

Article

Unconditionally Dynamically Consistent Numerical Methods with Operator-Splitting for a Reaction-Diffusion Equation of Huxley's Type

Husniddin Khayrullaev and Endre Kovács * 

Institute of Physics and Electrical Engineering, University of Miskolc, 3515 Miskolc, Hungary

* Correspondence: endre.kovacs@uni-miskolc.hu

Abstract

The efficiency of various numerical methods for solving Huxley's equation—which includes a diffusion term and a nonlinear reaction term—is investigated. Conventional explicit finite difference algorithms often suffer from severe stability limitations and can yield unphysical concentration values. In this study, we collect a range of stable, explicit time integration methods of first to fourth order, originally developed for the diffusion equation, and design treatments of the nonlinear term which ensure that the solution remains within the physically meaningful unit interval. This property, called dynamical consistency, is analytically proven and implies unconditional stability. In addition to this, the most effective ones are identified from the large number of constructed method combinations. We conduct systematic tests in one and two spatial dimensions, also evaluating computational efficiency in terms of CPU time. Our results show that higher-order schemes are not always the most efficient: in certain parameter regimes, second-order methods can outperform their higher-order counterparts.

Keywords: diffusion-reaction equations; Huxley's equation; dynamical consistency; explicit time integration; stiff problems

MSC: 35K57; 65M06



Academic Editor: Xuemin Tu

Received: 5 August 2025

Revised: 22 August 2025

Accepted: 1 September 2025

Published: 3 September 2025

Citation: Khayrullaev, H.; Kovács, E. Unconditionally Dynamically Consistent Numerical Methods with Operator-Splitting for a Reaction-Diffusion Equation of Huxley's Type. *Mathematics* **2025**, *13*, 2848. <https://doi.org/10.3390/math13172848>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In this work, we study Huxley's equation [1] in the following form

$$\frac{\partial u}{\partial t} = \alpha \nabla^2 u + \beta u^2(1 - u) \quad (1)$$

where α and β are non-negative real numbers. The initial and boundary conditions will be described later in this paper. A slightly generalized form of (1), which is often called the Huxley equation [2], is

$$\frac{\partial u}{\partial t} = \alpha \nabla^2 u + \beta u(1 - u)(u - \gamma), \quad (2)$$

where $\gamma \in [0, 1]$, and we always take $\gamma = 0$ in this paper. Equation (2) is sometimes called the FitzHugh–Nagumo equation [3], but more general equations also appear under that name. These equations are members of the important family of nonlinear diffusion-reaction partial differential equations (PDEs) used to model various phenomena in biology and physics. For example, it was demonstrated [1] that equations with a cubic reaction term,

such as (1) or (2), are useful to describe the propagation of a new, useful recessive gene in a population which reproduces itself sexually. They are also suitable [4] to model nerve pulse propagation in biological tissues, where the equation describes the transmission of electrical signals along nerve fibers. We note that Equation (1) is similar to Fisher's equation

$$\frac{\partial u}{\partial t} = \alpha \nabla^2 u + \beta u(1 - u), \quad (3)$$

where the nonlinear term is only quadratic, and thus it is nonlinear to a lesser degree than Huxley's equation.

It is well known that the correct solution of the diffusion or heat equation always adheres to the minimum and maximum principles, which reflect the second law of thermodynamics. The minimum principle guarantees that the equation does not produce negative values if the initial and boundary values are non-negative, which is called the positivity-preserving property. When there is a reaction or other source term in the equation, these criteria are not always applicable. However, the reaction term in Equation (1) has the special property that its limit value is zero both in the $u \rightarrow 0$ and the $u \rightarrow 1$ cases. Therefore, a similar principle, which will be called dynamical consistency, is valid: if the initial and boundary values of the u variable are in the unit interval, then u remains in this interval for any non-negative values of the parameter β . We note that this property holds for Equation (2) for $\gamma \in [0, 1]$, but the dynamics of the time development strongly depend on the value of γ . For example, if γ is 0, then $u \equiv 1$ is the only stable steady state while $u \equiv 0$ is an unstable steady state. On the other hand, if $\gamma = 1$, then the cases are the other way around.

Unfortunately, the plausible requirement that the applied numerical schemes should preserve this quality of the solution is rarely guaranteed, with a very few exceptions, such as a few exact or nonstandard finite difference (NSFD) schemes [5]. Due to the discretization, the solutions produced by standard finite difference methods can contain meaningless (negative or too large) values, causing oscillations and even numerical instabilities. The widespread explicit methods have a severely limited stability region [6], even for the pure diffusion- or heat equation: the solution is expected to blow up if the time-step size exceeds the so-called CFL (Courant–Friedrichs–Lewy) limit. In the case of a heterogeneous media, one has a space-dependent α coefficient [7], and thus the CFL limit for the explicit methods is rather small, and the stiffness ratio is high. The problem of restricted stability persists in the case of the most frequent explicit methods, which work via the method of lines: the PDE is first spatially discretized, and then the obtained system of ordinary differential equations (ODEs) is solved by a well-established ODE solver. If this solver is the forward Euler method, then one has the FTCS (forward time central space) scheme, the stability limit of which will be used in this work to characterize the difficulty level of the problem. Higher-order methods, such as the strong stability-preserving Runge–Kutta (RK) methods [8,9], also suffer from stability issues.

