mathematics

MDPI

Article

Differential Transform Method and Neural Network for Solving Variational Calculus Problems

Rafał Brociek *,†, **Mariusz Pleszczyński** †

Department of Mathematics Applications and Methods for Artificial Intelligence, Faculty of Applied Mathematics, Silesian University of Technology, 44-100 Gliwice, Poland; mariusz.pleszczynski@polsl.pl
* Correspondence: rafal.brociek@polsl.pl
† These authors contributed equally to this work.

Abstract: The history of variational calculus dates back to the late 17th century when Johann Bernoulli presented his famous problem concerning the brachistochrone curve. Since then, variational calculus has developed intensively as many problems in physics and engineering are described by equations from this branch of mathematical analysis. This paper presents two non-classical, distinct methods for solving such problems. The first method is based on the differential transform method (DTM), which seeks an analytical solution in the form of a certain functional series. The second method, on the other hand, is based on the physics-informed neural network (PINN), where artificial intelligence in the form of a neural network is used to solve the differential equation. In addition to describing both methods, this paper also presents numerical examples along with a comparison of the obtained results.Comparingthe two methods, DTM produced marginally more accurate results than PINNs. While PINNs exhibited slightly higher errors, their performance remained commendable. The key strengths of neural networks are their adaptability and ease of implementation. Both approaches discussed in the article are effective for addressing the examined problems.

Keywords: variational calculus; ordinary differential equation; differential transform method; physics-informed neural network

MSC: 65L05

Citation: Brociek, R.; Pleszczyński, M. Differential Transform Method and Neural Network for Solving Variational Calculus Problems. *Mathematics* **2024**, *12*, 2182. https://doi.org/10.3390/math12142182

Academic Editor: Quanxin Zhu

Received: 25 June 2024
Revised: 5 July 2024
Accepted: 8 July 2024
Published: 11 July 2024

1. Introduction

The issues in the calculus of variations have rich applications in many different problems [1]. Using these methods, one can, for example, search for curves connecting fixed points (also considering problems with a moving boundary) and satisfying certain conditions (such as minimal length or the shortest time to travel between them), surfaces meeting specific conditions (such as the surface with the smallest area connecting two fixed circles), problems concerning the propagation of light in inhomogeneous media, isoperimetric problems, geodesic curves, quantum mechanics, hydrodynamics, and many others [2–5].

The problem of variational calculus can be described in many different ways; this paper focuses on finding the extremum of the following functional:

$$\mathcal{J}(y(x)) = \int_a^b F\Big(x, y(x), y'(x), \ldots, y^{(n)}(x)\Big)dx, \tag{1}$$

where $y(x)$ is the sought-after n-times differentiable function defined in the interval $[a, b]$, which satisfies the following boundary conditions:

$$y^{(i)}(a) = A_i, \quad y^{(i)}(b) = B_i, \quad 0 \leq i \leq n-1, \tag{2}$$

where $a, b \in \mathbb{R}$, $a < b$, $A_i, B_i \in \mathbb{R}$, $0 \leq i \leq n-1$, and F is a certain function of $(n+2)$ variables' $(n+1)$-times differential, defined in the set $[a, b] \times \mathbb{R}^{n+1}$.

For such a formulated problem, a necessary condition for the functional (1) to attain a local extremum for the sought-after function $y(x)$ under the boundary conditions (2) is that the function F satisfies the following differential equation:

$$\frac{\partial F}{\partial y} - \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right) + \frac{d^2}{dx^2}\left(\frac{\partial F}{\partial y''}\right) + \ldots + (-1)^n \frac{d^n}{dx^n}\left(\frac{\partial F}{\partial y^{(n)}}\right) = 0, \tag{3}$$

(this is the so-called Euler–Poisson equation) with the conditions (2) [6,7].

In the general case, the condition (3) is only a necessary condition for the existence of an extremum of the functional (1). However, in most technical problems, it is also a sufficient condition. Therefore, in this paper, we search for these extrema in this manner.

Equation (3) is a differential equation of the order $2n$ and appears relatively rarely for larger values of n. It is more commonly encountered for $n = 1$ or $n = 2$. In these cases, it takes the following form:

$$\frac{\partial F}{\partial y} + \sum_{i=1}^{n} (-1)^i \frac{d^i}{dx^i}\left(\frac{\partial F}{\partial y^{(i)}}\right) = \begin{cases} \dfrac{\partial F}{\partial y} - \dfrac{d}{dx}\left(\dfrac{\partial F}{\partial y'}\right), & \text{for } n = 1, \\ \dfrac{\partial F}{\partial y} - \dfrac{d}{dx}\left(\dfrac{\partial F}{\partial y'}\right) + \dfrac{d^2}{dx^2}\left(\dfrac{\partial F}{\partial y''}\right), & \text{for } n = 2. \end{cases}$$

In the special case of $n = 1$, this equation is known as the Euler equation. Often, there are problems where the function F depends on many unknown functions, e.g., $y_i(x)$ (and their derivatives). In such cases, the necessary condition for the existence of an extremum of the functional (the Euler–Lagrange equations) reduces to a system of differential equations.

Unfortunately, very often, the differential equations (or their systems) obtained from the corresponding conditions are nonlinear or cannot be expressed in normal form (which is required by many methods for solving such problems, including classical ones like the Runge–Kutta method and less-known ones like the Adomian decomposition method [8,9]). For this reason, we need to seek new methods that can handle such forms of the problem.

Such methods include the differential transformation method (DTM) discussed in Section 2 and the physics-informed neural networks (PIINs) presented in Section 3. PINNs are a type of specifically designed neural network devoted to solving problems involving differential equations that describe physical laws. PINNs integrate knowledge of physical laws, allowing for the more accurate and efficient modeling of physical phenomena compared with traditional neural networks. PINNs can generalize better to unseen conditions because they are guided by underlying physical laws, which remain consistent across different scenarios. This contrasts traditional neural networks that might struggle to extrapolate beyond their training data. Section 4 is devoted to numerical examples illustrating the effectiveness of both methods and comparing them using selected examples.

2. DTM Method

The differential transform method (DTM) was proposed at the end of the 20th century and was quickly recognized and adapted to solve many different problems. Initially, it was used to solve ordinary differential equations, but its applications have expanded to the systems of such equations [10–15], partial differential equations [16,17], differential algebraic equations [18], integral equations [19,20], delay differential equations [21,22], fuzzy differential equations [23,24], and fractional differential equations [25–28]. This method has also been applied to certain important types of equations, such as Schrödinger equations and nonlinear Klein–Gordon equations, and in modeling predator–prey phenomena [29–31], among many other applications [32–36].

