University of Miskolc
Faculty of Mechanical Engineering and Informatics

Designing and Investigating Numerical Methods for Solving the Heat Conduction Equation

Booklet of PhD thesis

Prepared by

Mahmoud Saleh
Engineering of Mechanical (BSc),
Engineering of Manufacturing (MSc)

István Sályi Doctoral School of Mechanical Engineering Sciences

Topic Field of Basic Engineering Sciences
Topic Group of Transport Processes and Machines

Head of Doctoral School

Dr. Gabriella Bognár
DSc, Full Professor

Head of Topic Group

Dr. László Baranyi
Full Professor

Scientific Supervisor

Dr. Endre Kovács

Miskolc
2023

JUDGING COMMITTEE

Chair:

Secretary:

Members:

OFFICIAL REVIEWERS

1. INTRODUCTION

Variety of finite difference methods have been developed and applied to solve the PDEs, such as the heat conduction equation and its generalization, for example, the explicit method, the fully implicit method, and the ADI method [1]–[3]. One of the most common approaches to solve the PDEs numerically is to discretize the spatial variable which converts the PDEs into a system of ODEs. After that we can solve the system of ODEs at each time level [4]. Nevertheless, most of these methods are tested and evaluated under conditions where the coefficients in the equations, such as the diffusivity α, are independent of the space variable. However, there are systems in real applications where the physical properties of material can be drastically different at adjacent points, for instance in a microprocessor. As a result, the coefficients and, consequently, the eigenvalues of the matrix system can have a range of several order of magnitude so the problem can be extremely stiff.

The traditional explicit methods, either Adams-Bashforth or Runge-Kutta types, are conditionally stable, very small time-step sizes must be applied regardless of the measurement errors of the input data and the requirements on the accuracy of the output. It implies that the solution blows up if the time-step size exceeds the threshold number, or what is known as the CFL limit. Even the professional commercial adaptive time-step size solvers such as ode23 and ode45 of MATLAB can experience instability when the tolerance is not so small [5].

On the other hand, implicit methods offer much better stability properties, and this why they are widely applied to solve these equations [6]–[8]. For example, in his work [9] Mascagni applied the backward Euler method to the Hodgkin-Huxley equations. Manaa and Sabawi studied and compared the explicit Euler and implicit Crank-Nicolson methods when they are applied to Huxley equation. They found that the explicit Euler method was faster, but less stable and accurate than Crank-Nicolson [10]. Coupled hydrodynamics and nonlinear heat conduction problems were solved numerically by Kadioglu and Knoll by treating the heat equation implicitly and the hydrodynamics explicitly [11]. They mentioned that this technique, the so called IMEX, is typical for solving such kind of problems. Another example is in the field of reservoir-simulation, where the pressure equation is treated implicitly, and saturation equation is treated explicitly [12].

The most significant problem with implicit method is that each time step requires the solution of a system of algebraic equations, which cannot simply be parallelized. In case of one-dimensional system where the matrix is tridiagonal and the number of nodes is small, the numerical computation can be fast, and it is hard to beat the implicit method. In contrast, the numerical computations can be time-consuming when we use the implicit method to solve more complicated systems, such as reservoir-simulation with one trillion cells. However, there is an increasing trend towards parallelism in the recent years [13]. The second problem with most implicit or explicit methods is that they can lead to qualitatively unacceptable solutions, such as unphysical oscillations or negative values of the otherwise non-negative variables. These variables can be concentrations, densities, or temperatures measured in Kelvin, and the numerical methods should preserve their positivity.

There exist unconditionally stable explicit methods in the case of linear heat equation. For instance, the odd-even hopscotch algorithm [14] and the Alternating Direction Explicit (ADE) scheme [15] both have second order temporal accuracy.

In my research, I worked with my supervisor on improving and investigating families of conventional and novel explicit methods for solving linear and nonlinear diffusion equation based on fundamentally new way of thinking. In some cases, the improved methods are proven to be unconditionally stable, positivity preserving. Those schemes are applied to extremely stiff and inhomogeneous systems. Also, some adaptive time step controllers are constructed and investigated.

All the methods I developed and investigated are of the explicit type. They have been applied to the heat conduction equation after it had been spatially discretized. The second order linear parabolic partial diffusion equation, or the so-called heat can be written as follows

$$c\rho \frac{\partial u}{\partial t} = \nabla\left(k\nabla u\right) + c\rho q, \tag{1.1}$$

Where u is the temperature (the concentration in case of diffusion-equation), while q, k, c, and ρ are the intensity of heat sources (chemical reactions, radioactive decay, radiation, etc.), heat conductivity, specific heat and (mass) density, respectively.

