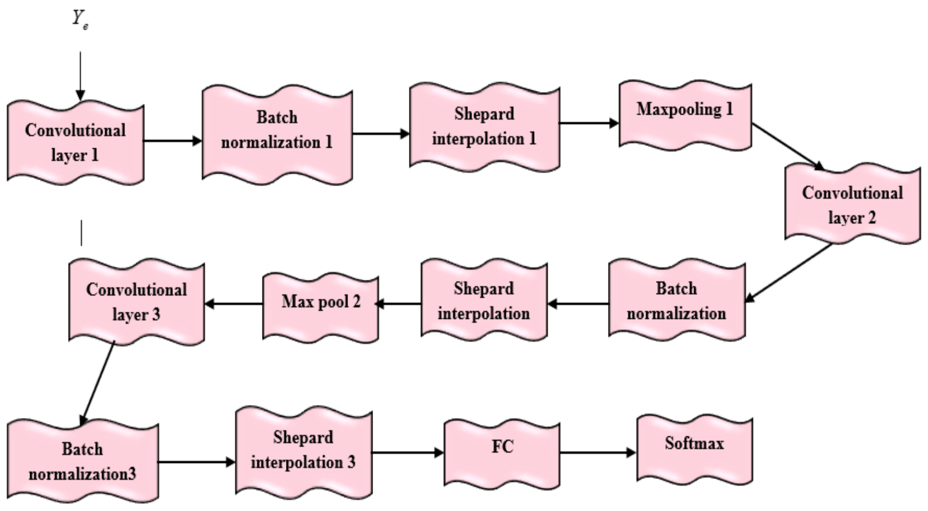


Figure 2. ShCNN structure

Thus, the ShCNN is used to solve the problems in DEs. In addition, the AdamGOA trains the ShCNN, which increases the system performance.

Training with AdamGOA

The ShCNN training with AdamGOA is seen below. AdamGOA is integrated with two optimization techniques, Adam [16] and GOA [17]. The GOA consists of U and V-shaped, which are likely to explore the optimum region across the search space. It ensures that the best answer is found in this region. GOA has quick running time in higher dimensions and provide the optimum option. It can address engineering design problems and can deliver the best solution in all situations. The GOA offers better processing capabilities for large-dimensional issues. Similarly, the Adam optimizer is a stochastic optimization strategy considers objective functions when updating locations. This method is suitable for a non-stationary target, as well as issues with excessively noisy or the sparse gradients. Furthermore, Adam's update rule selecting step sizes. The algorithmic procedure for AdamGOA is described below.

-Initialization

GOA begins with set of arbitrary solutions where the best solution is chosen as global solution and is formulated as,

$$N = \begin{bmatrix} g_{1,1} & \cdots & g_{1,l} & \cdots & g_{1,Q-1} & g_{1.Q} \\ g_{2,1} & \cdots & g_{2,l} & \cdots & g_{2,Q-1} & g_{2,Q} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \vdots & \vdots & g_l & \cdots & \cdots & \cdots \\ g_{U-1,1} & \cdots & g_{U-1,l} & \cdots & g_{U-1,Q-1} & g_{U-1,Q} \\ g_{U,1} & \cdots & g_{U,l} & \cdots & g_{U,Q-1} & g_{U,Q} \end{bmatrix} \tag{6}$$

where, g_r refers to the individual's location. Each element $g_{r,l}$ of the matrix N can be controlled by,

$$g_{r,l} = u_1 \times (G_l - A_l) + A, \; r = 1,2,\cdots U, \; l = 1,2,\cdots Q \tag{7}$$

where A_o, B_o denotes the upper and the lower boundaries of the dimensionality of the presented issue, V be total persons in population, U refers to the problem's dimensional size, and v_1 is a random number between 0 and 1.

Compute error

The fitness is produced with mean square error and is modeled as,

$$Fitness = \frac{1}{b}\sum_{v=1}^{b}[R_v - \rho]^2 \tag{8}$$

where, b, $Fitness$, ρ and R_υ is the total samples, fitness function, ShCNN output, and predicted output.