If a function, f, can be expanded into a Maclaurin series (such a function is called the original), then for this function, we can write the following:

$$f(x) = \sum_{k=0}^{\infty} \frac{f^{(k)}(0)}{k!} x^k = \sum_{k=0}^{\infty} F(k) x^k,$$

Thus, knowing the properties of such a transformation and the transforms of basic functions, we can solve many of the problems mentioned in the previous paragraph.

Below, we present the transforms of selected functions and selected properties of DTM (the included functions and properties were used in the examples presented in Section 4; proofs of these properties, among others, can be found in [16,25]).

Theorem 1. *Let x be an element of the domain of the considered function, f, and $k \in \mathbb{Z}_0$.*
If $f(x) = x^m$, $\mathbb{Z} \ni m \geq 0$, then

$$F(k) = \delta(k-m) = \begin{cases} 1, & k = m, \\ 0, & k \neq m. \end{cases} \tag{4}$$

If $f(x) = e^{ax}$, $a \in \mathbb{R}$, then

$$F(k) = \frac{a^k}{k!}. \tag{5}$$

If $f(x) = \sin(ax)$, $a \in \mathbb{R}$, then

$$F(k) = \frac{a^k}{k!} \sin \frac{k\pi}{2}. \tag{6}$$

If $f(x) = \cos(ax)$, $a \in \mathbb{R}$, then

$$F(k) = \frac{a^k}{k!} \cos \frac{k\pi}{2}. \tag{7}$$

If $f(x) = u(x) \pm v(x)$, then

$$F(k) = U(x) \pm V(x). \tag{8}$$

If $f(x) = a \cdot u(x)$, $a \in \mathbb{R}$, then

$$F(k) = a \cdot U(x). \tag{9}$$

If $f(x) = x^m \cdot u(x)$, then

$$F(k) = \begin{cases} 0, & k < m, \\ U(k-m), & k \geq m. \end{cases} \tag{10}$$

If $f(x) = u(x) \cdot v(x)$, then

$$F(k) = \sum_{i=0}^{k} U(i)V(k-i) = \sum_{i=0}^{k} V(i)U(k-i). \tag{11}$$

If $f(x) = u(x) \cdot v(x) \cdot w(x)$, then

$$F(k) = \sum_{i=0}^{k} \sum_{j=0}^{k-i} U(i)V(j)W(k-i-j) = \sum_{i=0}^{k} \sum_{j=0}^{i} U(j)V(i-j)W(k-i). \tag{12}$$

If $f(x) = u^{(n)}(x)$, then

$$F(k) = \frac{(k+n)!}{k!} U(k+n). \tag{13}$$

3. Physics-Informed Neural Network

Physics-informed neural networks (PINNs) represent a novel approach to solving differential equations (both ordinary and partial) and differential-integral equations, har-

nessing the power of deep learning. Among the pioneering works on this approach are the papers by Raisi et al. [37] and Karniadakis et al. [38]. PINNs integrate the principles of physics in the form of differential equations with a neural network architecture. This methodology embeds differential equations and initial/boundary conditions into the loss function of the neural network, ensuring that predictions align with the equations describing the system. As a result, PINNs offer a flexible and efficient alternative to traditional numerical methods, which often face limitations related to computational costs and scalability, especially in high-dimensional spaces. Once the model is trained, it can provide solutions for any grid, making this approach advantageous over classical methods like finite difference schemes. Additionally, PINNs excel in handling complex boundary and initial conditions, and they can directly incorporate experimental data into the model. This capability not only enhances solution accuracy but also enables the model to learn and adapt to real-world scenarios where data may be sparse or noisy. Consequently, PINNs have achieved significant success in various applications, including fluid dynamics, materials science, and biological systems [39–41].

A PINN architecture consists of classical components of the feedforward and backpropagation of neural networks: hidden layers, nodes per hidden layer, and activation functions. To train a neural network, an optimization method like the adam optimizer is used. The PINN system presented in the article consists of two main modules (see Figure 1): a fully connected neural network and a loss function module that computes the loss as the weighted $L2$ norm of both the governing ODE equation and boundary condition's residuals.

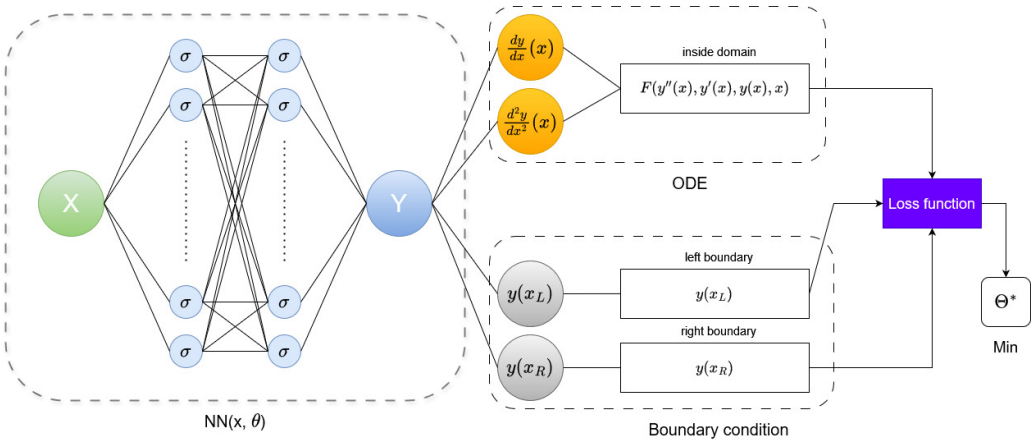

Figure 1. Diagram of physics-informed neural network for ODE with Dirichlet boundary conditions.

For the considered ordinary differential equation, the following applies:

$$F(y^{(n)}(x_i), \ldots, y''(x_i), y'(x_i), y(x_i), x_i) = 0, \quad x \in [x_L, x_R],$$

With Dirichlet boundary conditions, the following apply:

$$y(x_L) = y_L, \quad y(x_R) = y_R,$$

The loss function takes the following form:

$$\mathcal{L} = \sum_{i=1}^{N} F(y^{(n)}(x_i), \ldots, y''(x_i), y'(x_i), y(x_i), x_i) + (y(x_L) - y_L) + (y(x_R) - y_R), \tag{14}$$

where N is the number of collocation/training points inside domain $[x_L, x_R]$. After building the neural network and defining the loss function, the neural network is then trained to find the best parameters, Θ^* (weights and biases), by minimizing the loss function, $\mathcal{L}$.

