DOI:10.12921/cmst.2024.0000019

Solving the Nonlinear PDE Problems Numerically Using the Fast Fourier Transform

M.R. Dudek

University of Zielona Góra
Institute of Physics
ul. Szafrana 4a, 65-069 Zielona Góra, Poland
E-mail: M.Dudek@if.uz.zgora.pl

Received: 17 November 2024; accepted: 25 November 2024; published online: 7 December 2024

Abstract: A novel, highly efficient numerical method for solving nonlinear partial differential equations (PDEs) is proposed. This method involves a symbolic polynomial of a given degree N defined by a formal variable λ. The sole purpose of λ is to collect expressions that approximate the exact solution of the PDE in terms of the coefficients of this polynomial. It is a formal parameter which is set to 1 afterwards. What is crucial to the method is that the partial derivatives contained in these coefficients of the polynomial are computed using the Fast Fourier Transform. The method can be seen as a complement to finite difference methods. It can be applied to a variety of PDE problems, including those with higher order PDEs. The robustness of the simple iterative scheme that converts the solution of PDEs to the polynomial problem makes this method also promising for engineering applications.

Key words: non-linear PDE, Fast Fourier Transform, KdV equation, Fisher equation, inviscid Burgers equation

I. Introduction

There is no universal numerical method for solving nonlinear partial differential equations (PDEs). The PDEs associated with nonlinear wave propagation, nonlinear diffusion, heat flow, some chemical reaction-diffusion problems, etc., must be solved by using methods based on highly specialized algorithms that cannot be applied to a wide number of physical phenomena. Otherwise, they may lose the numerical stability, and the numerical solutions rapidly diverge from the exact ones. This becomes a major obstacle from the perspective of engineering applications, which typically involve several different physical phenomena. One cause of the possible numerical instability is a truncation error, which results from the truncation of the infinite Taylor series expansion used to construct the calculation algorithm. This error is independent of floating-point precision, and it is very difficult to control for nonlinear PDE problems. The divergence of the numerical solution from the exact one due to truncation error can take different forms, e.g. by the appearance of stable ghost fixed points within the numerical stability region [1]. Another effect of the nonlinearity in PDEs can be numerically induced spatial and temporal chaos in the computed solutions as has been shown for the sine-Gordon equation [2–4]. Thus, different numerical methods can be expected to contribute different types of numerical artifacts. In this paper, the possibility of implementing a simple iterative method that reduces finding the numerical solution of a PDE to an algebraic problem is shown. This method can be considered as an alternative to finite difference methods and can always serve as an additional test of the correctness of numerical solutions for other methods. Therefore, we will only refer to some numerical methods for nonlinear PDEs that are thematically close to the presented method. We also note that for many nonlinear problems, various transformations of the original PDE into another form can be applied, e.g., where the transformed PDEs become easier to compute and allow the use of general numerical techniques. Examples are the Bäcklund transformation [5], Hirota's bilinear method [6], Lax pairs [5], etc. Noteworthy for engineering applications is an interesting algorithm based on sparse optimization reconstruction of partial differential equations [7]. For the purposes of this work, a whole group of methods of homotopy analysis [8] should be mentioned. One example is the Adomian decomposition method [9]. The decomposition method has been successfully applied, for example, to the cubic nonlinear Schrödinger equation [10], the Korteweg-de Vries (KdV) equation [11, 12], and the sine-Gordon equation [13]. Other examples of homotopy analysis can be found, e.g., applied to modeling bifurcation of nonlinear problems [14], modeling nonlinear population dynamics [15], or Fisher's reaction-diffusion equation [16]. The great success of these methods is due to the easy access to various types of numerical packages that allow symbolic computations. The weakness of the homotopy methods in the symbolic algebra implementation is that the coefficients of higher degree symbolic polynomials or other complex analytic expressions can consist of thousands of terms. The latter leads to rapid consumption of available computer memory in the case of iterative algorithms. At the end of this short commentary on the selected computational approaches for nonlinear PDEs, it should be added that in engineering applications the finite element method is often used and it is an interesting complement to questions related to PDE problems, e.g. [17]. The recent engineering of the new functional materials becomes another challenge for the development of new methods for solving nonlinear PDEs. One example can be the experimental data [18, 19] on the propagation of topological solitons in mechanical metamaterials with potential applications for so-called soft robotics. Other examples can be the possibility of exploiting the dynamic solitons in liquid crystals to become the vehicles for 2D delivery of micro-cargos [20] or spatio-temporal solitons [21]. These examples suggest that the theoretical modeling of the related nonlinear phenomena should take into account the effects related both to the discreteness of the samples, their complicated topology, and the possibility of the appearance of phase transitions. All this justifies further searches for a new type of numerical algorithm for solving nonlinear PDEs which, in addition, is expected to be easy to use in engineering implementations. Also noteworthy are recent papers such as the paper [22] on numerical techniques for the nonlinear Burgers-Fisher equation, the paper [24] on the modular KdV technique, or the paper [23] on the analysis of conservation laws in nonlinear partial differential equations.

In this paper, a novel numerical algorithm is developed for solving PDEs related to time evolution problems such as the propagation of the nonlinear waves and the nonlinear diffusion. The PDE examples given below are limited to one-dimensional systems, but extending the method to more dimensions is straightforward. The algorithm uses the numerical implementation of the homotopy concept to collect the numerical expressions representing the approximate solution of the PDE, while the Fast Fourier Transform (FFT) is used to numerically compute the partial derivatives. At this point, one should note the interesting course on the numerical solution of the KdV equation by Prof. Mike Meylan [25], who gave the idea of using alternatively FFT and Inverse Fast Fourier Transform (IFFT) for the numerical solution of KdV using the split method. Another example of the hybrid method combining FFT and differential evolution can be found in the paper [26] on antenna application.

II. Concept of Symbolic Polynomials

The numerical algorithm for solving nonlinear PDEs discussed in this paper uses the concept of symbolic polynomials published in the paper [27] for ordinary differential equations (ODEs). This concept corresponds to the homotopy perturbation approach [8] but it can also be compared to the well-known Sommerfeld method [28] used by physicists, e.g. to solve the Schrödinger equation for the hydrogen atom. These two approaches can be reduced to the same problem of finding an approximate solution $u(t)$ for a given initial condition $u(t_0) = u_0$ of the following differential equation (the same holds to the higher order differential equations):

$$\frac{du(t)}{dt} = \lambda f(u(t)), \tag{1}$$

where λ is a formal parameter which is set to 1 afterwards and $u(t)$ is approximated by a symbolic polynomial $\mathcal{P}_N$ of degree N in λ as follows:

$$\mathcal{P}_N(t_0, \tau, \lambda) = u_0 + \lambda u_1(\tau) + \lambda^2 u_2(\tau) + \ldots + \lambda^N u_N(\tau), \tag{2}$$

where $\tau = t - t_0$ and $u_k(\tau)$ $(k = 1, 2, \ldots, N)$ are unknown functions with the property that $u_k(0) = 0$.

If $f(u(t))$ in Eq. (1) is not a polynomial in $u(t)$, it should be approximated by a Taylor expansion in a neighborhood at the point u_0 of the order of N (assuming that $f(u(t))$ is sufficiently regular), which takes the following form:

$$f(u(t)) \approx f(u_0) + \Sigma_{j=1}^{N} a_j (u(t) - u_0)^j =: f_N(u(t)) \,. \tag{3}$$

After substituting $\mathcal{P}_N$ for $u(t)$ in Eq. (1), and equating the coefficients of terms with the same power of λ on both sides of Eq. (1) to each other, the following set of N differential equations is obtained:

$$\frac{du_1(\tau)}{d\tau} = f(u_0), \tag{4}$$

$$\frac{du_2(\tau)}{d\tau} = a_1 u_1(\tau), \tag{5}$$

$$\frac{du_3(\tau)}{d\tau} = a_1 u_2(\tau) + a_2 u_1(\tau)^2, \tag{6}$$

and so on. These equations can be solved with the help of definite integrals from 0 to τ. In this case, $u_1(\tau) = f(u_0)\tau$, $u_2(\tau) = a_1 f(u_0)\tau^2/2$, $u_3(\tau) = (a_1^2 f(u_0) + a_2 f(u_0)^2)\tau^3/6$, etc. The analogous iterative scheme appears in the more general case when f depends on time t both directly and indirectly, i.e., when $f = f(u(t), t)$. Then, $u_1(\tau), \ldots, u_N(\tau)$ will generally have more complex functional form than simple powers of τ with the same value as powers of λ, but still they can be easily integrated as in the case of the forced Duffing oscillator [27].