Exploration stage

Gannets look for the prey in the water via air, so they detect prey and change their dive pattern based on the prey's diving depth. The iteration model is as follows:

$$x = 1 - \frac{\rho}{J} \tag{9}$$

where, ρ is the current iteration count and J is the maximum iterations.

$$b = 2 * \cos(2 * \pi * u_2) * x \tag{10}$$

$$\mathrm{H} = 2 * J(2 * \pi * u_3) * x \tag{11}$$

where, random numbers are denoted as u_2 and u_3, and they range from 0 to 1.

$$J(g) = \begin{cases} -\frac{1}{\pi} * g + 1, g \in (0, \pi) \\ \frac{1}{\pi} * g - 1, g \in (\pi, 2\pi) \end{cases} \tag{12}$$

The next stage is to utilize two dive methods to update the position. Gannet has same likelihood of selecting among two models while predating. Hence, random number is assigned to choose between two dive methods. The position update is modeled as follows:

$$CO_a(k+1)=\begin{cases} C_a(k)+\beta 1+\beta 2 & ;\kappa \geq 0.5 \\ C_a(k)+\sigma 1+\sigma 2 & ;\kappa < 0.5 \end{cases} \tag{13}$$

$$\beta 2=\chi *\left(C_a(k)-C_p(k)\right) \tag{14}$$

$$\sigma 2=\upsilon *\left(C_a(k)-C_b(k)\right) \tag{15}$$

$$\chi=\left(2*u_4-1\right)*b \tag{16}$$

$$\upsilon=\left(2*u_5-1\right)*\mathrm{H} \tag{17}$$

where, u_4, u_5 denotes random values between 0 and 1, $\sigma 1$ depicts random numbers amidst $-\mathrm{H}$ and H, $\beta 1$ represents random numbers between $-b$ and b, $C_a(k)$ refers to an a^{th} individual in the current population, $C_p(k)$ refers randomly picked individual in current population, and $C_b(k)$ indicates average location of the individuals. It is calculated as:

$$C_b(k)=\frac{1}{U}\sum_{a=1}^{U} C_a(k) \tag{18}$$

Consider the equation's condition $\kappa \geq 0.5$,

$$OC_a(k+1)=C_a(k)+\beta 1+\beta 2 \tag{19}$$

Consider $OC_a(k+1)=C_a(k+1)$, $\beta 1=\beta_1$ and $\beta 2=\beta_2$

$$C_a(k+1)=C_a(k)+\beta 1+\beta 2 \tag{20}$$

Here, $\beta 2=\chi *\left(C_a(k)-C_p(k)\right)$ (21)

$$C_a(k+1)=C_a(k)+\beta 1+\chi *\left(C_a(k)-C_p(k)\right) \tag{22}$$

$$C_a(k+1)=C_a(k)+\beta 1+\chi * C_a(k)-\chi * C_p(k) \tag{23}$$

$$C_a(k+1)=C_a(k)[1+\chi]+\beta 1-\chi * C_p(k) \tag{24}$$

The GOA includes Adam's updated condition to improve its exploring aspects. According to Adam,

$$C_a(k)=C_a(k-1)-\frac{\tau . f^{\wedge}(c)}{\left(\sqrt{\mathrm{T}^{\wedge}(c)}+\mu\right)} \tag{25}$$

where, the term μ step size. Substitute equation (25) to (24),

$$C_a(k+1)=\left[C_a(k-1)-\frac{\tau . f^{\wedge}(c)}{\left(\sqrt{\mathrm{T}^{\wedge}(c)}+\mu\right)}\right][1+\chi]+\beta 1-\chi * C_p(k) \tag{26}$$

Thus, the above equation is the updated equation for solving DE problems.