4. Numerical Examples

4.1. Example 1

Let us consider a functional,

$$\mathcal{J}(y(x)) = \int_0^{\ln 3} \Big((y'(x) - y(x) + x)^2 - x - y(x) \Big) dx \tag{15}$$

with boundary conditions,

$$y(0) = -3, \quad y(\ln 3) = \ln 3. \tag{16}$$

In this case,

$$F(x, y(x), y'(x)) = (y'(x) - y(x) + x)^2 - x - y(x),$$

and therefore, according to (3), for $n = 1$,

$$\frac{\partial F}{\partial y} - \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right) = -2(y'(x) - y(x) + x) - 1 - 2(y''(x) - y'(x) + 1),$$

which, after some transformation, gives the following equation:

$$2y''(x) - y(x) + 2x + 3 = 0. \tag{17}$$

Taking into account the conditions (16), we therefore have to solve an ordinary second-order differential equation (a linear, non-homogeneous one with constant coefficients), where the exact solution to this equation is the following function:

$$y(x) = \frac{3}{2} + x - \frac{9}{2}e^{-x}.$$

4.1.1. DTM—Example 1

Applying the DTM to Equation (17) (utilizing properties (4), (8), (9), (13)), for $k \geq 0$, we obtain the following:

$$2(k+2)(k+1)Y(k+2) - 2Y(k) + 2\delta(k-1) + 3\delta(k-0) = 0, \tag{18}$$

whereby from the condition $y(0) = -3$, we know that $Y(0) = -3$. Assuming $k = 0$ in (18), we obtain the following:

$$2 \cdot 2Y(2) - 2Y(0) + 2\delta(-1) + 3\delta(3) = 0 \Rightarrow Y(2) = -\frac{9}{4}.$$

Notice that neither for $k = 0$ nor for $k > 0$ are we able to determine the value of $Y(1)$—this is because the second condition (16) takes the form $y(\ln 3) = \ln 3$ (the desired form would be $y'(0) = a \in \mathbb{R}$). Therefore, let us assume $Y(1) = s \in \mathbb{R}$. Taking $k \geq 1$, we sequentially obtain the following:

$$\begin{aligned} &Y(3) = \frac{s-1}{6}, \quad Y(4) = -\frac{3}{16}, \quad Y(5) = \frac{s-1}{120}, \quad Y(6) = -\frac{1}{160}, \\ &Y(7) = \frac{s-1}{5040}, \quad Y(8) = \frac{s-1}{362880}, \quad Y(9) = -\frac{1}{8960}, \quad Y(10) = -\frac{1}{806400}, \ldots \end{aligned}$$

In the general case, we can note that $Y(0) = -3$ and $Y(1) = s$; for $k > 1$, we have $Y(2) = -\frac{9}{4}$ and $Y(i+2) = \frac{Y(i)}{(i+1)(i+2)}$ for even numbers, i, and $Y(3) = \frac{s-1}{6}$ and $Y(j+2) = \frac{Y(j)}{(j+1)(j+2)}$ for odd numbers, j.

Therefore, we can represent the sought-after function, y, as follows:

$$y(x) = x + (s-1)\sum_{i=1}^{\infty} \frac{x^{2i-1}}{(2i-1)!} - 3 - \frac{9}{2}\sum_{i=1}^{\infty} \frac{x^{2i}}{(2i)!},$$

which, after using a known Taylor series, takes the following form:

$$y(x) = x + (s-1)\sinh x + \frac{3}{2} - \frac{9}{2}\cosh x.$$

There is still an unknown value of s to find. Using the second condition (16), i.e., $y(\ln 3) = \ln 3$, we obtain the following:

$$\ln 3 = \ln 3 + (s+1)\frac{4}{3} + \frac{3}{2} - \frac{9}{2}\cdot\frac{5}{3} \Rightarrow s = \frac{11}{2}.$$

This result allows for the final form of the following solution,

$$y = \frac{3}{2} + x - \frac{9}{2}e^{-x},$$

which fully coincides with the exact solution. Of course, this is not the standard case—in most instances, we can only find an approximate solution, as we present in the subsequent examples.

4.1.2. PINNs—Example 1

Now, we solve Equation (17) using neural networks. In this case, a fully connected neural network of a depth of 4 with 3 hidden layers and 10 neurons per layer is used. As an optimizer, the Adam method is used with a learning rate of 0.001. The hyperbolic tangent is assumed as the activation function. Figure 2 shows the obtained approximate solution (left plot, black dots) along with the errors of this approximation (right plot). The obtained solution fits the exact solution quite well. The maximum error is approximately 0.04.

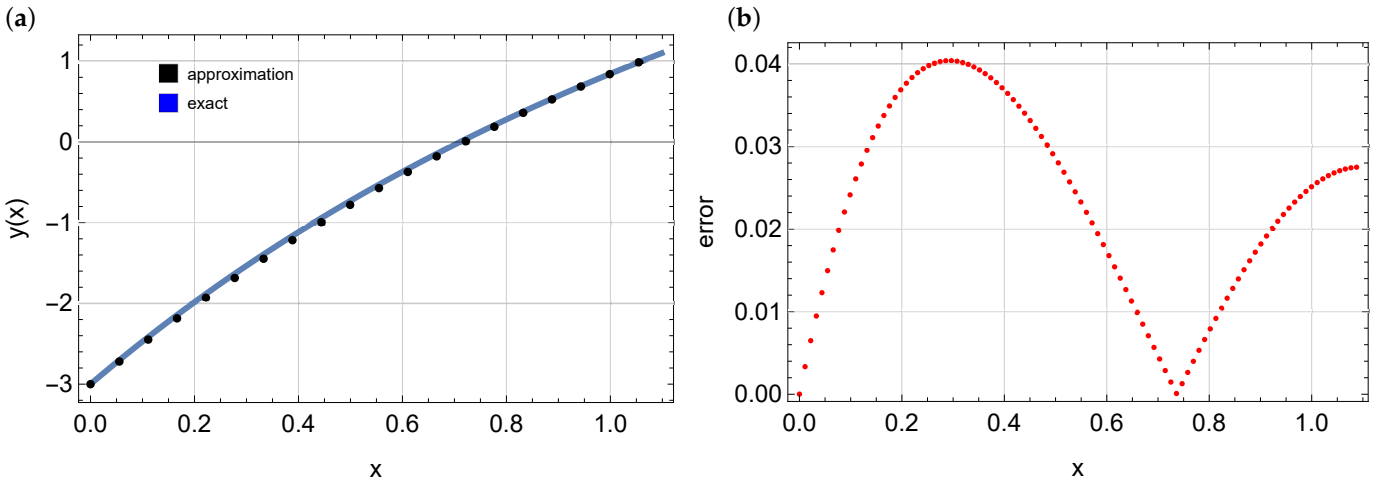

Figure 2. The exact solution, $y(x)$ (solid blue line); the approximate solution (black dots)—Figure (**a**); and the errors of the approximate solution—Figure (**b**), example 1.