The most typical finite difference scheme to discretize the space variable is the second order central difference formula, and if it is applied to Eq. (1.1) in one spatial dimension we get

$$\frac{du_{\mathrm{i}}}{dt} = \frac{A}{c_{\mathrm{i}}\rho_{\mathrm{i}}A\Delta x}\left(k_{\mathrm{i,i+1}} \cdot \frac{u_{\mathrm{i+1}} - u_{\mathrm{i}}}{\Delta x} + k_{\mathrm{i-1,i}} \cdot \frac{u_{\mathrm{i\text{-}1}} - u_{\mathrm{i}}}{\Delta x}\right) + Q_{\mathrm{i}}, \tag{1.2}$$

where u_i is the temperature of the cell i, $C = c.m = \rho c V$ is the heat capacity of that cell in $(\mathrm{J/K})$ units (m is the mass, $V = A\Delta x$ is the volume of the cell). We introduce two other quantities, the heat source term Q,

$$Q_{\mathrm{i}} = \frac{1}{V_{\mathrm{i}}}\int_{V_i} q dV \approx q \;\text{ in } \left[\frac{K}{s}\right]\text{units,} \tag{1.3}$$

and the thermal resistance $R_{\mathrm{ij}} = \frac{\Delta x}{k_{\mathrm{ij}}A}$ in $\left(K/W\right)$ units. In case of nonequidistant grid, the distances between the center of cells are $d_{ij} = \left(\Delta x_i + \Delta x_j\right)/2$ and the resistances can be calculated by the simple approximation as $R_{ij} \approx \frac{d_{ij}}{k_{ij} A_{ij}}$. Using the introduced quantities and Eq. (1.3) we can write

$$\frac{du_i}{dt} = \frac{u_{i-1} - u_i}{R_{i-1,i} C_i} + \frac{u_{i+1} - u_i}{R_{i+1,i} C_i} + Q_i. \tag{1.4}$$

In case of homogeneous one-dimensional system in the absence of the heat source and with equidistance grid, the previous equation can be written as follows

$$\frac{du_i}{dt} = \alpha \frac{u_{i-1} - 2u_i + u_{i+1}}{\left(\Delta x\right)^2}. \tag{1.5}$$

The ODE system for a general (perhaps unstructured) grid, which gives the time derivative of each temperature independently of any coordinate-system is given as follows

$$\frac{du_i}{dt} = \sum_{j \neq i} \frac{u_j - u_i}{R_{i,j} C_i} + Q_i. \tag{1.6}$$

The last equation can be considered as the spatial discretisation of Eq. (1.1) not only in one dimension but also in two or three dimensions.

2. METHODOLOGY OF STUDY

My research work, I designed and investigated numerical methods for solving the heat conduction equation. The work can be classified into three directions. In the first direction, I systematically constructed and tested novel numerical methods odd-even hopscotch-type. In the second direction, families of adaptive time-step controllers of I-type and PI-type were proposed. In the third direction, other families of adaptive controllers of I-type were designed and applied to solve the heat equation when the diffusion coefficient depends on time and space simultaneously.

2.1 The Odd-Even Hopscotch Schemes OEH

The odd-even hopscotch (OEH) algorithm firstly appeared in Gordon's work in 1965 [14]. To understand the structure of OEH, let us consider Eq. (1.6). We illustrate the so-called bipartite grid, where in that grid the set of nodes are divided into two subsets which are A (odd nodes, dark dots in Figure 2.1) and B (even nodes, light dots in Figure 2.1). The calculation of the values of the unknown variable u consists of two stages. In the first stage, the values u of the subset A are calculated at time level $(n+1)$ using the only the values at time level (n). This process is depicted by thin black arrows in Figure 2.1. In the second stage, the values u of the subset B are calculated at time level $(n+1)$ using the values at time levels (n) and $(n+1)$ as well. This process is depicted by thick red arrow in Figure 2.1. In the time level $(n+1)$, we change the role of the subsets A and B. In other words, the values of subset B are calculated in the first stage and then the values of subset A are calculated in the second stage. The first stage in the original OEH uses the explicit Euler scheme, while the second stage uses the implicit Euler scheme.

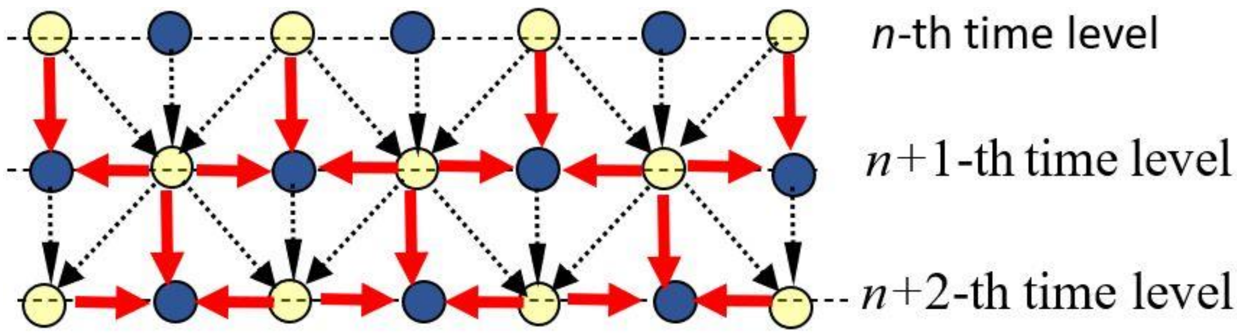

Figure 2.1. The stencil of the odd-even hopscotch algorithm. Thin black arrows (thick red arrows) indicate operations at Stage 1 (Stage 2).