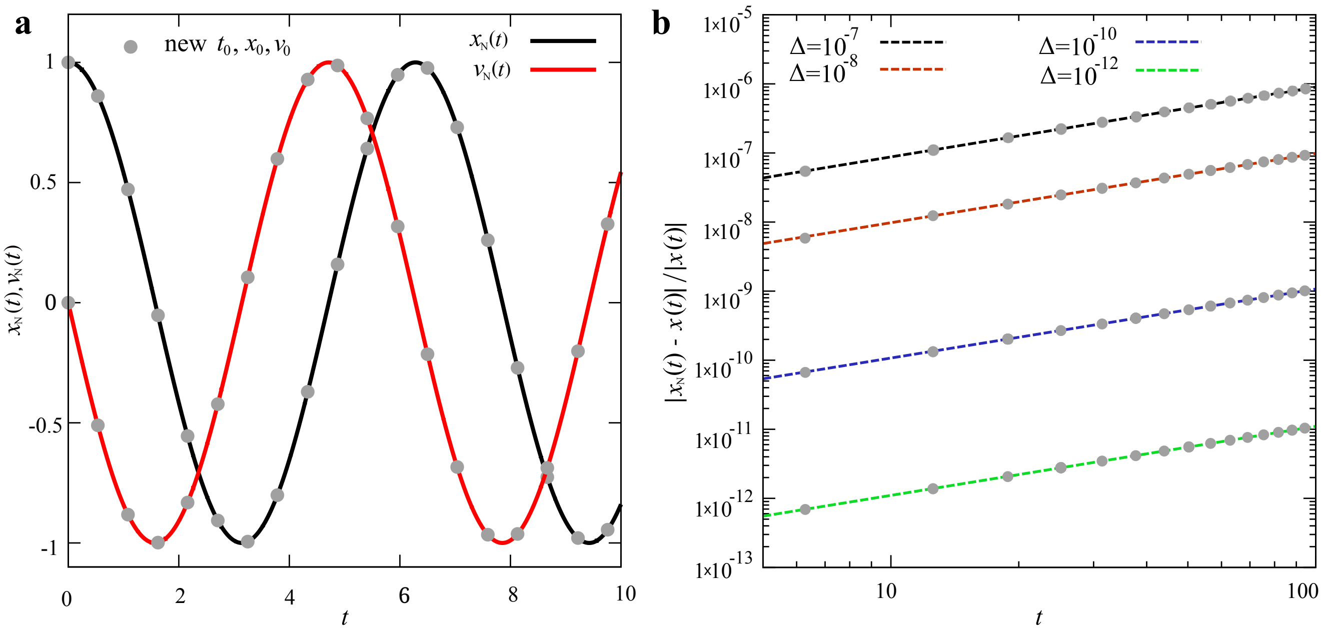

Fig. 1. Polynomial approximation of the solutions of Eq. (8) and Eq. (9) for $\lambda = 1$: a) plots of $x_N(t)$ and $v_N(t)$ when $\Delta = 10^{-10}$ and $N = 9$, b) dependence of relative absolute error on time for different values of $\Delta = 10^{-7}, 10^{-8}, 10^{-10}, 10^{-12}$. Error values were calculated for those values of t that satisfied the condition $|x(t) - 1| < 0.000015$ for the exact solution $x(t)$. The relative absolute error observed for $v_N(t)$ shows analogous behavior

The expression, $u_N(t) = \mathcal{P}_N(t_0, \tau, 1)$, represents a good approximation of the exact solution of Eq. (1) on an interval of $t \in [t_0, t_0 + \delta]$ such that

$$\left| \frac{du_N(t)}{d\tau} - f(u_N(t)) \right| < \Delta, \tag{7}$$

where $\Delta > 0$ is the given accuracy, and $t = t_0 + \tau$. The introduced method of symbolic polynomials does work in such a way that when this condition is no longer satisfied, the polynomial $u_N(t)$ with the new initial condition u_0 at new initial time $t_0 = t_0 + \delta$ is used [27], and the procedure is repeated. In the applied numerical algorithm, the parameter δ serves as a trial value. It adapts its value to the set Δ accuracy of the numerical solution, and a smaller value is chosen if the accuracy criterion is not met. In the numerical examples, δ is divided by 10 an appropriate number of times. The number n_{div} of divisions of δ is limited to $n_{\text{div}} = 6$.

In this method the number of round-off errors and the cumulative effect of the truncation error increase in proportion to the number of updates of the initial condition u_0. In contrast to the finite difference algorithms, the higher the degree N of the polynomial $\mathcal{P}_N$ approximating the exact solution, the less frequent the u_0 updates and the slower the numerical error cumulation. Successive updates of the initial conditions to new t_0 and u_0 values increase the deviation of $u_N(t)$ from the exact value but this type of numerical instability can be controlled by setting a sufficiently small value of Δ. This is very well illustrated by the following simple example of a Newtonian equation of motion for a harmonic oscillator with a unit mass and unit spring constant. Here the equation is written in the form of two linear differential equations, one for displacement x and one for velocity v, as follows:

$$\frac{dx}{dt} = \lambda v, \tag{8}$$

and

$$\frac{dv}{dt} = -\lambda x, \tag{9}$$

where we introduced the formal parameter $\lambda = 1$. If $x_0 = 1$ and $v_0 = 0$ are the initial conditions, the exact solution is represented by:

$$x(t) = cos(t)\,, v(t) = -sin(t). \tag{10}$$

Assuming that the degree of symbolic polynomials $\mathcal{P}_x$ and $\mathcal{P}_v$ approximating the solutions of Eq. (8) and Eq. (9) is equal to $N = 9$, then in this particular case, $\mathcal{P}_x$ and $\mathcal{P}_v$ take the following form:

$$\mathcal{P}_x(t_0, \tau, \lambda) = x_0 + x_1(\tau)\lambda + x_2(\tau)\lambda^2 + \ldots + x_9(\tau)\lambda^9\,, \tag{11}$$

and

$$\mathcal{P}_v(t_0, \tau, \lambda) = v_0 + v_1(\tau)\lambda + v_2(\tau)\lambda^2 + \ldots + v_9(\tau)\lambda^9\,. \tag{12}$$

After calculating the coefficients of $\mathcal{P}_x(t_0, \tau, \lambda)$ and $\mathcal{P}_v(t_0, \tau, \lambda)$ by substituting these polynomials for x and v in Eq. (8) and Eq. (9) , the approximate solutions $x_N(t) = \mathcal{P}_x(t_0, \tau, 1)$ and $v_N(t) = \mathcal{P}_v(t_0, \tau, 1)$ at $t = t_0 + \tau$ read as follows:

$$x_N(t_0+\tau)=x_0\left(1-\frac{\tau^2}{2}+\frac{\tau^4}{24}-\frac{\tau^6}{720}+\frac{\tau^8}{40320}\right)+ \\ +v_0\left(\tau-\frac{\tau^3}{6}+\frac{\tau^5}{120}-\frac{\tau^7}{5040}+\frac{\tau^9}{362880}\right), \tag{13}$$

$$v_N(t_0+\tau)=v_0\left(1-\frac{\tau^2}{2}+\frac{\tau^4}{24}-\frac{\tau^6}{720}+\frac{\tau^8}{40320}\right)+ \\ +x_0\left(-\tau+\frac{\tau^3}{6}-\frac{\tau^5}{120}+\frac{\tau^7}{5040}-\frac{\tau^9}{362880}\right). \tag{14}$$

They are a good approximation of the exact solution of Eq. (8) and Eq. (9) up to a given value of Δ on the interval of $t \in [t_0, t_0+\delta]$ such that

$$\left|\frac{dx_N(t_0+\tau)}{d\tau}-v_N(t_0+\tau)\right|<\Delta \ \wedge \\ \wedge \ \left|\frac{dv_N(t_0+\tau)}{d\tau}+x_N(t_0+\tau)\right|<\Delta. \tag{15}$$

If for some value of τ this condition is not satisfied, the polynomials $x_N(\tau)$ and $v_N(\tau)$ with the new initial condition x_0 and v_0 at new initial time $t_0 = t_0+\delta$ are used, and the procedure is repeated. In Fig.1, the symbols represented by the filled circles in panel (a) indicate values $x_N(t_0)$ representing a new initial condition with a new value of t_0 for a given value of $\Delta = 10^{-10}$, while the plots in panel (b) show the cumulative effect of the relative absolute error for the deviation of the approximate solution $x_N(t)$ from the exact $x(t)$ for the values of t for which $|x(t)-1| < 0.000015$. Regression analysis of the function $f(t) = at^b$ used for fitting suggests a linear dependence of the relative absolute error on t. In Fig.1, for t in the interval $[0, 100]$, the values of the parameter b are 1.01077 ± 0.001315 ($\Delta = 10^{-7}$), 1.00298 ± 0.002424 ($\Delta = 10^{-8}$), 0.998894 ± 0.0003931 ($\Delta = 10^{-10}$), and 0.999293 ± 0.0003336 ($\Delta = 10^{-12}$). Although the presented method is not numerically stable in the sense of numerical stability of finite difference methods, the deviation of the approximate solution from the exact one can be controlled.