Exploitation stage

Once the gannets rush into the ocean in two directions, two further scenarios are required for continued exploitation. Cunning fish are frequently depicted as turning fast movements to avoid gannet's chase. The gannet expends a lot of energy to catch fish desperately trying to escape. Thus, the capturability is modeled as,

$$\mathrm{Ez} = \frac{1}{\psi * \omega 2} \tag{27}$$

$$x2 = 1 + \frac{\rho}{J} \tag{28}$$

$$\psi = \frac{\gamma * vel^2}{\varepsilon} \tag{29}$$

$$\varepsilon = 0.2 + (2 - 0.2) * u_6 \tag{30}$$

Where, u_6 is a random value between 0 and 1, γ represents the gannet weight, which is 2.5kg, and the gannet speed is $vel = 1.5m/s$. The updated location is stated as:

$$OC_a(k+1) = \begin{cases} k * delta * \left(C_a(k) - C_{best}(k)\right) + C_a(k) & ; \mathrm{Ez} \geq \nu \\ C_{best}(k) - \left(C_a(k) - C_{best}(k)\right) * \ell * k & ; \mathrm{Ez} < \nu \end{cases} \tag{31}$$

$$delta = \mathrm{Ez} * | C_a(k) - C_{best}(k) | \tag{32}$$

$$\ell = Levy(\mathrm{Z}) \tag{33}$$

where, ν be constant of 0.2, $C_{best}(k)$ is optimal performing person, and $Levy(.)$ be Levy flight function. Levy flights are offered as,

$$Levy(Z) = 0.01 \times \frac{\hbar \times \Re}{|\Re|^{\frac{1}{\varphi}}} \tag{34}$$

$$\Re = \left(\frac{\mathrm{B}(1+\zeta) \times Sin\left(\frac{\pi\zeta}{2}\right)}{\mathrm{B}\left(\frac{1+\zeta}{2}\right) \times \zeta \times 2^{\left(\frac{\zeta-1}{2}\right)}} \right)^{\frac{1}{\zeta}} \tag{35}$$

where, ζ and $\hbar$ is the random number between 0 and 1, and $\zeta = 1.5$ denotes predefined constant.

Stimulate fault to find a superior solution

The error is regenerated for each solution in order to select the best solution using expression (6).

Termination

Repeat the preceding stages until the factor of decision is obtained. Table 1 specifies the AdamGBO pseudo code.

Table 1. Pseudo code of AdamGBO

```
**Input:** $K, Q, U$
**Output:** Best Gannet location $C_{best}(k)$
**Begin**
$C^{th}$ population is initialized
Memory matrix $OC$ is generated
Evaluate fitness by equation (12)
**While** stop requirement is not met,
  **If** $rand > 0.5$ then
    **for** $OC_a$ do
        **If** $\kappa \geq 0.5$ then
          Upgrade AdamGBO location by equation (26)
Update position by equation (13i)
        **else**
        **End if**
     **End for**
  **else**
    **for** $OC_a$ do
        **If** $\nu \geq 0.2$ then
          Update location using equation (31i)
        **else**
          Update location using equation (31ii)
        **End if**
      **End for**
  **End if**
  **For** $OC_a$ do
    Evaluate fitness of $OC_a$
    If the previous value is better, then replace it.
  **End for**
**End while**
**End**
```

3.4 Computation of heterogeneous parameters

While structure of DE is done, a coefficients need to be changed depending on global meta-data from a more extensive domain. To determine whether the coefficients of the discovered term are constants or variables, it is assumed that all coefficients are initially fluctuating. The discovered structure is written as follows:

$$\int_{\mu_j} v_S dx = \sum_{m=0}^{M_{terms}} \left[B_n(a) E_m \right]_{\mu_j} \tag{36}$$

where, the terms E_m and M_{terms} be terms and total terms in DF structure. For a one-dimensional issue, $\mu_j = \left[a_j - \frac{1}{2} I, a_j + \frac{1}{2} I \right]$, and hence the above equation is re-written by,

$$\int_{\mu_j} v_S dx = \sum_{m=0}^{M_{terms}} \left[B_n\left(a_j + \frac{1}{2} I \right) E_m\left(a_j + \frac{1}{2} I, z \right) - B_n\left(a_j - \frac{1}{2} I \right) E_m\left(a_j - \frac{1}{2} I, z \right) \right] \tag{37}$$

To determine the value of $B_n(a)$ at every point, an integral interval I is set to $2\Delta a$, with $\Delta a = a_{j+1} - a_j$. Metadata are generated on an evenly dispersed spatial-temporal basis, and Δa be constant. Then, the above equation is transformed into multi-element coupled linear equations using this specific value of I.