In The original hopscotch method, the explicit Euler method is used in the first stage while the implicit Euler method is used in the second stage. In our research we systematically constructed and tested new methods of odd-even hopscotch-type by changing the methods used in each stage. For simplicity, I will illustrate those method in the case of one space dimensional equation in the absence of the heat source and considering that the diffusion coefficient is constant as in Eq. (1.5) . Before introducing the formulas of the schemes, I will introduce the formula mesh ratio quantity

$$r = \frac{\alpha h}{\Delta x^2}, \quad h = \Delta t, \tag{2.1}$$

Where h is the time step size. The general formulas in the case of inhomogeneous 2D system are explained in more details in my dissertation. The formulas of the first and second stages of hopscotch can be written.

Stage 1 formulas

A) Explicit Euler

$$u_i^{\mathrm{n}+1} = (1-2r)u_i^{\mathrm{n}} + r\left(u_{i-1}^{\mathrm{n}} + u_{i+1}^{\mathrm{n}}\right). \tag{2.2}$$

B) UPFD

Unconditionally positivity-preserving scheme can be found in [16]. The formula can be written as follows

$$u_i^{\mathrm{n}+1} = \frac{u_i^{\mathrm{n}} + r\left(u_{i-1}^{\mathrm{n}} + u_{i+1}^{\mathrm{n}}\right)}{1+2r}. \tag{2.3}$$

C) Explicit-neighbour “Crank-Nicolson”

$$u_i^{\mathrm{n}+1} = \frac{(1-r)u_i^{\mathrm{n}} + r\left(u_{i-1}^{\mathrm{n}} + u_{i+1}^{\mathrm{n}}\right)}{1+r}. \tag{2.4}$$

D) CNe, constant neighbour

The constant neighbour method is a new method and can be found in [17]. The final formula can be written as follows

$$u_i^{\mathrm{n}+1} = u_i^{\mathrm{n}} e^{-2r} + \frac{u_{i-1}^{\mathrm{n}} + u_{i+1}^{\mathrm{n}}}{2}\left(1-e^{-2r}\right). \tag{2.5}$$

Stage 2 formulas

1. Explicit Euler

$$u_i^{\mathrm{n}+1} = (1-2r)u_i^{\mathrm{n}} + r\left(u_{i-1}^{\mathrm{n}+1} + u_{i+1}^{\mathrm{n}+1}\right). \tag{2.6}$$

2. UPFD (Implicit Euler)

$$u_i^{n+1} = \frac{u_i^n + r\left(u_{i-1}^{n+1} + u_{i+1}^{n+1}\right)}{1+2r}. \tag{2.7}$$

3. Crank-Nicolson

$$u_i^{n+1} = \frac{(1-r)u_i^n + r\left(u_{i-1}^{n+1} + u_{i+1}^{n+1}\right)}{1+r}. \tag{2.8}$$

4. Implicit-Neighbour Crank-Nicolson

$$u_i^{n+1} = \frac{(1-r)u_i^n + r\left(u_{i-1}^{n+1} + u_{i+1}^{n+1}\right)}{1+r}. \tag{2.9}$$

5. Constant Neighbour CNe

$$u_i^{n+1} = u_i^n e^{-2r} + \frac{u_{i-1}^{n+1} + u_{i+1}^{n+1}}{2}\left(1-e^{-2r}\right). \tag{2.10}$$

6. Linear Neighbour LNe

The linear neighbour method is a new method of the second order and the details of that method can be found in [18]. The scheme can be written as follows

$$u_i^{n+1} = u_i^n e^{-2r} + \frac{1}{2}\left(u_{i-1}^n + u_{i+1}^n\right)\left(\frac{1-e^{-2r}}{2r} - e^{-2r}\right) + \frac{1}{2}\left(u_{i-1}^{n+1} + u_{i+1}^{n+1}\right)\left(1 - \frac{1-e^{-2r}}{2r}\right). \tag{2.11}$$

Each formula from the first stage can be combined with six possible formulas from the second stage in hopscotch structure resulting in $4 \times 6 = 24$ possible methods. For instance, the methods denoted by A2 refers to the original well-known OEH, while the method denoted by A6 means that the explicit Euler formula is used in the first stage and the linear-neighbour formula is used in the second stage.

In the first phase, we examine two space-dimensional rectangle-type lattices of $N = N_x \times N_z$ cells with zero Neumann boundary conditions, i.e., thermal isolation. Several numerical experiments have been conducted in order to test the behaviour of the 24 methods. The methods have been evaluated based on three crucial properties which are the stability, positivity and the order of convergence. The results for the 24 OEH combinations are summarised in Table 2.1. I emphasize that at this point those properties are evaluated only through numerical experiments without any analytical investigation. Based on these intensive tests, I chose algorithms A2, B1, C4, C5, D4 and D5 for further numerical and analytical investigation. In the second phase, we applied those methods to linear diffusion equation considering strongly inhomogeneous media. Their performances were compared to the performance of well-known and widely used solvers such the “ode” package in MATLAB. Also, the methods were applied to solve nonlinear Fisher’s equation and compared with original hopscotch method. Figure 2.2 shows the error produced by the schemes as a function of the time-step size and the total running time when the methods are used to solve two space-dimensional. The system is subjected to zero Neumann boundary conditions and the properties of the media are extremely inhomogeneous. In the third phase, we mathematically analysed the stability and the convergence of B1 and D5 methods which showed the best performance comparing to others. I analytically proved that both of them are unconditionally stable when they are applied to a linear diffusion equation and they are of the second order in the time step size.