4.2. Example 2

Consider following functional,

$$\mathcal{J}(y(x)) = \int_0^5 \frac{(1-y(x))^2}{(y'(x))^2}dx \tag{19}$$

with conditions

$$y(0) = 0, \quad y(5) = -\frac{3}{4}. \tag{20}$$

In this case, we have

$$F(x, y(x), y'(x)) = \frac{(1-y(x))^2}{(y'(x))^2},$$

and therefore we, can, according to (3) for $n = 1$, write

$$\frac{\partial F}{\partial y} - \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right) = -\frac{6(y(x)-1)(y(x)+1)\left(y(x)^2y''(x) - y''(x) - 2y(x)y'(x)^2\right)}{y'(x)^4},$$

which, after some transformation, leads to the equation

$$y(x)^2y''(x) - y''(x) - 2y(x)y'(x)^2 = 0 \tag{21}$$

with conditions (20) (other solutions, $y(x) = 1$ and $y(x) = -1$, are neglected because they do not meet the conditions (20)).

4.2.1. DTM—Example 2

Applying the DTM of transformation to Equation (21) (here, we use the properties (8), (9), (12), and (13)), for $k \geq 0$, we obtain

$$\begin{aligned}\sum_{i=0}^{k}\sum_{j=0}^{i}((j+1)(j+2)Y(j+2)Y(i-j)Y(k-i)) - (k+1)(k+2)Y(k+2)\\ - 2\sum_{i=0}^{k}\sum_{j=0}^{i}((j+1)(i-j+1)Y(j+1)Y(i-j+1)Y(k-i)) = 0,\end{aligned} \tag{22}$$

and according to the condition, $y(0) = 0$ implies that $Y(0) = 0$.

Similarly to before, taking successive values, $k \geq 0$, in (22), we are not be able to find the value of $Y(1)$. Therefore, assume $Y(1) = s \in \mathbb{R}$; as a result, $Y(0) = 0$, $Y(1) = s$, and

$$\begin{aligned}&Y(2) = 0, \quad Y(3) = -\frac{s^3}{3}, \quad Y(4) = 0, \quad Y(5) = \frac{2s^5}{15}, \quad Y(6) = 0,\\ &Y(7) = -\frac{17s^7}{315}, \quad Y(8) = 0, \quad Y(9) = \frac{62s^9}{2835}, \quad Y(10) = 0, \quad Y(11) = -\frac{1382s^7}{155,925}, \ldots\end{aligned}$$

This time, we are not able to find a function defined by the Taylor series of the obtained terms. Therefore, we must limit ourselves to finding an approximate solution, $y_n(x)$, of the form

$$y_n(x) = \sum_{i=0}^{n} Y(i)x^i,$$

which depends on the unknown value of parameter s. The value of this parameter can be found using the second of the conditions (20), i.e., $y(5) = -\frac{3}{4}$. Proceeding in this way and taking $n = 7$, an approximate solution, $y_7(x)$, of the following form is obtained:

$$y_7(x) = sx - \frac{s^3}{3}x^3 + \frac{2s^5}{15}x^5 - \frac{17s^7}{315}x^7.$$

Solving the equation $y_7(5) = -\frac{3}{4}$, we unfortunately obtain three real solutions: $s = -0.24919$, $s = -0.20286$, and $s = 0.35538$. Therefore, it is necessary to choose the appropriate one. To achieve this, we examine the error as follows:

$$\int_0^5 |y_7(x,s)^2y_7''(x,s) - y_7''(x,s) - 2y_7(x)y_7'(x,s)^2|dx.$$

In this case, it turns out that the best value is $s = -0.20286$. For this value, we obtain an approximate solution:

$$y_7(x) = -0.2063x + 0.0028x^3 - 0.00004x^5 + 7.63 \cdot 10^{-7}x^7.$$

Following a similar approach but taking $n = 11$ (again choosing the best value of s), we obtain the following solution:

$$y_{11}(x) = -0.1955x + 0.0025x^3 - 0.00004x^5 + 5.88 \cdot 10^{-7}x^7 - 9.11 \cdot 10^{-9}x^9 + 1.41 \cdot 10^{-10}x^{11}.$$

In Figure 3, we present the exact solution, $y(x)$, the approximate solutions, $y_7(x)$ and $y_{11}(x)$, and the errors, $\Delta_n(x)$, of these solutions, where

$$\Delta_n(x) = |y(x) - y_n(x)|,$$

and the exact solution has the following form:

$$y(x) = \frac{1 - 7^{x/5}}{1 + 7^{x/5}}.$$

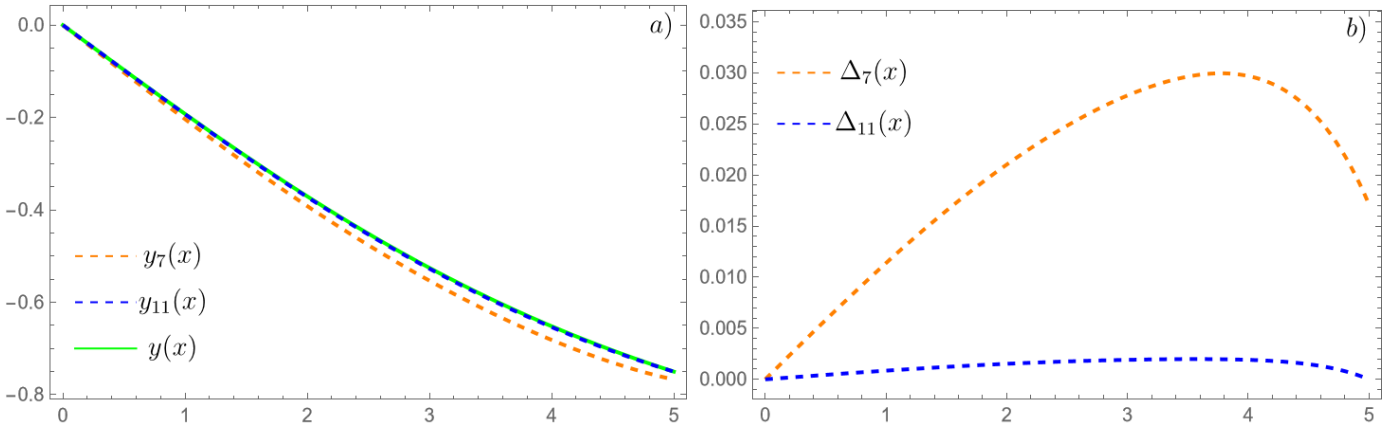

Figure 3. Exact solution, $y(x)$ (solid green line); approximate solution, y_7 (dashed orange line); approximate solution, y_{11} (dashed blue line)—Figure (**a**)—and errors, Δ, of these approximations—Figure (**b**), example 2.