Implicit methods, such as the so-called shifted airfoil collocation method applied by Anjuman et al. [10] for nonlinear drift-diffusion-reaction equations, have better stability properties, although sometimes step-size limitations can be required [11]. However, they demand the solution of a nonlinear algebraic equation system, which can be time- and memory-consuming if the size of the equation system is large. In spite of this, diffusion-reaction equations are often solved by implicit methods. Moreover, the diffusion term is sometimes even treated exactly [12] in the case of a stiff problem.

El-Danaf et al. solved the time-fractional Huxley equation by the variational iteration and the Adomian decomposition methods [13]. A semi-analytical reduced differential transform method (RDTM) [14] was utilized by Patel and Dhodiya. Bai et al. employed a

physics-informed neural network (PINN) and an improved PINN, called IPINN [15]. They observed that the latter one more accurately reproduces the different dynamic behaviors of the equation. However, this new method still has a limitation that, with the increase in the order and the increase in the nonlinear term of the PDEs, it cannot obtain the proper numerical solution of the PDEs. Manaa and Sabawi [16] used the explicit (Euler) and the Crank–Nicolson (CN) methods to solve Equation (2). They observed that the explicit method is faster, but as expected, the CN scheme is more accurate and does not suffer from serious stability issues. Dynamically consistent or at least unconditionally positive methods to solve these or similar equations are investigated by a few scholars [17–19]. Macías-Díaz constructed a finite-difference numerical scheme for the Burgers–Huxley equation with dynamical consistency [20]. However, he used the explicit Euler time-discretization for the diffusion term, and thus his method possesses good properties only for small time-step sizes below the CFL number. The same is true for the positivity-preserving scheme proposed by Chertock and Kurganov [21] for advection-diffusion-reaction equation-system modeling chemotaxis/haptotaxis problems, the nonstandard finite difference schemes (NSFDs) of the group of Appadu [3,22,23] for the Fisher and Nagumo equations, and that of Songolo [24] and Chapwanya et al. [25] for cross-diffusion equations.

Our goal in this work is to investigate numerical schemes which are unconditionally stable and dynamically consistent. A similar study, but with a smaller number of methods, has been carried over in our recent work [26] for Fisher’s equation. We also constructed efficient methods for Huxley’s equation [27], but none of them had the property of dynamical consistency. For us, the word unconditional always means that the desirable properties hold for arbitrary non-negative coefficients α and β , arbitrary positive time-step sizes, and arbitrary mesh for space discretization. Therefore, in Section 2, the spatial discretization of Equation (1) is considered not only in the simplest, one-dimensional case, but in a very general case as well. Since operator-splitting is applied, first the treatment of the diffusion term and then the nonlinear reaction term is discussed. In Section 3, we first briefly recall 13 numerical algorithms which, when applied to the pure diffusion equation, follow the maximum and minimum principle. Then, in Section 4, five different treatments of the reaction term are constructed, and the dynamical consistency property, valid for all of the $5 \times 13 = 65$ algorithm combinations, is analytically proved. In Section 5, we perform the verification using an analytical reference solution. Based on this, we immediately exclude one of the five treatments of the Huxley term from further examination. After this, via the Strang-splitting technique, we construct further versions of the methods, and we test the obtained algorithm combinations. The results of these tests help us to select the 28 combinations which we think are worth testing further. In Section 6, the performance of these 28 combinations with running-time measurements is carried out for two different two-dimensional systems to select the top nine methods. The last three further numerical case studies in Section 7 use only these nine combinations. There are two parameter sweeps for the nonlinear coefficient β (with a stiff and a non-stiff system) and one sweep for the stiffness ratio. The final conclusions and recommendations are summarized in Section 8.