	1) Exp Eu	2) UPFD	3) CrN	4) IN CrN	5) CNe	6) LNe
A) Exp Eu	**U**	**S 2**	**U**	**U**	**S**	**S**
B) UPFD	**S 2**	**P**	**S**	**S**	**P**	**S**
C) EN CrN	**U**	**S**	**S**	**S 2**	**S 2**	**S**
D) CNe	**S**	**P**	**S**	**S 2**	**P 2**	**P**

Table 2.1. Properties of the 24 algorithms. The letter U, S and P mean unstable, stable, and positivity preserving, respectively. The number 2 means that the algorithm is second order (all other schemes are first order).

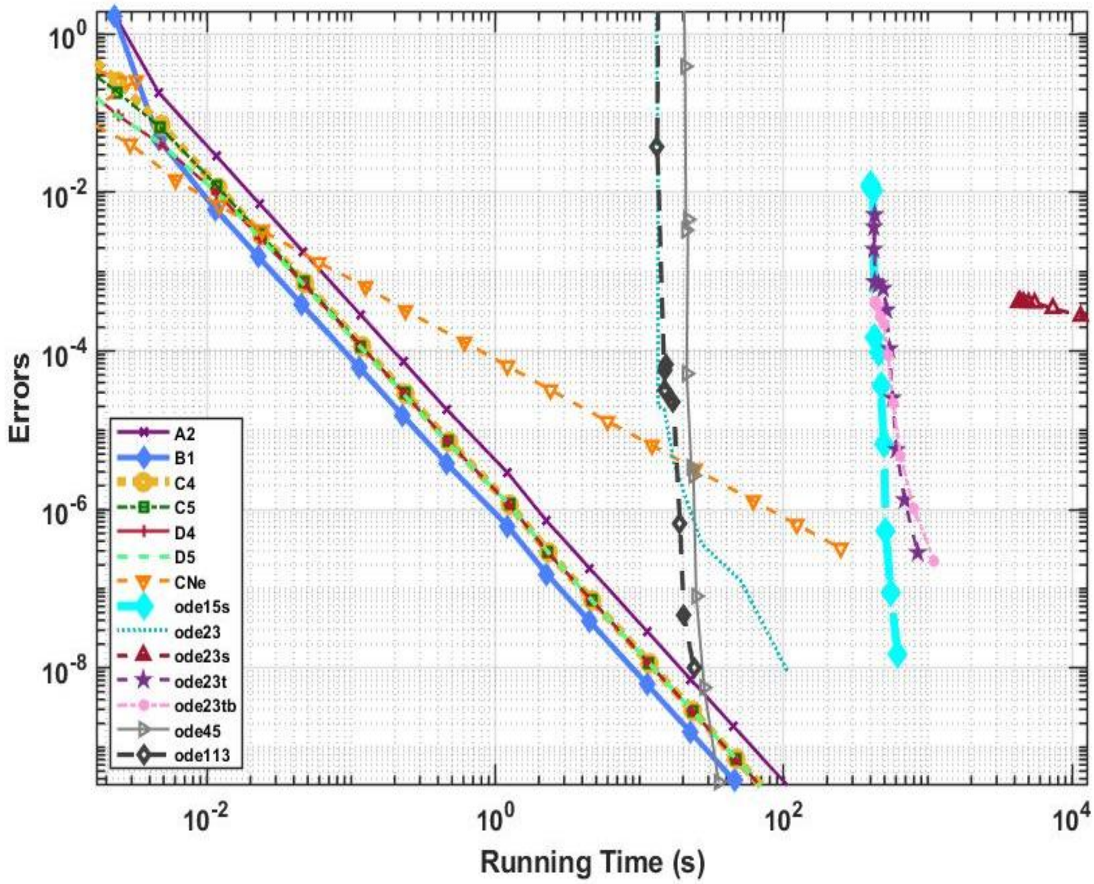

Figure 2.2. The errors as a function of the total running times in case of OEH type methods and seven different MATLAB solvers.

2.2 Families of Adaptive Time Step Controllers for Solving the Non-stationary Heat Conduction Equation

I systematically test families of explicit adaptive step-size controllers for solving the diffusion or heat equation. Those controllers are applied to a system of ODEs resulted from spatially discretized diffusion equation. The performance of the controllers was compared in the case of three different experiments. The first and the second system are heat conduction in homogeneous and inhomogeneous medium, while the third one contains a moving heat source that can correspond to a welding process.

There are at least three crucial factors when it comes to designing adaptive step-size integrators: the method of calculating the solution at the end of the actual time step, the method of estimating the local error in each time step, and the approach for changing the time-step [19]. Although I used different methods for estimating the local when designing the adaptive controller, but the main interest of my research was focused on comparing the performance of two approaches for changing the step size, which are the well-known elementary controller (I) and the proportional-integral controller (PI) [20].

In the literature, the controllers of PI were applied to change the step size only in the case of small systems of ODEs. Those algorithms might be efficient when they are applied to a single ordinary differential equation or a system of ODEs including limited number of equations. They proved to show better performance than the controllers of type I. In my research, I designed and extensively tested adaptive step size controllers of both types. The performance of the both types were compared when they are applied to a big system of ODEs resulted from spatially discretized diffusion equation. Our numerical experiments showed that adaptive schemes using PI controller do not have any advantage compared to the same schemes using I controller.