4.2.2. PINNs—Example 2

The same Equation (21) was also solved using a physics-informed neural network. In this example, a fully connected neural network of a depth of 4 with 3 hidden layers and 10 neurons per layer was used. The Adam optimizer with a learning rate of 0.001 was employed and the hyperbolic tangent is assumed as the activation function. Inside the domain, 128 training points were considered. Figure 4 depicts the obtained approximate solution (left plot, black dots) along with the errors of this approximation (right plot). The obtained solution is very close to the exact solution. The maximum error does not exceed 0.016.

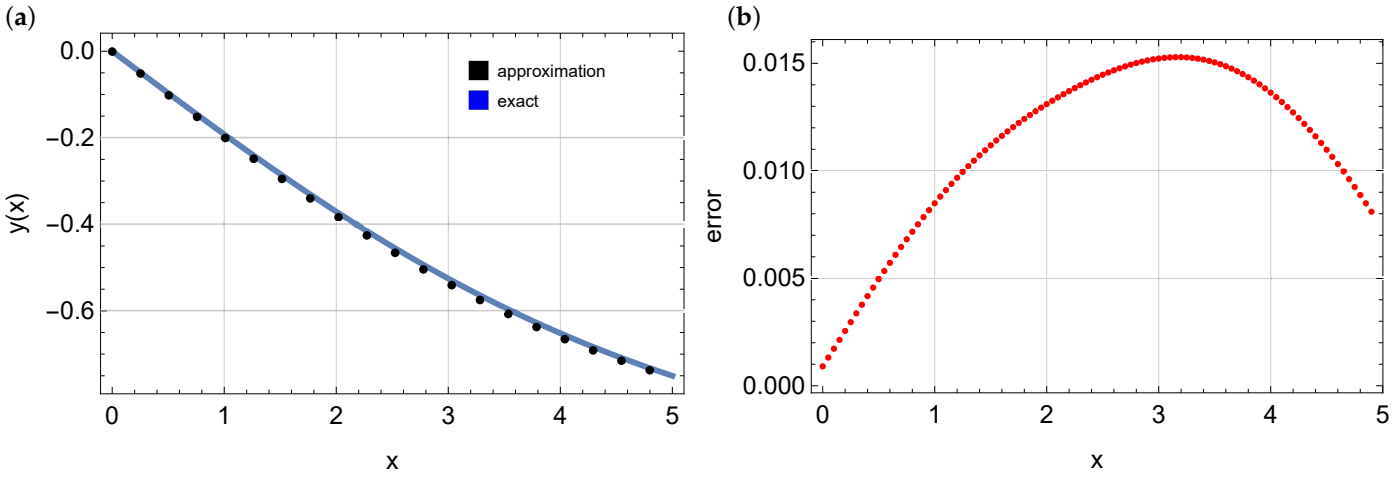

Figure 4. The exact solution $y(x)$ (solid blue line), the approximate solution (black dots)—Figure (**a**)—and the errors of the approximate solution—Figure (**b**), example 2.

4.3. Example 3

In the third example, the following equation is considered,

$$\mathcal{J}(y(x)) = \int_{-1}^{1} \Big(y''(x)y'(x)e^{-x}\sin x + (y'(x) - y''(x))^2 - (y(x) - y''(x))y'(x) \Big) dx \quad (23)$$

which is supplemented by the following conditions:

$$y(-1) = 1, \quad y'(-1) = -1, \quad y(1) = 1, \quad y'(1) = -2. \quad (24)$$

In this case,

$$F(x, y(x), y'(x)) = y''(x)y'(x)e^{-x}\sin x + (y'(x) - y''(x))^2 - (y(x) - y''(x))y'(x),$$

and therefore, according to (3), for $n = 2$, we obtain the following,

$$\frac{\partial F}{\partial y} - \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right) + \frac{d^2}{dx^2}\left(\frac{\partial F}{\partial y''}\right) = e^x\Big(2y^{(4)}(x)e^x + y''(x)(\cos x - \sin x - 2e^x) - 2y'(x)\cos x\Big),$$

which brings us to the follwing equation,

$$2y^{(4)}(x)e^x + y''(x)(\cos x - \sin x - 2e^x) - 2y'(x)\cos x = 0 \quad (25)$$

with conditions (24).

4.3.1. DTM—Example 3

Applying the DTM of transformation to Equation (25) (we use the properties (5)–(9), (11), and (13)), the following equation is obtained for $k \geq 0$:

$$\begin{aligned} 2\sum_{i=0}^{k} \frac{(i+1)(i+2)(i+3)(i+4)Y(i+4)}{(k-i)!} + \sum_{i=0}^{k} \frac{(i+1)(i+2)Y(i+2)\left(\cos\frac{\pi(k-i)}{2} - \sin\frac{\pi(k-i)}{2} - 2\right)}{(k-i)!} \\ - 2\sum_{i=0}^{k} \frac{(i+1)Y(i+1)\cos\frac{\pi(k-i)}{2}}{(k-i)!} = 0. \end{aligned} \quad (26)$$

Given successive values of $k \geq 0$ in (26), it is impossible to find the values $Y(0)$, $Y(1)$, $Y(2)$ and $Y(3)$, so we assume $Y(0) = s_0$, $Y(1) = s_1$, $Y(2) = s_2$ and $Y(3) = s_3$, $s_i \in \mathbb{R}$, $0 \leq i \leq 3$. Therefore $Y(0) = s_0$ i $Y(1) = s_1$, $Y(2) = s_2$, $Y(3) = s_3$, and

$$Y(4) = \frac{s_1 + s_2}{24}, \quad Y(5) = \frac{-s_1 + 4s_2 + 3s_3}{120}, \quad Y(6) = \frac{s_1 - 11s_2 + 36s_3}{1440},$$
$$Y(7) = \frac{11s_1 - 12s_2 - 69s_3}{10,080}, \quad Y(8) = \frac{-75s_1 + 109s_2 + 96s_3}{161,280}, \quad Y(9) = \frac{33s_1 - 184s_2 + 301s_3}{483,840}, \ldots$$