2. The Spatial Discretization of the Diffusion Equation Using Uniform and Non-Uniform Meshes

In this section, the pure diffusion equation is discretized. First, the simple equidistant discretization $x_i = x_0 + i\Delta x$, $i = 1, \dots, N$, $N\Delta x = L$ of the interval $x \in [x_0, x_N = x_0 + L] \subset \mathbb{R}$ is constructed. The well-known central difference approximation is applied to the second spatial derivatives leading to the ODE system

$$\frac{du_i}{dt} = \frac{\alpha}{\Delta x^2} (u_{i-1} - 2u_i + u_{i+1}) \quad (4)$$

for the nodes $i = 1, \dots, N - 1$. A matrix denoted by M is defined as containing a lot of zeros and the following nonzero elements:

$$m_{ii} = -\frac{2\alpha}{\Delta x^2} (1 < i < N), \quad m_{i,i+1} = \frac{\alpha}{\Delta x^2} (1 \leq i < N), \quad m_{i,i-1} = \frac{\alpha}{\Delta x^2} (1 < i \leq N). \quad (5)$$

In the one-dimensional case, M is tridiagonal. The first and the last rows depend on the boundary conditions (BCs). For example, the time development of the first and the last nodes will be determined by the analytical solution in Section 5 in the case of Dirichlet BCs. Thus, the appropriate elements of M can be set to zero since they will not be used. The ODE system (4) may be rephrased in the following matrix form:

$$\frac{dU}{dt} = MU, \quad (6)$$

where $U = (u_0, \dots, u_N)^T$. We are going to deal with more general problems where the material and geometrical properties of the studied system have a space dependence. The space discretization, and later, the methods should be extended to reflect this level of generality. If α (which is a property of the media) depends on the space variable, but we are still in 1D, the PDE

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x} \left(\alpha(x) \frac{\partial u}{\partial x} \right)$$

can be used. In this equation, it is possible to simultaneously discretize the function α and $\partial u / \partial x$, which gives

$$\left. \frac{\partial u}{\partial t} \right|_{x_i} = \frac{1}{\Delta x} \left[\alpha \left(x_i - \frac{\Delta x}{2} \right) \frac{u(x_i - \Delta x) - u(x_i)}{\Delta x} + \alpha \left(x_i + \frac{\Delta x}{2} \right) \frac{u(x_i + \Delta x) - u(x_i)}{\Delta x} \right].$$

The average diffusivity between cell i and its (left) neighbor may be denoted by $\alpha_{i,i-1}$, which, in practice, can be approximated by its value between these two nodes. With this, we have

$$\frac{du_i}{dt} = \frac{1}{\Delta x} \left(\alpha_{i,i+1}^n \frac{u_{i+1} - u_i}{\Delta x} + \alpha_{i,i-1}^n \frac{u_{i-1} - u_i}{\Delta x} \right),$$

which is a generalized form of Equation (2). Our goal is to introduce a very flexible resistance–capacitance model by considering cells instead of nodes. These cells have finite volumes, which can be given as $V_i = \Delta x$ in the one-dimensional equidistant case. We have to interject that our methods are not finite volume methods (FVMs), but they are much closer to the finite difference methods. In the case of the simplest diffusion, the capacity of each cell equals its volume: $C_i = V_i = \Delta x$. The resistance between two cells can be given simply as $R_{i,i+1} = R_{i+1,i} = \frac{\Delta x}{\alpha_{i,i+1}}$, $i = 1, \dots, N - 1$. Using these, one can obtain the ODE system for the time derivative of each cell variable u in one spatial dimension:

$$\frac{du_i}{dt} = \frac{u_{i-1} - u_i}{R_{i,i-1}C_i} + \frac{u_{i+1} - u_i}{R_{i,i+1}C_i}. \quad (7)$$

Equation (7) can be written in the same matrix form as (6), where the non-diagonal elements have the form $m_{i,i-1} = \frac{1}{R_{i,i-1}C_i}$ and $m_{i,i+1} = \frac{1}{R_{i,i+1}C_i}$, while the diagonal elements are the negative sum of the non-diagonal ones: $m_{i,i} = -\frac{1}{R_{i,i-1}C_i} - \frac{1}{R_{i,i+1}C_i}$.

We will also work in two spatial dimensions, where the numbering of the cells begins along the x -directional (horizontal) rows running from 1 to N_x , then along the y axis

again from $N_x + 1$ to $2N_x$, and so on. The ODE system (7) can now be straightforwardly generalized to the case with two spatial dimensions:

$$\frac{du_i}{dt} = \frac{u_{i-1} - u_i}{R_{i,i-1}C_i} + \frac{u_{i+1} - u_i}{R_{i,i+1}C_i} + \frac{u_{i-N_x} - u_i}{R_{i,i-N_x}C_i} + \frac{u_{i+N_x} - u_i}{R_{i,i+N_x}C_i} = \sum_{j \neq i} \frac{u_j - u_i}{R_{ij}C_i}. \quad (8)$$

One can find further details on the discretization via this resistance–capacitance model, e.g., in [28,29].