III. Numerical Algorithm for Non-linear PDEs

III. 1. Description of the Method

A hybrid numerical algorithm for solving nonlinear PDEs is introduced, using symbolic polynomials to determine the time dependence of their solution $u(x,t)$ and the FFT method to compute partial derivatives of $u(x,t)$ with respect to the variable x. For simplicity, we will consider the Korteweg-de Vries equation (KdV) [29, 30] which is an example of a third-order PDE. Generalization, e.g. to PDEs with the higher order partial derivatives, or the higher space dimension, does not require any change in the algorithm. If the functional form of the nonlinear terms in the PDE does not represent polynomials, a Taylor expansion of these terms in a neighborhood of $u_0(x)$ should be performed as it was suggested in Eq. (3).

Assume the KdV equation to be in the following form:

$$\frac{\partial u(x,t)}{\partial t} = -6u(x,t)\frac{\partial u(x,t)}{\partial x} - \frac{\partial^3 u(x,t)}{\partial^3 x}\,, \tag{16}$$

with the initial condition

$$u_0(x) = \frac{1}{2}\mathrm{sech}^2(x/2). \tag{17}$$

In this case, the exact solution of KdV reads as the following:

$$u(x,t) = \frac{1}{2}\mathrm{sech}^2((x-t)/2), \tag{18}$$

which is a 1D soliton of height 1/2 moving to the right with velocity 1. In analogy to the previous section, we will introduce the formal parameter λ on the right-hand side of Eq. (16). We also rewrite the term with the first order partial derivative in a different form so that the resulting differential equation reads as follows:

$$\frac{\partial u(x,t)}{\partial t} = -\lambda\left(3\frac{\partial u^2(x,t)}{\partial x} + \frac{\partial^3 u(x,t)}{\partial^3 x}\right). \tag{19}$$

In the numerical implementation of our algorithm, space variable x takes $\mathcal{N}$ equally spaced values $x_0, x_0 + +\Delta x, \ldots, x_0 + (\mathcal{N}-1)\Delta x$, and to skip the problem of numerically computing partial derivatives, we use the Fast Fourier Transform to convert this equation to the following:

$$\frac{\partial \hat{u}(k,t)}{\partial t} = \lambda\left(-3ik\widehat{u^2}(k,t) + ik^3\hat{u}(k,t)\right), \tag{20}$$

where k represents wave vector, $\hat{u} = \mathrm{FFT}(u)$, and $\widehat{u^2} = = \mathrm{FFT}(u^2)$. The corresponding $\mathcal{N}$ values of k with the spacing $\Delta k = 2\pi/(\mathcal{N}\Delta x)$ take the values arranged as follows: $0, \Delta k, \ldots, (\mathcal{N}/2-1)\Delta k, -(\mathcal{N}/2)\Delta k, \ldots, -\Delta k$. We choose $\mathcal{N}$ to be even and this arrangement of the values of k is consistent with the FFT implementation in the NumPy libraries in Python.

The symbolic polynomial approximation, which represents the solution of Eq. (19) for $\lambda = 1$, is defined as:

$$\mathcal{P}_N(t_0,\tau,x,\lambda) = u_0(x) + \lambda u_1(x,\tau) + \lambda^2 u_2(x,\tau)+ \\ +\ldots+\lambda^N u_N(x,\tau), \tag{21}$$

and its Fast Fourier Transform which represents this solution in k-space takes the following form:

$$\hat{\mathcal{P}}_N(t_0,\tau,k,\lambda) = \hat{u}_0(k) + \lambda\hat{u}_1(k,\tau) + \lambda^2\hat{u}_2(k,\tau)+ \\ +\ldots+\lambda^N\hat{u}_N(k,\tau). \tag{22}$$

The unknown coefficients in these polynomials can be determined by analogy to the iterative scheme introduced in the previous section, by substituting $\mathcal{P}_N(t_0,\tau,x,\lambda)$ for $u(x,t)$ in Eq. (19) and $\hat{\mathcal{P}}_N(t_0,\tau,k,\lambda)$ for $\hat{u}(k,t)$ in Eq. (20), and equating the terms with the same power of λ on both sides of these equations. The main difference is that to skip the problem of computing the convolution in k-space these coefficients are determined alternately in x-space and k-space. In the special case of $N = 7$ the iterative scheme yields seven differential equations, which take the following form in k-space:

$$\frac{d\hat{u}_1(k,\tau)}{d\tau} = ik^3\hat{u}_0(k) - 3ik\widehat{u_0^2}(k), \tag{23}$$

$$\frac{d\hat{u}_2(k,\tau)}{d\tau} = ik^3\hat{u}_1(k,\tau) - 6ik\widehat{u_0u_1}(k,\tau), \tag{24}$$

$$\frac{d\hat{u}_3(k,\tau)}{d\tau} = ik^3\hat{u}_2(k,\tau) - 6ik\widehat{u_0u_2}(k,\tau) - 3ik\widehat{u_1^2}(k,\tau), \tag{25}$$

$$\frac{d\hat{u}_4(k,\tau)}{d\tau} = ik^3\hat{u}_3(k,\tau) - 6ik\widehat{u_0u_3}(k,\tau) - 6ik\widehat{u_1u_2}(k,\tau), \tag{26}$$

$$\begin{aligned}\frac{d\hat{u}_5(k,\tau)}{d\tau} &= ik^3\hat{u}_4(k,\tau) - 6ik\widehat{u_0u_4}(k,\tau) - 6ik\widehat{u_1u_3}(k,\tau) + \\ &\quad - 3ik\widehat{u_2^2}(k,\tau),\end{aligned} \tag{27}$$

$$\begin{aligned}\frac{d\hat{u}_6(k,\tau)}{d\tau} &= ik^3\hat{u}_5(k,\tau) - 6ik\widehat{u_0u_5}(k,\tau) - 6ik\widehat{u_1u_4}(k,\tau) + \\ &\quad - 6ik\widehat{u_2u_3}(k,\tau),\end{aligned} \tag{28}$$

$$\begin{aligned}\frac{d\hat{u}_7(k,\tau)}{d\tau} &= ik^3\hat{u}_6(k,\tau) - 6ik\widehat{u_0u_6}(k,\tau) - 6ik\widehat{u_1u_5}(k,\tau) + \\ &\quad - 6ik\widehat{u_2u_4}(k,\tau) - 3ik\widehat{u_3^2}(k,\tau),\end{aligned} \tag{29}$$

where $\widehat{u_0^2} = \mathrm{FFT}(u_0^2)$, $\widehat{u_1^2} = \mathrm{FFT}(u_1^2)$, $\widehat{u_2^2} = \mathrm{FFT}(u_2^2)$, $\widehat{u_3^2} = \mathrm{FFT}(u_3^2)$, $\widehat{u_0u_j} = \mathrm{FFT}(u_0u_j)$ $(j = 1, 2, \ldots, 6)$, $\widehat{u_1u_j} = \mathrm{FFT}(u_1u_j)$ $(j = 1, 2, \ldots, 5)$, $\widehat{u_2u_4} = \mathrm{FFT}(u_2u_4)$, $\widehat{u_2u_4} = \mathrm{FFT}(u_2u_4)$. Note that after $\hat{u}_1$ is calculated, the real part of its Inverse Fast Fourier Transform (IFFT), $u_1 = = \mathrm{re}(\mathrm{IFFT}(\hat{u}_1))$, is required because in the following equation (Eq. (24)) there is a term $\widehat{u_0u_1} = \mathrm{FFT}(u_0u_1)$ where the Fourier Transform is applied to the product u_0u_1 in x-space. We proceed in the same way with the subsequent equations Eqs. (25-29) with $\hat{u}_2, \hat{u}_3, \ldots$, and $\hat{u}_7$.