$$\int_{\mu_j} v_S(a_j, z_i) dx = \sum_{m=0}^{M_{terms}} \left[B_n(a_{j+1}) E_m(a_{j+1}, z_i) - B_n(a_{j+1}) E_m(a_{j+1}, z_i) \right] \tag{38}$$

where, $j = 2,3,...,M_a^{meta} - 1$ and $j = 1,2,...,M_z^{meta}$. In a multi-element coupled linear equations, $B_n(a_j)$ is unknown, resulting in $M_{terms}(M_a^{meta} - 2)$undetermined values. Hence, the above equation is a overdetermined because meta M_z^{meta} exceeds M_{terms}. As a result, it is reduced to a problem of solving linear equations, as follows:

$$x = Z\varpi \tag{39}$$

where, the term x be the vector, the coefficient matrix is denoted as Z, and ϖ denotes the vector of $B_n(a_j)$ that is resolved by the size $M_{terms}(M_z^{meta} - 2) \times 1$.

4. Results and discussion

This section presents findings and analyses of a AdamGOA_ShCNN for solving DE issues.

4.1. Experimental setup

AdamGOA_ShCNN is run in MATLAB under Windows 10. Here, the optimal layers and neurons in every layer are simulated.

4.2 Performance metrics

The AdamGOA_ShCNN performance is measured by Mean Square Error (MSE), convergence and the accuracy. Here, the MSE metric is clearly mentioned in equation (8).

Convergence: Convergence is the property of getting closer to a limit when an argument of the function maximized or minimized, or as the total amount of variables in series increases.

Accuracy: It measures how closely expected and predicted values match. Furthermore, the equation used for the accuracy test is given below:

$$Acc = \frac{\mathrm{T}^{pos} + \mathrm{T}^{neg}}{\mathrm{T}^{pos} + \mathrm{T}^{neg} + \mathrm{E}^{pos} + \mathrm{E}^{neg}} \tag{40}$$

where, the terms $\mathrm{T}^{pos}, and\, \mathrm{T}^{neg}$ represents true positive and negative. $\mathrm{E}^{pos}, \mathrm{E}^{neg}$ be false positive and negative.

4.3 Evaluation of Achievement

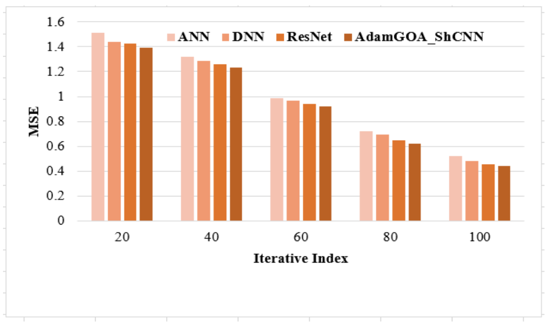
The AdamGOA_ShCNN's performance is evaluated using a variety of metrics. Then, the comparison is made to analytical solution and numerical solution generated through traditional methods.

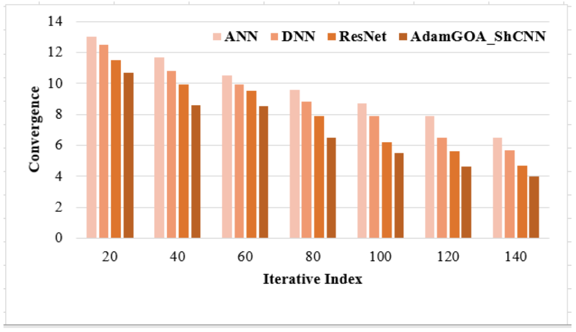
4.4 Comparative methods

The performance improvement of AdamGOA_ShCNN is compared to typical techniques, such as ANN [7], DNN [5], and ResNet [2].