Again, we are unable to find a function with a Taylor series defined by these terms based on the form of the found terms. We must therefore limit ourselves to finding an approximate solution of the form $y_n(x)$,

$$y_n(x) = \sum_{i=0}^{n} Y(i)x^i,$$

which additionally depends on an unknown parameter, s_i, $0 \leq i \leq 3$. These parameters are determined using conditions (24) (this time we have a unique set of exactly four constants s_i). Proceeding in this way and taking $n = 4$, we obtain an approximate solution of the form $y_4(x)$:

$$y_4(x) = 1.2692 + 0.75x - 0.2885x^2 - 0.75x^3 + 0.0192x^4.$$

Proceeding similarly, but taking $n = 7$ (again the conditions (24) imply a unique solution to the parameters s_i, $0 \leq$ and ≤ 3), we obtain the following solution:

$$y_7(x) = 1.2398 + 0.7288x - 0.2448x^2 - 0.7024x^3 + 0.0202x^4 - 0.0318x^5 - 0.0152x^6 + 0.0053x^7.$$

Figure 5 shows the exact solution $y(x)$, the approximate solutions, $y_4(x)$ and $y_7(x)$, and the errors, $\Delta_n(x)$, of these solutions. In following formula of error,

$$\Delta_n(x) = |y(x) - y_n(x)|,$$

$y(x)$ denotes the exact solution obtained form Wolfram Mathematica 14.0 [42,43]. As can be seen in Figure 5, the difference in errors between solutions $y_4(x)$ and $y_7(x)$ is significant. The maximum error for $y_7(x)$ does not exceed 0.004, whereas for the solution $y_4(x)$, this error is approximately 0.034.

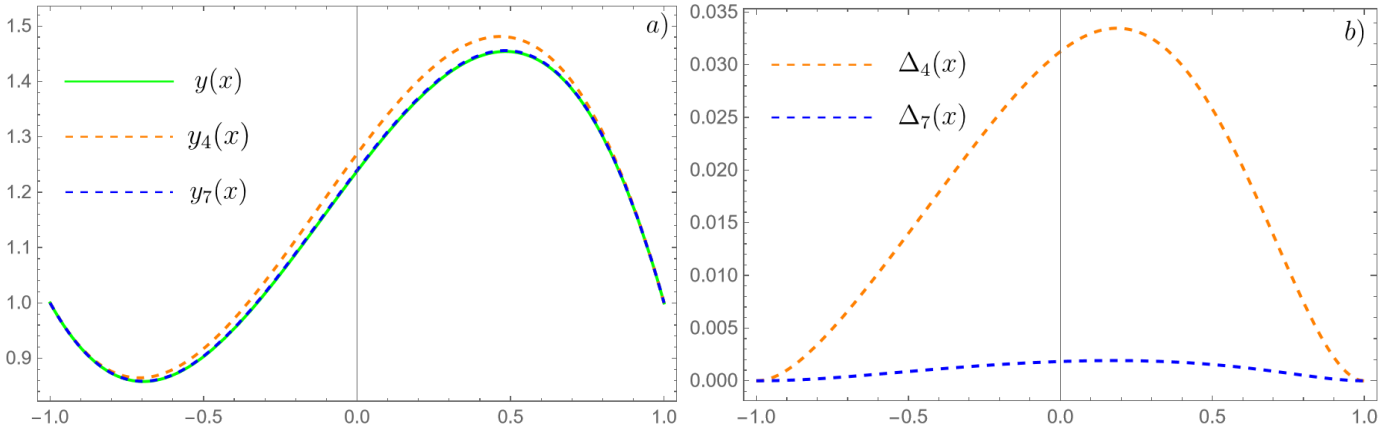

Figure 5. Exact solution, $y(x)$ (solid green line); approximate solution, y_7 (dashed orange line); approximate solution, y_{11} (dashed blue line)—Figure (**a**)—and errors Δ of these approximations—Figure (**b**), example 3.

4.3.2. PINNs—Example 3

Similar to previous examples, the following network architecture is assumed: 3 hidden layers with 10 neurons per layer. Inside the domain, 128 training points are used. The hyperbolic tangent is assumed as the activation function. The solution obtained after training the model is depicted in Figure 6. The solution using PINNs fits the exact solution quite well, with errors not exceeding 0.04.

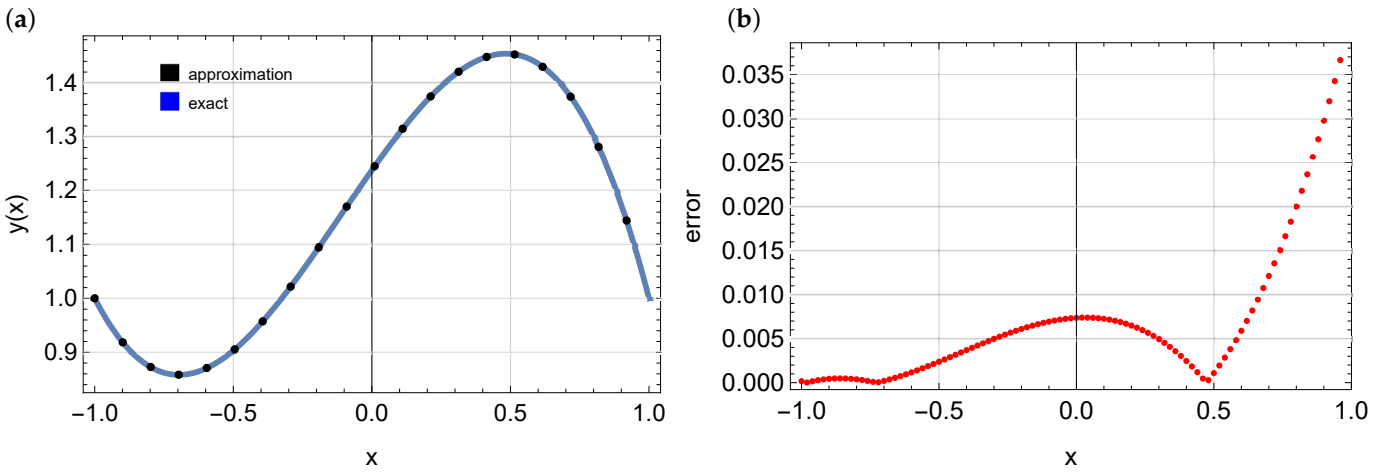

Figure 6. The exact solution $y(x)$ (solid blue line); the approximate solution (black dots)—Figure (**a**)—and the errors of the approximate solution—Figure (**b**), example 3.

4.4. Comparison of DTM and PINNs Results

In this section, errors obtained from the two methods are compared for examples 2 and 3. In the case of example 1, the DTM method provides the exact solution. Figure 7 presents error plots obtained from both methods, specifically for example 2 (Figure 7a) and example 3 (Figure 7b). As observed in Figure 7, the smallest errors were obtained for DTM with $n = 11$ (for example 2) and $n = 7$ (for example 3). The errors of the solutions obtained using neural networks are slightly larger but still smaller than those from DTM with $n = 7$ (for example 2) and $n = 4$ (for example 3). Both methods produced solutions of good quality.