In the case of the KdV equation, where there is no direct dependence on time t, the set of equations Eqs. (23-29) can be rewritten in a simpler form by separating τ. We will use this option for the sake of clarity in the algorithm presentation. If we introduce the notation that $\hat{s}_1(k)$ denotes the right-hand side of Eq. (23), i.e. $\hat{s}_1(k) = ik^3\hat{u}_0(k) + -3ik\widehat{u_0^2}(k)$, and if $s_1 = \mathrm{re}(\mathrm{IFFT}(\hat{s}_1))$, then $u_1(x,\tau) = = s_1(x)\tau$. In this case, Eq. (24) can be converted to the following:

$$\frac{d\hat{u}_2(k,\tau)}{d\tau} = ik^3\hat{s}_1(k)\tau - 6ik\widehat{s_0s_1}(k)\tau\,, \tag{30}$$

where we have introduced the notation $s_0(x) = u_0(x)$, and $\widehat{s_0s_1} = \mathrm{FFT}(s_0s_1)$. If we define $\hat{s}_2(k) = = ik^3\hat{s}_1(k) - 6ik\widehat{s_0s_1}(k)$, and $s_2 = \mathrm{re}(\mathrm{IFFT}(\hat{s}_2))$, then $u_2(x,\tau) = s_2(x)\tau^2/2$. In an analogous way, we can obtain $u_3(x,\tau) = s_3(x)\tau^3/6$, $u_4(x,\tau) = s_4(x)\tau^4/24$, $u_5(x,\tau) = = s_5(x)\tau^5/120$, $u_6(x,\tau) = s_6(x)\tau^6/720$, and $u_7(x,\tau) = = s_7(x)\tau^7/5040$.

Fig. 2 shows the graphical representation of the polynomial approximation $u_N(x,t) = P(t_0,\tau,x,1)$ of the KdV equation (Eq. (16)) when the value of τ is set to 0.15. The sum of eight plots in panel (b) is shown in panel (a) as the $u_N(x,\tau)$.

By analogy with the method of symbolic polynomials introduced in the previous section, the polynomial $u_N(x,t_0,\tau)$ represents a good approximation of the exact solution $u(x,t_0,\tau)$ only on a small interval $t \in [t_0, t_0 + \delta]$, such that the condition

$$\left|\frac{\partial u_N(x,t_0,\tau)}{\partial\tau} + 3\frac{\partial u_N^2(x,t_0,\tau)}{\partial x} + \frac{\partial^3 u_N(x,t_0,\tau))}{\partial x^3}\right| < \Delta \tag{31}$$

is fulfilled for a given accuracy Δ for all values of $x \in \{x_0, x_0+\Delta x, \ldots, x_0+(\mathcal{N}-1)\Delta x\}$. In this case, the partial derivatives can be represented by the following expressions:

$$\begin{aligned}\frac{\partial u_N(x,t_0,\tau)}{\partial x} &= s_1(x) + s_2(x)\tau + s_3(x)\tau^2/2 + \\ &\quad + s_4(x)\tau^3/6 + s_5(x)\tau^4/24 + \\ &\quad + s_6(x)\tau^5/120 + s_7(x)\tau^6/720,\end{aligned} \tag{32}$$

$$\begin{aligned}\frac{\partial u_N^2(x,t_0,\tau)}{\partial x} &= \mathrm{re}(\mathrm{IFFT}(ik\widehat{u_N^2}))(x,\tau), \\ \frac{\partial^3 u_N(x,t_0,\tau)}{\partial x^3} &= \mathrm{re}(\mathrm{IFFT}(-ik^3\hat{u}_N)(x,\tau).\end{aligned} \tag{33}$$

When t reaches $t_0+\delta$, a new polynomial $\mathcal{P}_N$ is required with the new initial condition $u_0(x) = u_N(x, t_0+\delta)$ where $t_0+\delta$ becomes new initial time t_0, and the procedure is repeated.

In Fig. 3, the effect of the number of iterations (N) on the updates of the initial condition $u_0(t_0, x)$ is shown for the numerical solution of Eq. (19), where the bin width $\Delta t = 0.05$. The results represent the case of $N = 3, 5$, and 7 iterations used to obtain the numerical approximation of the solution of Eq. (19). The numerical results have the same accuracy of $\Delta = 10^{-6}$ and $n_{\mathrm{div}} = 6$.

The main source of numerical instability in the presented method is the acceptance of the numerical solution at δ not satisfying the criterion Δ. A possible control of such instability is the assignment of a threshold for such accumulation. An alternative is to stop the computation of the solution for times later than when the condition Δ is not satisfied. For a given value Δ of the accuracy of the numerical solution, the smaller the value of N, the greater will be the slope of the curve of accumulation of numerical errors. In the case of nonlinear partial differential equations, the FFT method applied alternately, with successive iterations of the construction of the approximate solution, may lead to the appearance of additional instabilities associated with spectral leakage. This type of numerical instability significantly slows down the operation of the algorithm because, in such a case, the self-adaptation of δ to a given accuracy Δ causes the contribution of expressions of the approximation with a higher value of N to be minimized by decreasing the value of δ.

When PDEs have no known exact solutions, one can use solution analysis methods such as in the paper [31]. In the present work, the aim is to compare the results of the spec-

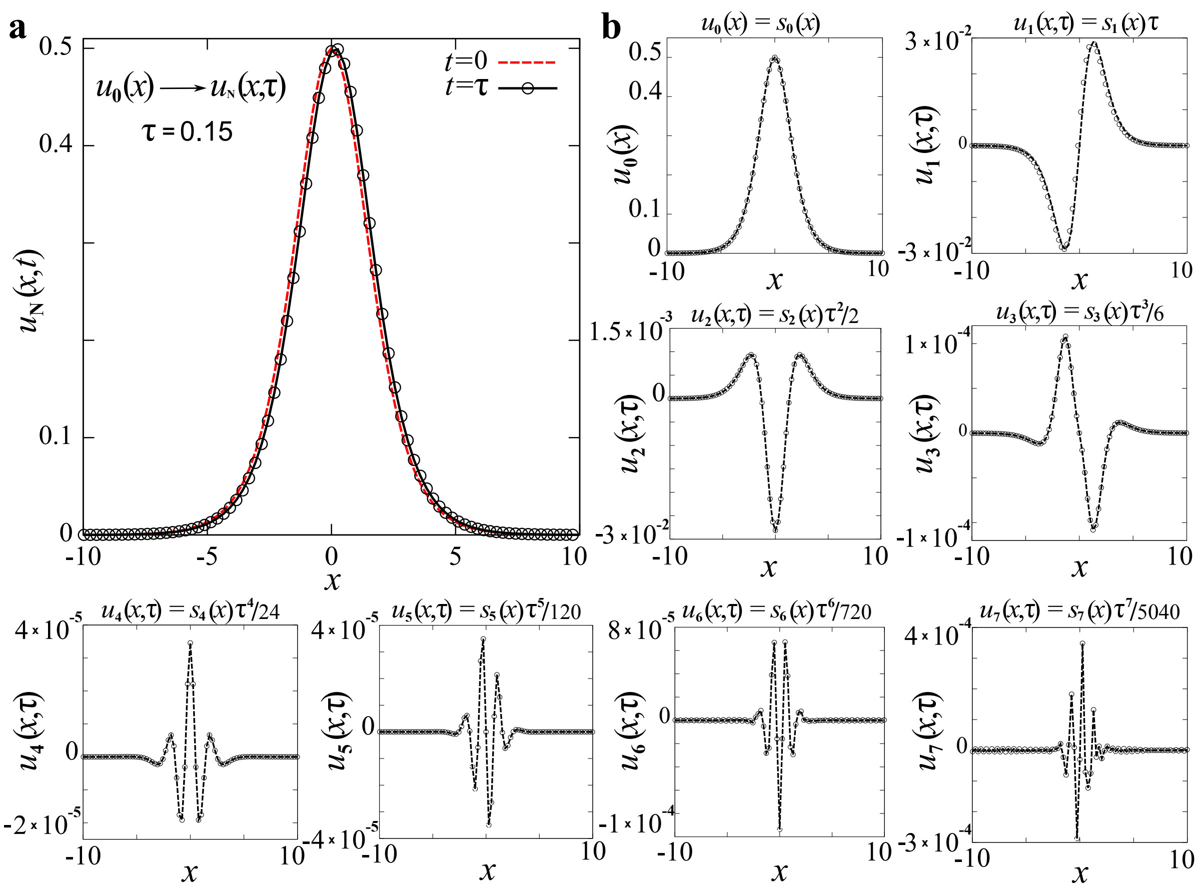

Fig. 2. Polynomial approximation of the solution of KdV equation: a) plots representing soliton at $t_0 = 0$ and its polynomial approximation $u_N(x,t)$ when t is set to $t_0 + \tau$, b) graphical representation of the components of $u_N(x,\tau)$ when $N = 7$

tral method for different Δ accuracies. Most of the numerical calculations in this paper were performed on computers with AMD EPYC 9354 processors (CPU mark: 93.212) – base 3.25 GHz, turbo 3.75 GHz. In the case of the KdV equation in Eq.(16), the CPU expressed in seconds to compute $\mathcal{N} = 2^{12}$ discrete values of $u_N(x,t_0,\tau)$ was equal to 1.52 sec for $N = 7$ and 1.32 sec for $N = 3$.