2.3 Adaptive Time Step Controllers for the Transient Diffusion Equation with Coefficient Depending on both Space and Time

I dealt with the time-dependent diffusion equation in one dimension, where the diffusion coefficient itself depends simultaneously on space and time and its formula can be written as follows

$$\bar{\alpha}(\eta) = \alpha \eta^m, \tag{2.12}$$

where α is a constant, whose physical dimension depends on the concrete value of m. The variable η in the last equation depends on space and time: $\eta = \frac{x}{\sqrt{t}} \in \mathbb{R}$. Inserting that into the diffusion equation we obtain

$$\frac{\partial u(x,t)}{\partial t} = \alpha \frac{\partial}{\partial x}\left(\eta^m \frac{\partial u(x,t)}{\partial x}\right) = \alpha\left(m\eta^{m-1}\frac{\partial \eta}{\partial x}\frac{\partial u(x,t)}{\partial x} + \eta^m \frac{\partial^2 u(x,t)}{\partial x^2}\right). \tag{2.13}$$

In our published work [21], Ferenc Barna has introduced the analytical solution of Eq. (2.13) which was used as a reference solution in my numerical experiments.

I introduce and design four families of adaptive time-step controllers to solve numerically Eq. (2.13). Two of these families (families A and B) are novel and have been proposed and applied for the first time. The other two families (families C and D) are the well-known Runge-Kutta Cash-Karp and Runge-Kutta-Fehlberg methods. I have to emphasize that those adaptive controllers are only of type-I since I showed in Subsection 2.2 that controller of type-PI does not have any advantage when it is applied to a big system of ODEs. Briefly I will introduce the core of these families.

A) The adaptive LNe3 schemes

The LNe3 scheme is a novel explicit scheme which consists of three stages [18]. In the first stage, it uses the constant neighbour method (CNe) to compute the values of the unknown variable. This value is taken as predictor for the second stage. In the second stage, the LNe3 uses the linear neighbour method to compute the value of the unknown. Again, this value is taken as a predictor for the third stage in which the linear neighbour method is used. The difference between the computed values of the unknown variable in between each two stages can be considered as an estimation of the local error. So, in LNe3 scheme the local error can be estimated by three possibilities, for example, the difference between the values calculated in the first and second stages. If we considered that local error estimator and used the I-type controller to change the time step, an adaptive controller is constructed and will be denoted by ALNe3-C1L2. The letters C1 and L2 in the last nomenclature refer to the stages used to estimate the local error. In similar way can construct two other adaptive controllers denoted by ALNe3-C1L3 and ALNe3-L2L3.

B) The adaptive CLL schemes

The CLL scheme [22] is a modification of the LNe3 algorithm in order to achieve third order temporal convergence. It consists of three stages. It uses fractional time steps during the first and second stages and a full-time step in the third stage. Generally, the length at the first stage is $h_1 = ph$, $2/3 \le p < 2$, but at the second stage it is always $h_2 = 2h/3$. In CLL scheme, the local error can be estimated by only two possibilities. We can estimate the local error by considering the difference between the computed values of the unknown variable in the first and second stage provided that $p = 2/3$. Using the I-type controller, we can design an adaptive time step controller denoted by ACLL-C1L2. If $p = 1$, another local error estimation can be considered and we obtain an adaptive time step controller denoted by ACLL-C1L3.

C) Runge-Kutta Cash-Karp Method RKCK

Since it is a well-known method, and explained in detail in [23, p. 717], I think that it is not necessary to describe the tedious processes of implementing the method. This method is denoted by RKCK in my work.

D) Runge-Kutta-Fehlberg Method

Plenty of references discussed and implemented this method. Here I will refer to [24], where the authors show how to estimate the local error using Eq. (5.55) in that reference. it will be referred to as RKF in my work.

Those controllers were investigated and the numerical experiments showed that showed that the performances of the novel adaptive controllers severely overcome the widely used schemes, which are Fehlberg Runge-Kutta and Cash-Karp Runge-Kutta. Figure 2.3 shows the result of my numerical experiments for the following values

$m=7.2$, $\alpha=11.4$, $c=0.0042$, $N=500$, $x_0=0.48$, $\Delta x=5\cdot10^{-4}$, $t^0=0.9$, $t^{\text{fin}}=1.5$.