4.5 Comparative assessment

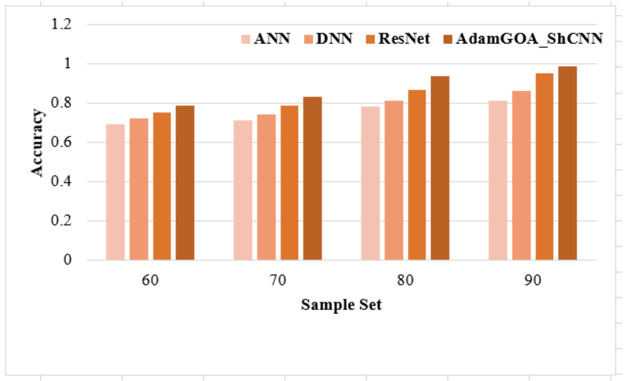
Figure 3 depicts the AdamGOA_ShCNN being evaluated using performance metrics in which the iterative index and sample data are modified to ascertain accuracy, MSE, and convergence. Figure 3a) depicts the MSE-based evaluation after varying the iterative index. AdamGOA_ShCNN's MSE value for iterative index=20 is 1.391, whereas the MSE values for standard techniques such as ANN, DNN, and ResNet are 1.512, 1.442, and 1.423, respectively. While the MSE of AdamGOA_ShCNN drops as iteration indexes grow, performance appears to remain consistent. DE problem solution produces the greatest results when the iterative index is 100, ensuring the influence of successive experiments. Figure 3b) shows the convergence curve after altering the iterative index. While AdamGOA_ShCNN improved by 8.6 for the 40th iteration index, ANN, DNN, and ResNet achieved convergence values of 11.7, 10.9, and 9.9. The smallest value indicates the best performance. When the iterative index is set to 120, the ANN convergence is 7.9, DNN is 6.5, ResNet is 5.6, and AdamGOA_ShCNN is 4.6. When compared to AdamGOA_ShCNN, ResNet has the lowest convergence, ranking second for DE issue solving. Figure 3c) depicts the findings of the accuracy-based assessment. In this study, AdamGOA_ShCNN outperforms the other three classic approaches for tackling DE difficulties. Thus, when the sample set is 60, the various algorithms have accuracy values of 0.691, 0.724, 0.753, and 0.786, respectively. For sample set 60, the corresponding accuracy improvement percentages are 9.5%, 6.2%, and 3.3%. Thus, AdamGOA_ShCNN addresses the concerns with DE by incorporating a set of variables as well as user input. However, it requires a huge number of repetitions for a small number of neurons, which causes a computational time difficulty. Increasing the number of neurons improves the model's performance. However, an arbitrary increase is unneeded.





(i)

(ii)



(iii)

Figure 3. Assessment of AdamGOA_ShCNN a) MSE, b) Convergence, and c) Accuracy

4.6 Comparative Discussion

Table 1 compares AdamGOA_ShCNN according to assessment measures. The AdamGOA_ShCNN attained convergence, accuracy, and MSE values of 4, 0.989, and 0.539. Similarly, established techniques, such as ANN, DNN, and ResNet, produced MSEs of 0.618, 0.582, and 0.553, respectively, with convergence of 6.5, 5.4, and 4.2 and accuracy of 0.813, 0.861, and 0.953. The efficacy of integrating heterogeneous parameters in ShCNN improved the performance metrics of the created model. The table below shows that ANN, DNN, and ResNet have more accuracy than the comparison approach. In the majority of cases, the ANN, DNN, and ResNet surpassed the other approaches in terms of accuracy. As a result, AdamGOA_ShCNN can effectively tackle the difficulties.

Table 5. Comparative discussion

Metrics	ANN	DNN	ResNet	AdamGOA_ShCNN
Accuracy	0.812	0.861	0.953	**0.989**
Convergence	6.6	5.4	4.2	**4**
MSE	0.618	0.582	0.553	**0.539**

5. Conclusion

This study presents AdamGOA_ShCNN, which finds the governing equation of underlying DEs by integrating two optimization procedures. Unlike most prior research, which focused on finding the differential form, the goal is to find a unified integral form of DE that can be used to DEs with both constant and variable coefficients. Here, sparse or noisy observation data are inputted to ShCNN. The ShCNN is trained by Adam and GOA to generate meta-data and compute derivatives through automatic differentiation, then calculate integral terms. In addition, the heterogeneous parameters are included to find the optimal structure, and the appropriate coefficient values are calculated. The AdamGOA_ShCNN demonstrated remarkable performance, with a high accuracy of 0.989, a low convergence, and MSE of 4, and 0.539. In future, more sophisticated optimization techniques will be used to examine the AdamGOA_ShCNN flexibility.