III. 2. FFT and Spectral Leakage

This subsection has a technical meaning, but it is very important for the successful numerical implementation of the presented method, e.g. under Python. Note that in the presented numerical method FFT is applied to $\mathcal{N}$ discrete values of u_N=$\{u_N(x_0,\tau),\ldots,u_N(x_0+(\mathcal{N}-1)\Delta x,\tau)\}$ and its components. The FFT method assumes a periodic repetition of the transformed data, but if they are not exactly periodic, then some extra residual frequency can be generated resulting in the noisy results. This is what is called spectral leakage [32]. In this case, the filtering techniques such as Hamming window or von Hann window [32, 33], which are often used in signal processing, can be applied to reduce such leakage.Then, the FFT is applied to the product of such window and the data under consideration. In this paper, the von Hann window W_{Hann}=$\{W_{\text{Hann}}(x_0),\ldots,W_{\text{Hann}}(x_0+ +(\mathcal{N}-1)\Delta x)\}$ was used to counteract the possible occurrence of spectral leakage. For example, to compute $\hat{u}_1$, first the product of the von Hann window function $W_{\text{Hann}}(x)$ and $u_1(x)$ was computed for each $x \in \{x_0, x_0 + +\Delta x, \ldots, x_0 + (\mathcal{N}-1)\Delta x\}$, i.e. $\hat{u}_1 = \text{FFT}(W_{\text{Hann}}u_1)$. It should be noted that, in general, noisy data can appear even after applying the application of an appropriate filter such as the von Hann window. This has been shown in Fig. 4 which presents exemplary plots of the term $s_6(x)$ in $u_6(x,\tau)$ for different values of Δx.

It is evident that the proper selection of the value of Δx is important for obtaining the data without the leakage noise. More, some averaging techniques have to be used in particular cases. The higher order of the jth term $u_j(x,\tau)$ in $u_N(x,\tau) = \mathcal{P}_N(t_0,\tau,x,1)$ the higher probability for the noisy FFT results. In particular, in the case of $N = 9$, we selected $\Delta x = 0.2556$ and $\mathcal{N} = 2^{16}$.

The numerical results in this work are based on Python numerical libraries, including mpmath libraries for arbitrary precision calculations. To use arbitrary precision arithmetic for FFT the standard FFT algorithm was modified with the help of mpmath libraries based on the code available here[1] and python package apfft.

[1] https://jakevdp.github.io/blog/2013/08/28/understanding-the-fft/

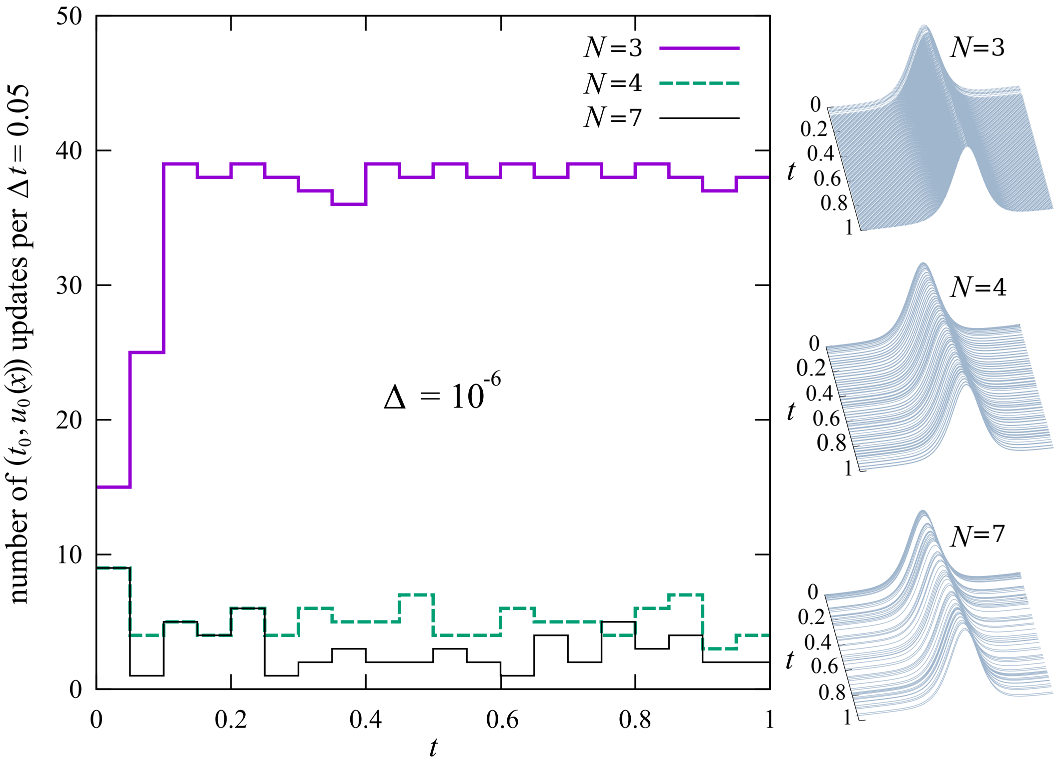

Fig. 3. Histograms of the initial condition updates over time in the case of the numerical solution of Eq. 19 with different numbers of $N = 3, 5$, and 7 iterations used in this solution construction. The parameters used: $\Delta = 10^{-6}$, $n_{\text{div}} = 6$, bin width $\Delta t = 0.05$

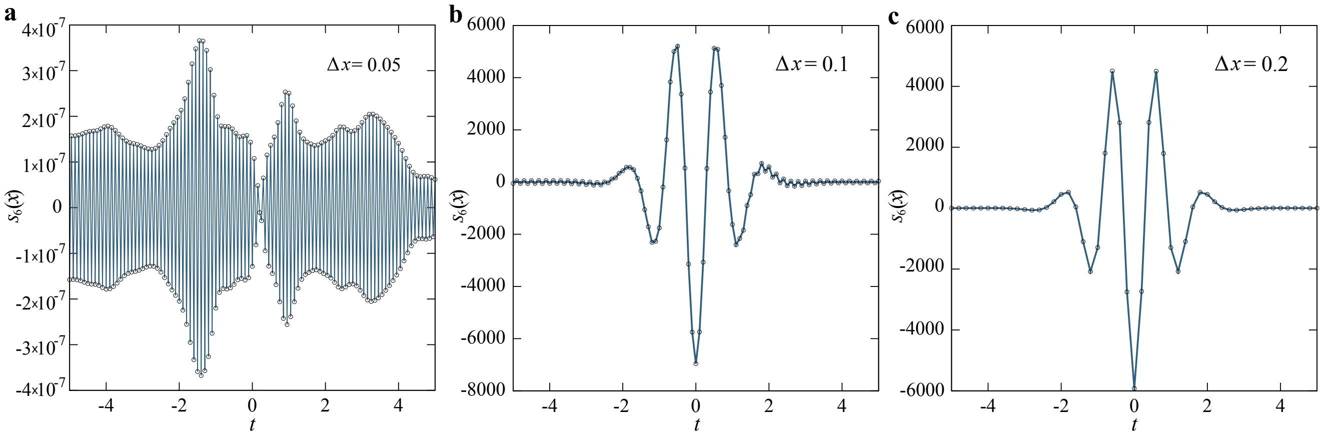

Fig. 4. Effect of the spectral leakage at different values of spacing Δx when $N = 2^{12}$: a) $\Delta x = 0.05$, b) $\Delta x = 0.1$, c) $\Delta x = 0.2$

IV. Results and Discussion

This section presents examples of modeling nonlinear waves using the method outlined above, using the FFT to compute partial derivatives. Numerical error accumulation relationships as a function of time t are discussed.