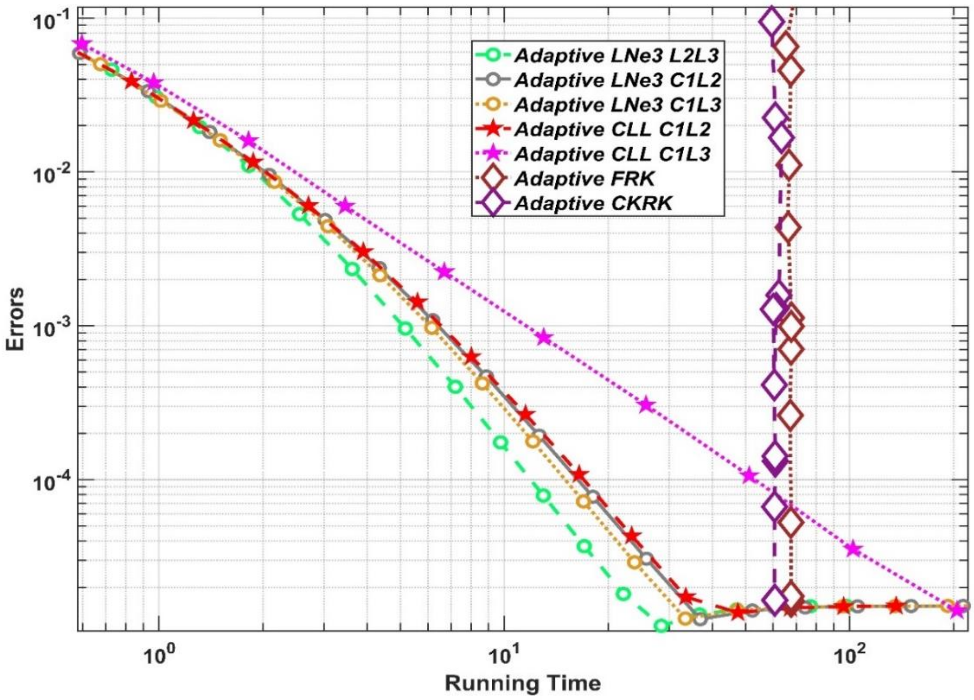

Figure 2.3. The errors as a function of the running times.

3. THESES – NEW SCIENTIFIC RESULTS

T1. I constructed 24 schemes by combining conventional and non-conventional schemes within the odd-even hopscotch structure to obtain two stage methods. Then I produced preliminary numerical results and based on these I chose the 6 most efficient methods for further investigation. However, the original hopscotch method (A2) was one of these six methods. The performance of the selected methods was examined in the case of two 2-dimensional systems containing 10000 cells with very inhomogeneous randomly generated parameters and initial conditions. I showed that the proposed methods are competitive as they can give results with acceptable accuracy orders of magnitude faster than the well-optimized MATLAB routines.

T2. The results showed that our novel hopscotch-based methods B1, C4, C5, D4, D5 are faster than the original hopscotch method (A2) when they are applied to a linear system with relatively high stiffness ratio. However, B1 method has the best performance comparing to all selected methods, and its advantage becomes larger when the system becomes more stiff. Based on the different numerical results, I selected those two methods which were proven to have the most valuable properties, namely, the reversed hopscotch (B1) and the CNe-CNe hopscotch (D5) algorithms. Then I analytically proved that their stability is guaranteed for the linear diffusion equation and that their convergence is second order in the time step size.

T3. Our novel hopscotch-based methods were applied to Fisher's equation. The results showed that the performance of the original hopscotch method (A2) is very poor when it is compared to performances of our novel hopscotch-based methods. I could prove analytically, in case of the CNe-CNe hopscotch (D5) scheme, that the way I treated the nonlinear reaction term guaranteed that the values of the unknown variable will remain in the unit interval if the initial values of that unknown are in unit interval, which in turn implies the positivity preserving property.

T4. I systematically designed and tested families of adaptive time step controllers, based on PI and I controllers, for solving a system of ODEs resulted from spatially discretized linear diffusion equation. Several studies claimed that the adaptive step controllers of type PI are better than I type for solving a system of ODEs. Those studies compared the two types of controllers when they are applied to a single ODE or a small system of ODEs, and in the literature, I could not find any study which compare them when they are applied to a big system of ODEs. However, our result showed that adaptive schemes using PI controller do not have any advantage compared to the same schemes using I controller when they are applied to a big system of ODEs resulted from discretizing the space variable in the linear diffusion equation.

T5. Using the linear neighbour LNe3 method, I designed a novel adaptive time step controller of type I and applied it to a system of ODEs resulted from spatially discretized linear diffusion equation in the absence and in the presence of the heat source. In both cases, I conducted the numerical experiments in inhomogeneous media with relatively high stiffness ratio. The result showed that the adaptive LNe3 is much faster than the adaptive schemes of Runge-Kutta when high accuracy is not required. The adaptive Runge-Kutta schemes are faster at the level of accuracy which is not required in the engineering application.

T6. Using LNe3 and CLL methods, families of novel adaptive time step controllers of type I are constructed. I treated the non-steady-state linear diffusion equation, where the diffusion coefficient itself depends simultaneously on space and time. I discretized the space variable in that equation to obtain a system of ODEs, then I used our adaptive controller to solve that system of equations. The numerical experiments showed that the performances of our adaptive controllers severely outperform the widely used schemes, which are Fehlberg Runge-Kutta and Cash-Karp Runge-Kutta. Recall that a lot of efforts have been made to improve traditional solvers by using the so-called PI and PID controllers. The LNe3 and the CLL-based adaptive controllers could change the time-step size smoothly using only the elementary controller without any need to implement the PI controller. I consider this as another advantage of these methods.

4. LIST OF PUBLICATIONS RELATED TO THE TOPIC OF THE RESEARCH FIELD

(1) M. Saleh, E. Kovács, and I. F. Barna, "Analytical and the Transient Diffusion Equation with Diffusion Coefficient Depending on Both Space and Time," *Algorithms*, vol. 16, no. 4, p. 184, Mar. 2023, doi: 10.3390/ Numerical Results for a16040184.