References

[1] Hao Xu, Haibin Chang, and Dongxiao Zhang,"DL-PDE: Deep-learning based data-driven discovery of partial differential equations from discrete and noisy data", arXiv preprint arXiv:1908.04463, 2019.

[2] Kailiang Wu and Dongbin Xiu," Data-driven deep learning of partial differential equations in modal space", Journal of Computational Physics, vol.408, pp.109307, 2020.

[3] Burcu Beykal, Melis Onel, Onur Onel, and Efstratios N. Pistikopoulos,"A data-driven optimization algorithm for differential algebraic equations with numerical infeasibilities", AIChE journal, vol.66, no.10, pp. e16657, 2020.

[4] Huijuan Zhou, Juncai Pu, and Yong Chen, "Data-driven forward-inverse problems for the variable coefficients hirota equation using deep learning method", Nonlinear Dynamics, vol.111, no.16, pp.14667-14693, 2023.

[5] Zichao Jiang , Junyang Jiang , Qinghe Yao, and Gengchao Yang,"A neural network-based PDE solving algorithm with high precision", Scientific Reports, vol.13, no.1, p.4479, 2023.

[6] Li Jing,"Data simulation of optimal model for numerical solution of differential equations based on deep learning and genetic algorithm", Soft Computing, vol.27, no.14, pp.10025-10032, 2023.

[7] Tamirat Temesgen Dufera,"Deep neural network for system of ordinary differential equations: Vectorized algorithm and simulation", Machine Learning with Applications, vol.5, p.100058,2021.

[8] Mikhail Maslyaev, and Alexander Hvatov,"Discovery of the data-driven differential equation-based models of continuous metocean process", Procedia Computer Science, vol.156, pp.367-376, 2019.

[9] Siddhartha Mishra,"A machine learning framework for data driven acceleration of computations of differential equations", arXiv preprint arXiv:1807.09519, 2018.

[10] Hongli Sun, Muzhou Hou, Yunlei Yang, Tianle Zhang, Futian Weng, and Feng Han,"Solving Partial Differential Equation Based on Bernstein Neural Network and Extreme Learning Machine Algorithm",Neural Processing Letters, vol.50, pp.1153-1172, 2019.

[11] R. J. LeVeque,” Finite difference methods for ordinary and partial differential equations, steady state and time dependent problems”, SIAM, 2007.

[12] Nirmal Kumar, Ali Akbar Shaikh, Sanat Kumar Mahato, and Asoke Kumar Bhunia," Applications of new hybrid algorithm based on advanced cuckoo search and adaptive Gaussian quantum behaved particle swarm optimization in solving ordinary differential equations", Expert Systems with Applications, vol.172, pp.114646, 2021.

[13] A. Malek, and R. Shekari Beidokhti,"Numerical solution for high order differential equations using a hybrid neural network-Optimization method",Applied Mathematics and Computation, vol.183, no.1, pp.260-271, 2006.

[14] Steven L. Bruntona, Joshua L. Proctor, and J. Nathan Kutz,"Discovering governing equations from data by sparse identification of nonlinear dynamical systems", In Proceedings of the national academy of sciences, vol.113, no.15, pp.3932-3937, 2016.

[15] Hayden Schaeffer, "Learning partial differential equations via data discovery and sparse optimization", In proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol.473, no.2197, pp.20160446, 2017.

[16] Kingma, D.P. and Ba, J., “Adam: A method for stochastic optimization”, arXiv preprint arXiv:1412.6980, 2014.

[17] Pan, J.S., Zhang, L.G., Wang, R.B., Snášel, V. and Chu, S.C., “Gannet optimization algorithm: A new metaheuristic algorithm for solving engineering optimization problems”, Mathematics and Computers in Simulation, vol.202, pp.343-373, 2022.

[18] Hao Xu, Dongxiao Zhang and Nanzhe Wang, "Deep-learning based discovery of partial differential equations in integral form from sparse and noisy data",Journal of Computational Physics, vol.445, pp.110592, 2021.

[19] Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis,"DeepXDE: A Deep Learning Library for Solving Differential Equations", SIAM review, vol.63, no.1, pp.208-228, 2021.