IV. 1. KdV Equation

Fig. 5 shows the results for the propagating solitary wave which is described with the help of the approximate polynomial solution $u_N(x,t)$ ($N = 9$) of Eq. (16), and where the initial condition $u_0(x)$ is defined in Eq. (17). The plotted range of x, which has been shown in the figure, is the interval $[-10, 25]$ but the actual range of x for performing the FFT is equal to approximately 1047 ($\mathcal{N} = 2^{12}$, $\Delta x = 0.2556$). The profile of the propagating soliton is shown in Fig. 5b. Fig. 5c shows the cumulative effect of the average relative absolute error $\bar{\varepsilon}_{\text{abs}}$, which is defined as follows:

$$\bar{\varepsilon}_{\text{abs}} = \frac{1}{N_{\max}} \sum_{x_{\max}-r}^{x_{\max}+r} |(u_N(x,t) - u(x,t))|/|u(x,t)| \,, \quad (34)$$

where r is the range of x values from the $x_{\max}$ value representing the location of the maximum of the exact $u(x,t)$, and $N_{\max}$ is the number of $u_N(x,t)$ values included in this range. The error plot for the range of times t in the interval [0,15] suggests a quadratic power relationship. The error values exhibit oscillatory properties related both to the displacement of the soliton maximum and to its fit to a discrete set of x values. However, the above estimate allows to determine in a controlled way the accuracy of the used approximation.

As an example of practical mechanical application, Fig. 6 shows the time evolution effect of the solution $u_N(x,t)$ of the equation Eq. (16) for $t \in [0,5]$ when the initial condition is represented by a Gaussian function as follows:

$$u_0(x) = \frac{1}{\sqrt{2\pi}} \mathrm{e}^{-x^2/2} . \tag{35}$$

Such a posted problem is important, for example, for modeling the effect of wind on steep waves [34]. The space-time surface of the solution in the figure is decimated. However, there are still visible fragments where the solutions show denser packing. They correspond to time moments with smaller intervals $[t_0, t_0+\delta]$. This example shows that, depending on the phenomenon to be modeled, it is possible to take into account changes that occur for any value of t.

IV. 2. Fisher Equation

As the second example let us consider the nonlinear Fisher reaction-diffusion equation in a form as in papers [35, 36]:

$$\frac{\partial u(x,t)}{\partial t} = \frac{\partial^2 u(x,t)}{\partial x^2} + u^2(x,t)(1-u(x,t)) , \tag{36}$$

with the initial condition

$$u_0(x) = \frac{1}{1+\mathrm{e}^{\frac{1}{\sqrt{2}}x}} , \tag{37}$$

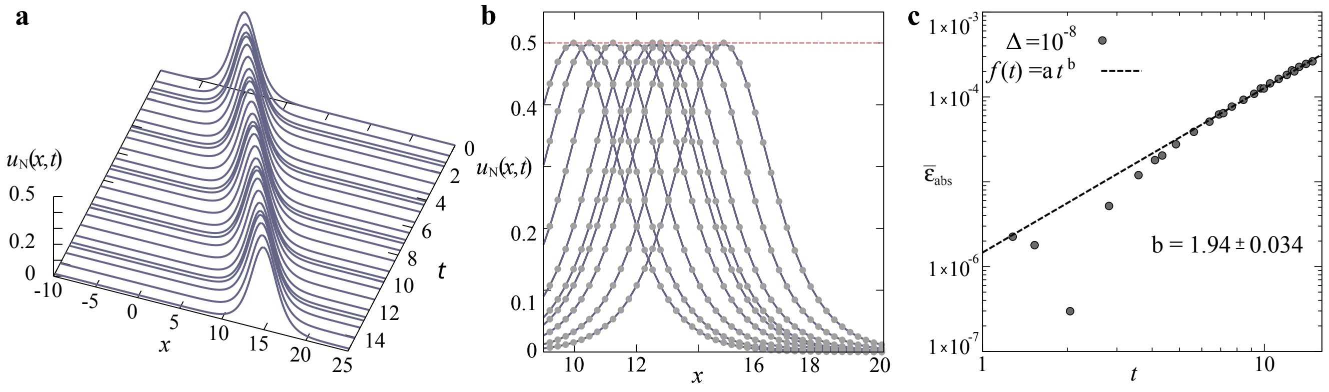

Fig. 5. Polynomial approximation $u_N(x,t)$ $(N=9)$ of the soliton (Eq. (16)) moving in the x-direction with a speed of 1: a) plots of $u_N(x,t)$ for which the value x of the position of the maximum of the corresponding exact $u(x,t)$ satisfies the inequality $|x-t| < 0.0003$, b) profiles $u_N(x,t)$ for the final fragment from panel (a), c) dependence of the mean relative absolute error on time. Parameters: $N=9$, $\Delta x = 0.2556$, tolerance accuracy $\Delta = 10^{-8}$

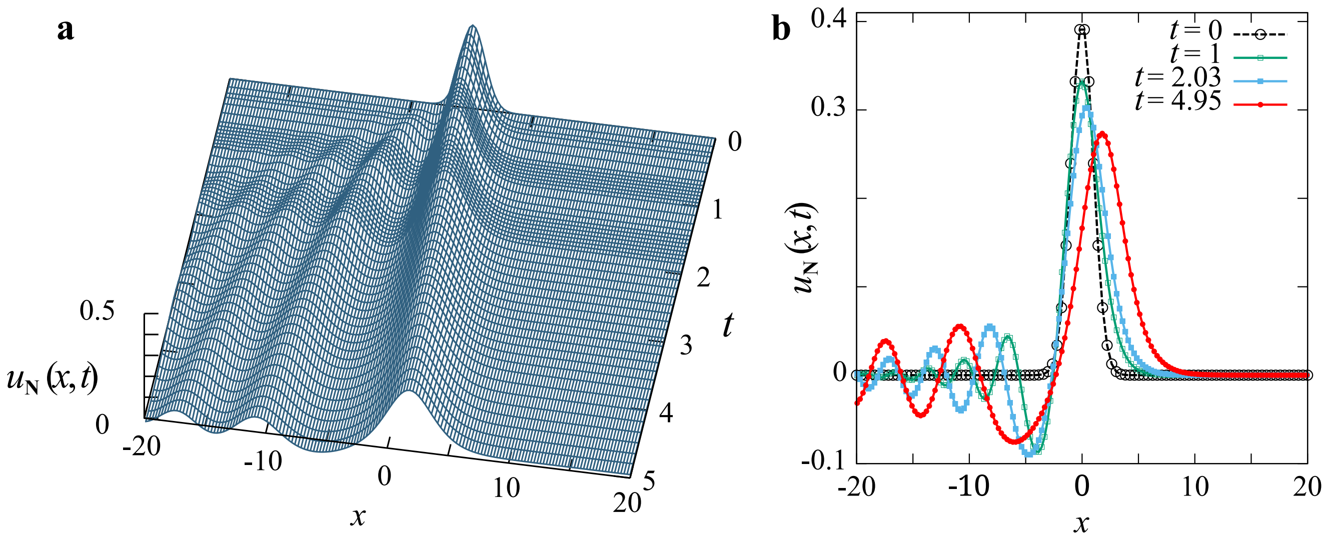

Fig. 6. Evolution of the KdV solution with the Gaussian initial condition: a) space-time surface of the solution, b) selected solutions at different times, $t = 0, 1, 2.03, 4.95$. Other parameters as in Fig. 5

and the exact solution which reads as the following:

$$u(x) = \frac{1}{1 + \mathrm{e}^{\frac{1}{\sqrt{2}}(x - \frac{1}{\sqrt{2}}t)}} \,, \tag{38}$$

where it represents reaction-diffusion wave front propagating with time t along x-direction. If we introduce a formal parameter λ into the right-hand side of Eq. (36) (λ is set to 1 afterwards), and substitute the symbolic polynomial $\mathcal{P}(t_0, \tau, x, \lambda)$ (Eq.(21)) for $u(x,t)$, then analogously as in Sec. III we can obtain N equations for unknown polynomial coefficients $u_1(x,\tau), \ldots, u_N(x,\tau)$.