(2) M. Saleh, E. Kovács, and N. Kallur, "Adaptive step size controllers based on Runge-Kutta and linear-neighbor methods for solving the non-stationary heat conduction equation," *Networks and Heterogeneous Media*, vol. 18, no. 3, pp. 1059–1082, 2023, doi: 10.3934/nhm.2023046.

(3) M. Saleh, E. Kovács, and Á. Nagy, "New stable, explicit, second order hopscotch methods for diffusion-type problems," *Math Comput Simul*, vol. 208, pp. 301–325, Jun. 2023, doi: 10.1016/j.matcom.2023.01.029.

(4) Endre Kovács, Mahmoud Saleh, Imre Ferenc Barna, László Mátyás, "New Analytical Results and Numerical Schemes for Irregular Diffusion Processes," DIFFUSION FUNDAMENTALS 35 pp. 1-15. , 15 p. (2022)

(5) M. Saleh, E. Kovács, I. F. Barna, and L. Mátyás, "New Analytical Results and Comparison of 14 Numerical Schemes for the Diffusion Equation with Space-Dependent Diffusion Coefficient," *Mathematics*, vol. 10, no. 15, p. 2813, Aug. 2022, doi: 10.3390/math10152813.

(6) E. Kovács, Á. Nagy, and M. Saleh, "A New Stable, Explicit, Third-Order Method for Diffusion-Type Problems," *Adv Theory Simul*, vol. 5, no. 6, p. 2100600, Jun. 2022, doi: 10.1002/adts.202100600.

(7) E. Kovács, Á. Nagy, and M. Saleh, "A Set of New Stable, Explicit, Second Order Schemes for the Non-Stationary Heat Conduction Equation," *Mathematics*, vol. 9, no. 18, p. 2284, Sep. 2021, doi: 10.3390/math9182284.

(8) M. Saleh and E. Kovács, "Drag coefficient calculation of modified Myring-Savonius wind turbine with numerical simulations," *Design of Machines and Structures*, vol. 10, no. 2, pp. 73–84, 2020, doi: 10.32972/dms.2020.017.

(9) Á. Nagy, M. Saleh, I. Omle, H. Kareem, and E. Kovács, "New Stable, Explicit, Shifted-Hopscotch Algorithms for the Heat Equation," *Mathematical and Computational Applications*, vol. 26, no. 3, p. 61, Aug. 2021, doi: 10.3390/mca26030061.

(10) M. Saleh, Á. Nagy, and E. Kovács, "Construction and investigation of new numerical algorithms for the heat equation: Part 1," *Multidiszciplináris tudományok*, vol. 10, no. 4, pp. 323–338, 2020, doi: 10.35925/j.multi.2020.4.36.

(11) M. Saleh, Á. Nagy, and E. Kovács, “Construction and investigation of new numerical algorithms for the heat equation: Part 2,” *Multidiszciplináris tudományok*, vol. 10, no. 4, pp. 339–348, 2020, doi: 10.35925/j.multi.2020.4.37.

(12) M. Saleh, Á. Nagy, and E. Kovács, “Construction and investigation of new numerical algorithms for the heat equation: Part 3,” *Multidiszciplináris tudományok*, vol. 10, no. 4, pp. 349–360, 2020, doi: 10.35925/j.multi.2020.4.38.

(13) M. Saleh, E. Kovács, and G. Pszota, “Testing and improving a non-conventional unconditionally positive finite difference method,” *Multidiszciplináris tudományok*, vol. 10, no. 4, pp. 206–213, 2020, doi: 10.35925/j.multi.2020.4.24.

Under Review:

Endre Kovács, János Majár, Mahmoud Saleh, “Unconditionally positive, explicit, fourth order method for the diffusion- and Nagumo-type diffusion-reaction equations,”. Submitted to the “Journal of Scientific Computing D1” on 27 Sep 2022