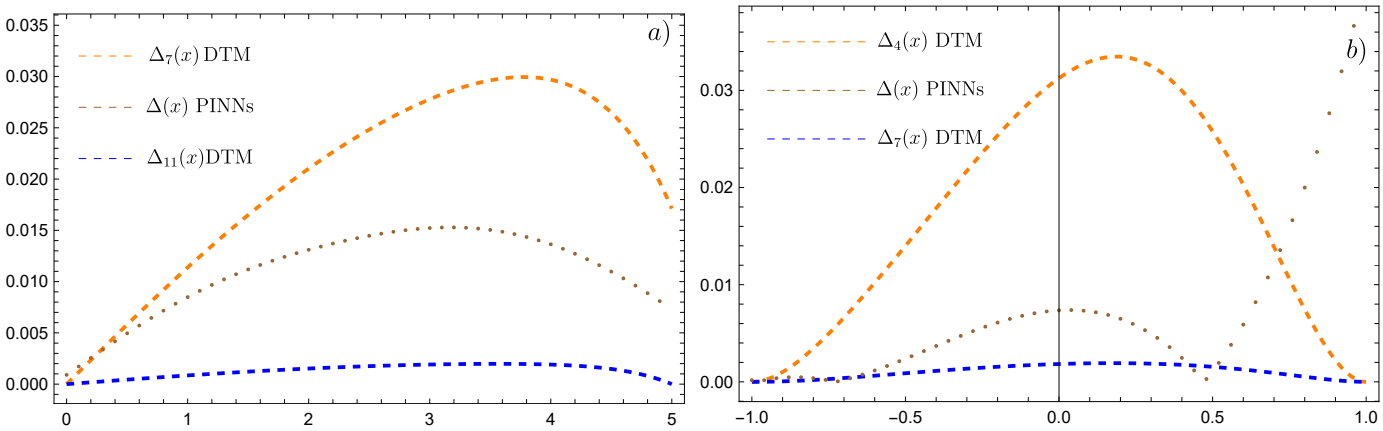

Figure 7. Comparison of absolute errors of the DTM and PINNs for example 2—figure (**a**) and for example 3—figure (**b**).

5. Conclusions

The paper presents two different methods for solving variational calculus problems involving differential equations.

The first method, the DTM, is characterized by its flexibility in both the form of the differential equation and the boundary conditions. One of its strengths is its scalability to handle approximate solutions of varying orders, and sometimes it even allows for a prediction of the exact solution based on the form of the coefficients found for $Y(k)$. However, there are some drawbacks to this method. One notable challenge is the difficulty in automating the process of solving a given problem. It is complex to develop a program that can generate an approximate solution solely based on the form of the differential equation and the boundary conditions. It is often easier to manually construct a discrete counterpart of the differential equation, develop a methodology for finding the unknown coefficients, $Y(k)$, and then utilize appropriate software tools for further refinement.

The second presented approach—PINNs (physics-informed neural networks)—utilizes neural networks to solve the relevant differential equation. One advantage of this method is its relatively straightforward implementation and flexibility. Once the model for the differential equation is trained, it can provide solutions for different grids (input points) without recalculating the problem each time.

Comparing both methods, it can be observed that slightly more accurate results were obtained using the DTM. The results from PINNs includes slightly larger errors compared with those obtained using the DTM, but they are still of good quality. The advantage of neural networks lies in their flexibility and relatively straightforward implementation. Both methods presented in the article are suitable for solving the considered issues.

In the future, the authors plan to apply PINNs to solve direct and inverse problems in partial differential equations involving fractional-order derivatives.

Author Contributions: Conceptualization, R.B. and M.P.; methodology, R.B. and M.P.; software, R.B. and M.P.; validation, R.B. and M.P.; formal analysis, R.B. and M.P.; investigation, R.B. and M.P.; writing—original draft preparation, R.B. and M.P.; writing—review and editing, R.B. and M.P.; visualization, R.B. and M.P.; supervision, R.B. and M.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The author declare no conflict of interest.

References

1. Struwe, M. *Variational Methods*; Springer: New York, NY, USA, 2000; pp. 32–58.
2. Borghi, R. The variational method in quantum mechanics: An elementary introduction. *Eur. J. Phys.* **2018**, *39*, 035410. [CrossRef]
3. Esteban, M.; Lewin, M.; Séré, E. Variational methods in relativistic quantum mechanics. *Bull. Am. Math. Soc.* **2008**, *45*, 535–593. [CrossRef]
4. Mihlin, S.G. *Variational Methods of Solving Linear and Nonlinear Boundary Value Problems. Differential Equations and Their Applications*; Publishing House of the Czechoslovak Academy of Sciences: Prague, Czech Republic, 1963; pp. 77–92. Available online: http://eudml.org/doc/220899 (accessed on 7 July 2024).
5. Sysoev, D.V.; Sysoeva, A.A.; Sazonova, S.A.; Zvyagintseva, A.V.; Mozgovoj, N.V. Variational methods in relativistic quantum mechanics. In *IOP Conference Series: Materials Science and Engineering*; IOP Publishing: Bristol, UK, 2021; Volume 1047, p. 012195.
6. Courant, R.; Hilbert, D. *Methods of Mathematical Physics*; Wiley-VCH: New York, NY, USA, 1962; Volume I.
7. Fox, C. *An Introduction to the Calculus of Variations*; Courier Corporation: North Chelmsford, MA, USA, 1987.
8. Grzymkowski, R.; Pleszczynski, M.; Słota, D. Comparing the Adomian decomposition method and the Runge–Kutta method for solutions of the Stefan problem. *Int. J. Comput. Math.* **2006**, *83*, 409–417. [CrossRef]
9. Grzymkowski, R.; Pleszczynski, M.; Słota, D. The two-phase Stefan problem solved by the Adomian decomposition method. In Proceedings of the 15th IASTED International Conference Applied Simulation and Modelling, Rhodos, Greece, 26–28 June 2006; pp. 511–516.
10. Zhou, J.K. *Differential Transformation and Its Applications for Electrical Circuits*; Huazhong University Press: Wuhan, China, 1986.
11. Ayaz, F. Solutions of the system of differential equations by differential transform method. *Appl. Math. Comput.* **2004**, *147*, 547–567. [CrossRef]
12. Grzymkowski, R.; Pleszczynski, M. Application of the Taylor transformation to the systems of ordinary differential equations. In *Information and Software Technologies, Proceedings of the 24th International Conference, ICIST 2018, Vilnius, Lithuania, 4–6 October 2018*; Springer International Publishing: Berlin/Heidelberg, Germany, 2018; Volume 24, pp. 379–387.
13. Hetmaniok, E.; Pleszczyński, M. Comparison of the selected methods used for solving the ordinary differential equations and their systems. *Mathematics* **2022**, *10*, 306. [CrossRef]
14. Keskin, Y.; Oturanc, G. Reduced differential transform method for partial differential equations. *Int. J. Nonlinear Sci. Numer. Simul.* **2009**, *10*, 741–750. [CrossRef]
15. Mirzaee, F. Differential transform method for solving linear and nonlinear systems of ordinary differential equations. *Appl. Math. Sci.* **2011**, *5*, 3465–3472.
16. Ayaz, F. On the two-dimensional differential transform method. *Appl. Math. Comput.* **2003**, *143*, 361–374. [CrossRef]
17. Kanth, A.R.; Aruna, K. Differential transform method for solving linear and non-linear systems of partial differential equations. *Phys. Lett. A* **2008**, *372*, 6896–6898. [CrossRef]
18. Ayaz, F. Applications of differential transform method to differential-algebraic equations. *Appl. Math. Comput.* **2004**, *152*, 649–657. [CrossRef]
19. Biazar, J.; Eslami, M.; Islam, M.R. Differential transform method for special systems of integral equations. *J. King Saud Univ.-Sci.* **2012**, *24*, 211–214. [CrossRef]
20. Celik, E.; Tabatabaei, K. Solving a Class of Volterra Integral Equation Systems by the Differential Transform Method. *Int. J. Nonlinear Sci. Numer. Simul.* **2013**, *16*, 87–91.
21. Liu, B.; Zhou, X.; Du, Q. Differential transform method for some delay differential equations. *Appl. Math.* **2015**, *6*, 585. [CrossRef]
22. Hetmaniok, E.; Pleszczyński, M.; Khan, Y. Solving the Integral Differential Equations with Delayed Argument by Using the DTM Method. *Sensors* **2022**, *22*, 4124. [CrossRef] [PubMed]
23. Allahviranloo, T.; Kiani, N.A.; Motamedi, N. Solving fuzzy differential equations by differential transformation method. *Inf. Sci.* **2009**, *179*, 956–966. [CrossRef]