The Fourier transform of Eq. (36) modified by λ is the following:

$$\frac{d\hat{u}(k,t)}{dt} = \lambda \left(-k^2 \hat{u}(k,t) + \widehat{u^2}(k,t) - \widehat{u^3}(k,t) \right) . \tag{39}$$

Note that in this case, thanks to the additivity of the FFT, it is sufficient to calculate in the k-space only the $\mathcal{N}$ values of the product -$K^2(k)\hat{u}(k)$, where

$$K^2 = \{0, (\Delta k)^2, \ldots, (-\Delta k)^2\}, \tag{40}$$

and

$$\hat{u} = \{\hat{u}(0), \hat{u}(\Delta k), \ldots, \hat{u}(-\Delta k)\} \,. \tag{41}$$

Then, the array representing $\frac{\partial^2 u(x,t)}{\partial x^2}$ in Eq. (36) can be calculated simply as $\mathrm{re}(\mathrm{IFFT}(-K^2\hat{u}))$, whereas conversion to k-space is not required from the remaining terms in Eq. (36). The numerical results corresponding to the solution of the Fisher equation in Eq. (36) have been shown in panel (a) and (b) of Fig. 7. The relative absolute error in panel (b) represents the maximum value of relative absolute error for the plot $u_N(x,t)$. The reason for the slower accumulation of absolute error compared to KdV is the lower number of FFT operations. They were only required to calculate $\frac{\partial^2 u(x,t)}{\partial x^2}$.

IV. 3. Inviscid Burgers Equation

To show the limitations of the presented numerical method we will refer to Example 6.2 of the paper [37] concerning the inviscid Burgers equation

$$\frac{\partial u(x,t)}{\partial t} = -u(x,t)\frac{\partial u(x,t)}{\partial x} \,, \tag{42}$$

where the initial condition $u_0(x)$ represents a piecewise continuous function of the following form:

$$u_0(x) = \begin{cases} 0\,, & |x| \geq 1/4, \\ 1/2 + 2x\,, & -1/4 < x \leq 0, \\ 1/2 - 2x\,, & 0 < x \leq 1/4, \end{cases} \tag{43}$$

which is determined on the spatial domain of $x \in [-0.5, 0.5]$. The expression representing the exact solution of Eq. (42) is available in the paper [37] at example 6.2. It develops a shock at time $t = 0.5$ and $x = 1/4$.

The appearance of a shock solution is a bit of a challenge for the spectral method presented in this study, where the method works effectively, when the right side of the differential equation is sufficiently smooth. Any solution that contains discontinuities and cusp-like shapes makes the method very inefficient. In practice, this means that for a given accuracy Δ of the numerical solution of the differential equation, the value of the self-adapting δ becomes very small due to the mismatch between the sine and cosine functions used in

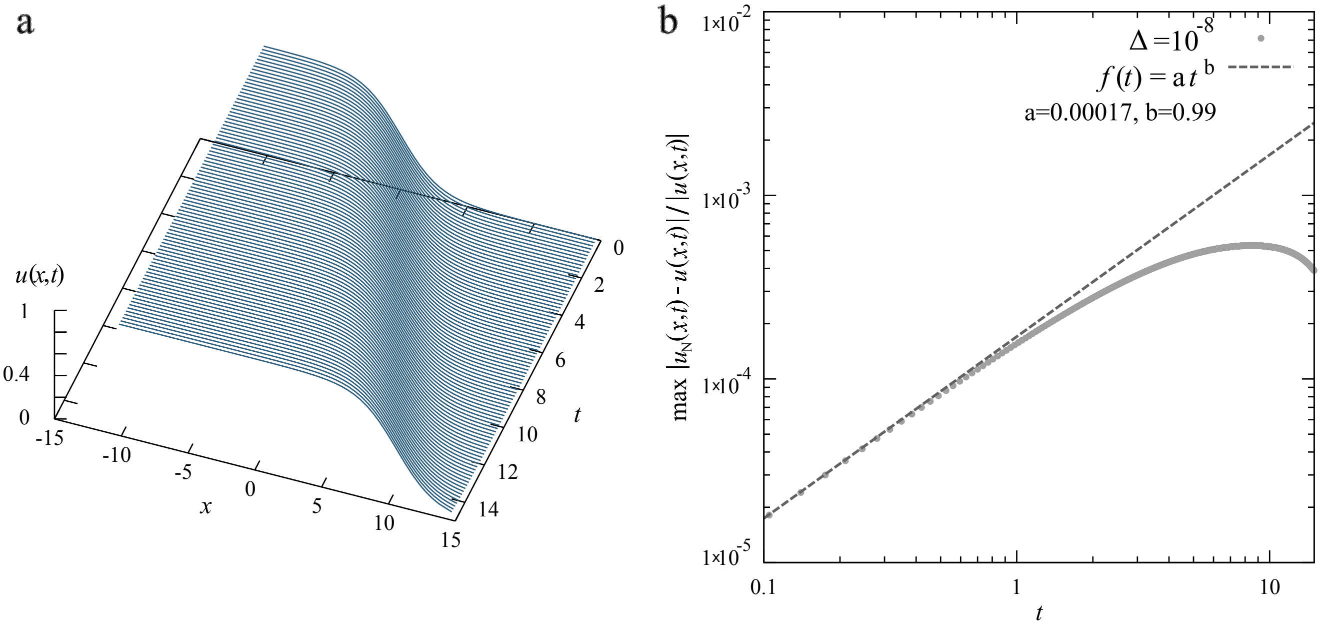

Fig. 7. Polynomial approximation $u_N(x,t)$ ($N = 8$) of the Fisher equation (Eq. (36)) reaction-diffusion front moving in x-direction: a) 2D surface representing space-time evolution of the front shape, b) dependence of the maximum value of relative absolute error for $u_N(x,t)$ on time, where $\Delta = 10^{-8}$

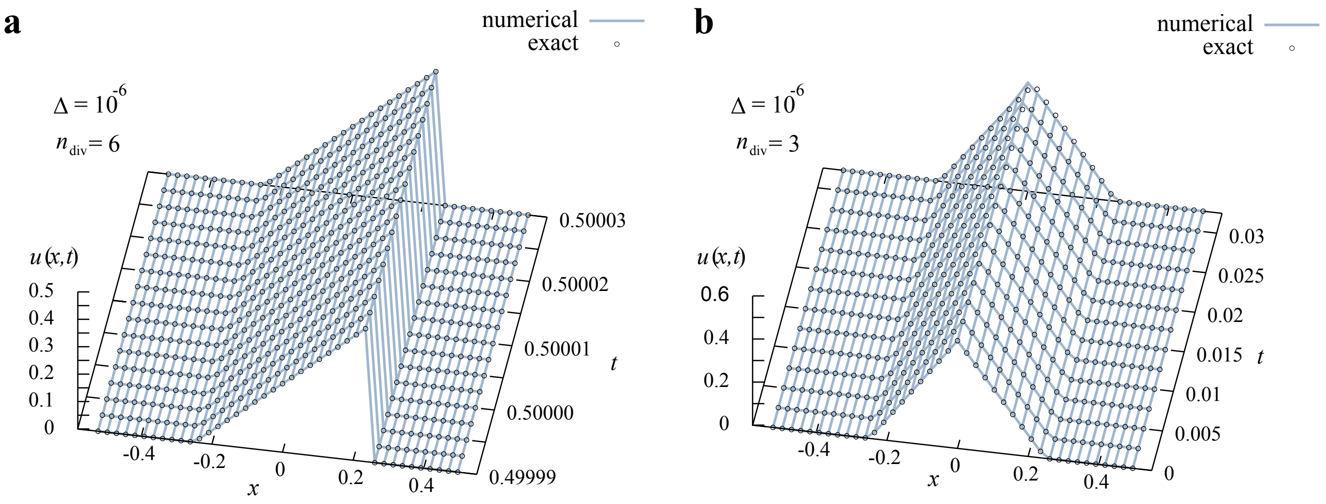

Fig. 8. Numerical solution (continuous lines) and exact ones (circle symbols) for the inviscid Burgers equation in Eq. (42) and the initial condition in Eq. (43)): a) for $t_0 = 0.49999$, $N = 7$, $n_{\text{div}} = 6$, b) for $t_0 = 0$, $N = 7$, $n_{\text{div}} = 3$. The value of $\Delta = 10^{-6}$

the FFT method and the shape of the solution. This in turn causes the number of updates of the initial condition $u_0(x)$ to increase significantly, analogous to the case with the number of iterations $N = 3$ in Fig. 3 for the soliton solution. This implies a rapid accumulation of the number of solutions that do not satisfy the Δ accuracy criterion. As can be seen from the example in panel (b) of Fig. 8, for values of the parameter $\Delta = 10^{-6}$ and $n_{\text{div}} = 3$, significant deviations from the exact value of the solution appear already for values of times of the order of 10^{-2}. In contrast, the results shown in panel (a) demonstrate the feasibility of the spectral method even in the region of discontinuity (the appearance of the shock). In the latter case, the initial condition close to the shock was chosen on the basis of the exact solution from the article [37] for $t_0 = 0.49999$. This example serves only to demonstrate the potential of the method for a larger class of solutions.