5. REFERENCES

[1] U. M. Ascher, S. J. Ruuth, and B. T. R. Wetton, “Implicit-Explicit Methods for Time-Dependent Partial Differential Equations,” *SIAM J Numer Anal*, vol. 32, no. 3, pp. 797–823, Jun. 1995, doi: 10.1137/0732037.
[2] G. D. Smith, *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, 3rd ed. Clarendon Press, 1986.
[3] Jui-Ling Yu, “A FULLY EXPLICIT OPTIMAL TWO-STAGE NUMERICAL SCHEME FOR SOLVING REACTION-DIFFUSION-CHEMOTAXIS SYSTEMS,” Michigan State University, 2005.
[4] D. A. Anderson, J. C. Tannehill, R. H. Pletcher, M. Ramakanth, and V. Shankar, *Computational Fluid Mechanics and Heat Transfer*. Fourth edition. | Boca Raton, FL : CRC Press, 2020. | Series: Computational and physical processes in mechanics and thermal sciences: CRC Press, 2020. doi: 10.1201/9781351124027.
[5] E. Kovács, Á. Nagy, and M. Saleh, “A Set of New Stable, Explicit, Second Order Schemes for the Non-Stationary Heat Conduction Equation,” *Mathematics*, vol. 9, no. 18, p. 2284, Sep. 2021, doi: 10.3390/math9182284.
[6] P. O. Appau, O. K. Dankwa, and E. T. Brantson, “A comparative study between finite difference explicit and implicit method for predicting pressure distribution in a petroleum reservoir,” *International Journal of Engineering, Science and Technology*, vol. 11, no. 4, pp. 23–40, Oct. 2019, doi: 10.4314/ijest.v11i4.3.
[7] A. Moncorgé, H. A. Tchelepi, and P. Jenny, “Modified sequential fully implicit scheme for compositional flow simulation,” *J Comput Phys*, vol. 337, pp. 98–115, May 2017, doi: 10.1016/j.jcp.2017.02.032.
[8] A. Costa-Solé, E. Ruiz-Gironés, and J. Sarrate, “High-order hybridizable discontinuous Galerkin formulation with fully implicit temporal schemes for the simulation of two-phase flow through porous media,” *Int J Numer Methods Eng*, vol. 122, no. 14, pp. 3583–3612, Jul. 2021, doi: 10.1002/nme.6674.
[9] M. Mascagni, “The Backward Euler Method for Numerical Solution of the Hodgkin–Huxley Equations of Nerve Conduction,” *SIAM J Numer Anal*, vol. 27, no. 4, pp. 941–962, Aug. 1990, doi: 10.1137/0727054.
[10] S. Manaa and M. Sabawi, “Numerical Solution and Stability Analysis of Huxley Equation,” *AL-Rafidain Journal of Computer Sciences and Mathematics*, vol. 2, no. 1, pp. 85–97, Jun. 2005, doi: 10.33899/csmj.2005.164070.
[11] S. Y. Kadioglu and D. A. Knoll, “A fully second order implicit/explicit time integration technique for hydrodynamics plus nonlinear heat conduction problems,” *J Comput Phys*, vol. 229, no. 9, pp. 3237–3249, May 2010, doi: 10.1016/j.jcp.2009.12.039.
[12] H. Chen, J. Kou, S. Sun, and T. Zhang, “Fully mass-conservative IMPES schemes for incompressible two-phase flow in porous media,” *Comput Methods Appl Mech Eng*, vol. 350, pp. 641–663, Jun. 2019, doi: 10.1016/j.cma.2019.03.023.

[13] F. Gagliardi, M. Moreto, M. Olivieri, and M. Valero, “The international race towards Exascale in Europe,” *CCF Transactions on High Performance Computing*, vol. 1, no. 1, pp. 3–13, May 2019, doi: 10.1007/s42514-019-00002-y.

[14] P. Gordon, “Nonsymmetric Difference Equations,” *Journal of the Society for Industrial and Applied Mathematics*, vol. 13, no. 3, pp. 667–673, Sep. 1965, doi: 10.1137/0113044.

[15] H. Liu and S. Leung, “An alternating direction explicit method for time evolution equations with applications to fractional differential equations,” *Methods and Applications of Analysis*, vol. 26, no. 3, pp. 249–268, 2019, doi: 10.4310/MAA.2019.v26.n3.a3.

[16] M. K. Kolev, M. N. Koleva, and L. G. Vulkov, “An Unconditional Positivity-Preserving Difference Scheme for Models of Cancer Migration and Invasion,” *Mathematics*, vol. 10, no. 1, p. 131, Jan. 2022, doi: 10.3390/math10010131.

[17] E. Kovács, “New stable, explicit, first order method to solve the heat conduction equation,” *Journal of Computational and Applied Mechanics*, vol. 15, no. 1, pp. 3–13, 2020, doi: 10.32973/jcam.2020.001.

[18] E. Kovács, “A class of new stable, explicit methods to solve the non-stationary heat equation,” *Numer Methods Partial Differ Equ*, vol. 37, no. 3, pp. 2469–2489, May 2021, doi: 10.1002/num.22730.

[19] L. F. Shampine and H. A. Watts, “Comparing Error Estimators for Runge-Kutta Methods,” *Math Comput*, vol. 25, no. 115, p. 445, Jul. 1971, doi: 10.2307/2005206.

[20] K. Jell Gustafsson, M. Lundh, G. St, and) Derlind, “A PI STEPSIZE CONTROL FOR THE NUMERICAL SOLUTION OF ORDINARY DIFFERENTIAL EQUATIONS,” *BIT*, vol. 28, pp. 270–287, 1988.

[21] M. Saleh, E. Kovács, and I. F. Barna, “Analytical and Numerical Results for the Transient Diffusion Equation with Diffusion Coefficient Depending on Both Space and Time,” *Algorithms*, vol. 16, no. 4, p. 184, Mar. 2023, doi: 10.3390/a16040184.

[22] E. Kovács, Á. Nagy, and M. Saleh, “A New Stable, Explicit, Third-Order Method for Diffusion-Type Problems,” *Adv Theory Simul*, vol. 5, no. 6, p. 2100600, Jun. 2022, doi: 10.1002/adts.202100600.

[23] William H. Press, *Numerical Recipes The Art of Scientific Computing*, vol. 3. Cambridge University Press, 2007. [Online]. Available: www.cambridge.org/9780521880688

[24] K. E. Atkinson, W. Han, and D. Stewart, *Numerical Solution of Ordinary Differential Equations*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 2009. doi: 10.1002/9781118164495.