24. Osman, M.; Almahi, A.; Omer, O.A.; Mustafa, A.M.; Altaie, S.A. Approximation Solution for Fuzzy Fractional-Order Partial Differential Equations. *Fractal Fract.* **2022**, *6*, 646. [CrossRef]
25. Arikoglu, A.; Ozkol, I. Solution of fractional differential equations by using differential transform method. *Chaos Solitons Fractals* **2007**, *34*, 1473–1481. [CrossRef]
26. Neelma; Eiman; Shah, K. Analytical and Qualitative Study of Some Families of FODEs via Differential Transform Method. *Foundations* **2022**, *2*, 6–19. [CrossRef]
27. Odibat, Z.; Momani, S.; Erturk, V.S. Generalized differential transform method: Application to differential equations of fractional order. *Appl. Math. Comput.* **2008**, *197*, 467–477. [CrossRef]
28. Rysak, A.; Gregorczyk, M. Differential Transform Method as an Effective Tool for Investigating Fractional Dynamical Systems. *Appl. Sci.* **2021**, *11*, 6955. [CrossRef]
29. Kanth, A.R.; Aruna, K. Differential transform method for solving the linear and nonlinear Klein–Gordon equation. *Comput. Phys. Commun.* **2009**, *180*, 708–711. [CrossRef]
30. Kanth, A.R.; Aruna, K. Two-dimensional differential transform method for solving linear and non-linear Schrödinger equations. *Chaos Solitons Fractals* **2009**, *41*, 2277–2281. [CrossRef]
31. Tari, A. The Differential Transform Method for solving the model describing biological species living together. *Iran. J. Math. Sci. Inform.* **2012**, *7*, 63–74.
32. Gupta, R.; Selvam, J.; Vajravelu, A.; Nagapan, S. Analysis of a Squeezing Flow of a Casson Nanofluid between Two Parallel Disks in the Presence of a Variable Magnetic Field. *Symmetry* **2023**, *15*, 120. [CrossRef]
33. Kumar, R.S.V.; Sarris, I.E.; Sowmya, G.; Abdulrahman, A. Iterative Solutions for the Nonlinear Heat Transfer Equation of a Convective-Radiative Annular Fin with Power Law Temperature-Dependent Thermal Properties. *Symmetry* **2023**, *15*, 1204. [CrossRef]
34. Zhang, L.; Han, M.; Zhang, Q.; Hao, S.; Zhen, J. Analysis of Dynamic Characteristics of Attached High Rise Risers. *Appl. Sci.* **2023**, *13*, 8767. [CrossRef]
35. Demir, Ö. Differential Transform Method for Axisymmetric Vibration Analysis of Circular Sandwich Plates with Viscoelastic Core. *Symmetry* **2022**, *14*, 852. [CrossRef]
36. Brociek, R.; Pleszczyński, M. Comparison of Selected Numerical Methods for Solving Integro-Differential Equations with the Cauchy Kernel. *Symmetry* **2024**, *16*, 233. [CrossRef]
37. Raissi, M; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
38. Karniadakis, G.E.; Kevrekidis, I.G.; Lu, L.; Perdikaris, P.; Wang, S.; Yang, L. Physics-informed machine learning. *Nat. Rev. Phys.* **2021**, *3*, 422–440. [CrossRef]
39. Eivazi, H.; Wang, Y.; Vinuesa, R. Physics-informed deep-learning applications to experimental fluid mechanics. *Meas. Sci. Technol.* **2024**, *35*, 075303. [CrossRef]
40. Cuomo, S.; Schiano Di Cola, V.; Giampaolo, F.; Rozza, G.; Raissi, M.; Piccialli, F. Scientific Machine Learning Through Physics–Informed Neural Networks: Where we are and What's Next. *J. Sci. Comput.* **2022**, *92*, 88. [CrossRef]
41. Ahmadi Daryakenari, N.; De Florio, M.; Shukla, K.; Karniadakis, G.E. AI-Aristotle: A physics-informed framework for systems biology gray-box identification. *PLoS Comput. Biol.* **2024**, *20* , 1–33. [CrossRef] [PubMed]
42. Hastings, C.; Mischo, K.; Morrison, M. *Hands-on Start to Wolfram Mathematica and Programming with the Wolfram Language*, 3rd ed.; Wolfram Media, Inc.: Champaign, IL, USA, 2020.
43. Wolfram, S. *The Mathematica Book*, 5th ed.; Wolfram Media, Inc.: Champaign, IL, USA, 2003.