V. Conclusion

The presented method for finding numerical solutions to PDE problems is a complementary method to finite difference methods. It can be compared to existing methods such as Adomian decomposition [9] and the main difference is the use of FFT and related spectral methods in the construction of the numerical solution. In contrast, the construction of obtaining an approximate numerical solution for ODE problems is very similar. The method is ineffective for discontinuous solutions such as the shock wave problem, but shows potential for extension in this direction. The examples considered suggest that, despite numerical instability, the accumulation of numerical errors can be controlled. The method has the advantage of a simple iterative scheme that does not use the finite difference method and that reduces the solution of PDE problems to problems of an algebraic nature, in particular the polynomial problem.

Acknowledgment

M.D. thanks Prof. Maria Przybylska, Prof. Andrzej Drzewinski and Dr. Krzysztof Dudek for their comments. M.D. acknowledges the support of the project by the Ministry of Science under the "Regional Excellence Initiative" program, project NO. RID/SP/0050/2024/1. Some calculations have been carried out using resources provided by Wroclaw Centre for Networking and Supercomputing (https://wcss.pl), grant No. 82

References

[1] J.H.E. Cartwright, O. Piro, *The dynamics of Runge–Kutta methods*, Int. J. Bifurcat. Chaos **2**, 427–449 (1992).

[2] M.J. Ablowitz, B.M. Herbst, C. Schober, *On the Numerical Solution of the sine–Gordon Equation: I. Integrable Discretizations and Homoclinic Manifolds*, J. Comput. Phys. **126**, 299–314 (1996).

[3] M.J. Ablowitz, B.M. Herbst, C.M. Schober, *Discretizations, integrable systems and computation*, J. Phys. A- Math. Gen. **34**, 10671 (2001).

[4] D.J. Kouri, D.S. Zhang, G.W. Wei, T. Konshak, D.K. Hoffman, *Numerical solutions of nonlinear wave equations*, Phys. Rev. E **59**, 1274–1277 (1999).

[5] E. Infeld, G. Rowlands, *Nonlinear Waves, Solitons and Chaos*, Cambridge University Press (2000).

[6] R.S. Johnson, *A Modern Introduction to the Mathematical Theory of Water Waves*, Cambridge University Press (1997).

[7] H. Schaeffer, *Learning partial differential equations via data discovery and sparse optimization*, Proc. R. Soc. A **473**, 20160446 (2016).

[8] J.-H. He, *Homotopy perturbation technique*, Comput. Method. Appl. M. **178**, 257–262 (1999).

[9] G. Adomian, *Solving Frontier Problems of Physics: The Decomposition Method*, Springer (1994).

[10] A. Bratsos, M. Ehrhardt, I.T. Famelis, *A discrete Adomian decomposition method for discrete nonlinear Schrödinger equations*, Appl. Math. Comput. **197**, 190–205 (2008).

[11] H.N.A. Ismail, K.R. Raslan, G.S.E. Salem, *Solitary wave solutions for the general KdV equation by Adomian decomposition method*, Appl. Math. Comput. **154**, 17–29 (2008).

[12] T.A. Abassy, M.A. El-Tawil, H.K. Saleh, *The Solution of KdV and mKdV Equations Using Adomian Pade Approximation*, Int. J. Nonlin. Sci. Num. **5**, 327–340 (2004).

[13] D. Kaya, *A numerical solution of the sine-Gordon equation using the modified decomposition method*, Appl. Math. Comput. **143**, 0096–3003 (2003).

[14] J.-H. He, *Homotopy Perturbation Method for Bifurcation of Nonlinear Problems*, Int. J. Nonlin. Sci. Num. **6**, 207–208 (2005).

[15] M.S.H. Chowdhury, I. Hashim, O. Abdulaziz, *Application of homotopy-perturbation method to nonlinear population dynamics models*, Phys. Letters A **368**, 251–258 (2007).

[16] A.C. Loyinmi, T.K. Akinfe, *Exact solutions to the family of Fisher's reaction-diffusion equation using Elzaki homotopy transformation perturbation method*, Eng. Rep. **2**, e12084 (2020).

[17] E. Valseth, C. Dawson, *An unconditionally stable space–time FE method for the Korteweg–de Vries equation*, Comput. Methods Appl. Mech. Engrg. **371**, 0045–7825 (2020).

[18] B.G. Chen, N. Upadhyaya, V. Vitelli, *Nonlinear conduction via solitons in a topological mechanical insulator*, PNAS **111**, 13004–13009 (2014).

[19] B. Deng, M. Zanaty, A.E. Forte, K. Bertoldi, *Topological solitons make metamaterials crawl*, Phys. Rev. Appl. **17**, 014004 (2022).

[20] Y. Shen, I. Dierking, *Dynamics of electrically driven solitons in nematic and cholesteric liquid crystals*, Comm. Phys. **3**, 14 (2020).

[21] B.A. Malomed, D. Mihalache, F. Wise, L. Torner, *Spatiotemporal optical solitons*, J. Opt. B-Quantum S. O. **7**, R53 (2005).

[22] M. Izadi, H.M. Srivastava, *Numerical treatments of nonlinear Burgers-Fisher equation via a combined approximation technique*, Kuwait J. Sci. **51**, 100163 (2024).

[23] A.V. Slunyaev, A.V. Kokorina, E.N. Kokorina, *Nonlinear waves, modulations and rogue waves in the modular Korteweg-de Vries equation*, Commun. Nonlinear Sci. **127**, 107527 (2023).

[24] L. Ju, J. Zhou, Y. Zhang, *Conservation laws analysis of nonlinear partial differential equations and their linear soliton solutions and Hamiltonian structures*, Commun. Anal. Mech. **15**, 24–49 (2023).

[25] M. Meylan, *Nonlinear PDE's course*, https://wikiwaves.org/Numerical_Solution_of_the_KdV.

[26] X.-K. Wang, G.-B. Wang, *A Hybrid Method Based on the Iterative Fourier Transform and the Differential Evolution for Pattern Synthesis of Sparse Linear Arrays*, Int. J. Antenn. Propag. **2018**, 6309192 (2018).

[27] B. Brzostowski, M.R. Dudek, B. Grabiec, T. Nadzieja, *Non-finite-difference algorithm for integrating Newton's motion equations*, Phys. Status Solidi B **244**, 851–858 (2005).

[28] C. Peterson, *The Radial Equation for Hydrogen-Like Atoms*, J. Chem. Educ. **52**, 92–94 (1975).

[29] D.J. Korteweg, G. de Vries, *On the change of form of long waves advancing in a rectangular canal, and on a new type of long stationary waves*, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **39**, 422–443 (1895).

[30] A.C. Scott, F.Y.F. Chu, D.W. McLaughlin, *The soliton: A new concept in applied science*, Proceedings of the IEEE **61**, 1443–1483 (1973).

[31] M. Izadi, Ş. Yüzbaşi, *A hybrid approximation scheme for 1-D singularly perturbed parabolic convection-diffusion problems*, Math. Commun. **27**, 47–62 (2022).

[32] A.B. Downey, *Think DSP. Digital Signal Processing in Python*, Green Tea Press (2014).

[33] F. Harris, *On the use of windows for harmonic analysis with the discrete Fourier transform*, Proceedings of the IEEE **66**, 51–83 (1978).

[34] J. Chambarel, C. Kharif, O. Kimmoun, *Generation of two-dimensional steep water waves on finite depth with and without wind*, Eur. J. Mech. B-Fluid. **29**, 132–142 (2010).

[35] J.G. Verwer, W.H. Hundsdorfer, B.P Sommeijer, *Convergence properties of the Runge-Kutta-Chebyshev method*, Math. Sci. **57**, 157–178 (1990).

[36] C.R. Mittal, J.R. Kumar, *Numerical solutions of nonlinear Fisher's reaction–diffusion equation with modified cubic B-spline collocation method*, Math. Sci. **7**, 12 (2013).

[37] M. Izadi, *Applications of the Newton-Raphson method in a SDFEM for inviscid Burgers equation*, Comput. Meth. Diff. Eq. **8**, 708–732 (2020).

Mirosław Roman Dudek was born in 1956. He is employed as Professor at the University of Zielona Góra. His main field of interest is nanotechnology: experiment and theory. He has experience in the methods of statistical physics, computer modeling, issues of genetic evolution, and methods of DNA analysis.