When performing numerical experiments in 2D, we always consider zero Neumann BCs. These will be implemented in the RC model very simply: the appropriate resistances are considered to be infinity, which means that the matrix elements responsible for conduction across the boundaries vanish. In this case, the system matrix M has a zero eigenvalue corresponding to the conserved quantity. All other eigenvalues are negative, which is true without exception for the case of zero Dirichlet BCs. Let us denote by λ_{MIN} and λ_{MAX} the eigenvalues with the (nonzero) smallest and largest absolute values, respectively. The stiffness ratio of the problem is usually defined as $SR = \lambda_{MAX}/\lambda_{MIN}$. On the other hand, the CFL limit $h_{MAX}^{RK1} = |2/\lambda_{MAX}|$ accurately gives the maximum time-step size that the explicit Euler (first-order explicit RK) scheme can employ for the pure (linear) diffusion problem (4), (7), or (8). From stability polynomial analysis, one can obtain similar time-step size thresholds for higher-order explicit RK methods. For example, for the second-, third-, and fourth-order RK schemes, one has $h_{MAX}^{RK2} = h_{MAX}^{RK1}$, $h_{MAX}^{RK3} \approx 1.256 \cdot h_{MAX}^{RK1}$, and $h_{MAX}^{RK4} \approx 1.392 \cdot h_{MAX}^{RK1}$, respectively. If the user exceeds these limits, the solutions blow up due to the instability. The two numbers SR and h_{MAX}^{RK1} characterize the degree of difficulty of the linear part of the problem. We have to stress once more that the unconditionally stable methods used in this work have no CFL limits, as we will see soon.

3. The 13 Examined Methods for the Diffusion Equation with the Convex-Combination Property

The numerical algorithms for the linear diffusion equation are briefly described here, and the treatments of the Huxley term will be presented in the next section. The time variable is always discretized uniformly using the following notations:

$$t^n = t^{\text{ini}} + nh, \quad t \in [t^{\text{ini}}, t^{\text{fin}}], \quad n = 1, \dots, T, \quad hT = t^{\text{fin}} - t^{\text{ini}}$$

In the special case of an equidistant one-dimensional mesh with constant diffusivity α , the mesh ratio is usually defined as $r = \frac{\alpha h}{\Delta x^2} = -\frac{m_{ij}h}{2} > 0$, $0 < i < N - 1$. Since the general case of space-dependent α and non-uniform mesh contains this special case, only the generalized formula of the numerical algorithms is presented. Let us define the quantity

$$r_i = h \sum_{j \neq i} \frac{1}{C_i R_{ij}} = -hm_{ii}, \quad (9)$$

which is the generalization of the mesh ratio r , since, in the 1D special case, $r_i = 2r$. We also introduce a quantity to collect the effect of the neighbors of the cell with index i as follows:

$$A_i = h \sum_{j \neq i} \frac{u_j^n}{C_i R_{ij}} = h \sum_{j \neq i} m_{ij} u_j^n$$

In the special 1D case, $A_i = r(u_{i-1}^n + u_{i+1}^n)$, hence $\frac{A_i}{r_i} = \frac{u_{i-1}^n + u_{i+1}^n}{2}$. Using these relations, one can easily obtain the formulas applicable in the 1D special case from their generalized variants. The application of the methods is very similar to that of the standard

explicit Euler or RK methods, with only the formulas used being different. For the two exceptions, we will display a flowchart to make understanding easier.

1. The simplest method we use is called the unconditionally positive finite difference (UPFD) scheme. This was introduced by Chen-Charpentier and Kojouharov [17] for the linear diffusion-advection-reaction equation discretized by an equidistant mesh. We do not have the advection term, and we adapt the formula to our general Equations (7) and (8) as follows:

$$u_i^{n+1} = \frac{u_i^n + A_i}{1 + r_i} \quad (10)$$

2. In the case of the CNe, or constant neighbor method [30], the positive r_i quantities containing the time-step size are in the exponent with a negative sign:

$$u_i^{n+1} = u_i^n \cdot e^{-r_i} + \frac{A_i}{r_i} (1 - e^{-r_i}) \quad (11)$$

This ensures the outstanding stability properties of the methods built on this CNe formula.

In this work, we test two different odd–even hopscotch-type algorithms. To do this, we need a special checkerboard-like characterization of the cells, where the neighbors of the cells with the label ‘odd’ are labeled as ‘even’ and vice versa. The time-marching for the u values goes alternately for the set of the odd and the even cells. When one set is calculated, the latest values of the neighbors $u_{i\pm 1}$, $u_{i\pm Nx}$ belonging to the other set are always used in the formula for A_i . We note that only one function evaluation is required per cell per time step, so these solvers are quite fast.

3. OOEH-CNe method: The original odd–even hopscotch time-space structure is applied, as one can see in the flowchart in Figure 1. It means that the odd values are refreshed first, and then the even values. However, after each time step, the odd–even labels are interchanged. The length of the time steps is always the full h , and the CNe formula (11) is employed in each stage.

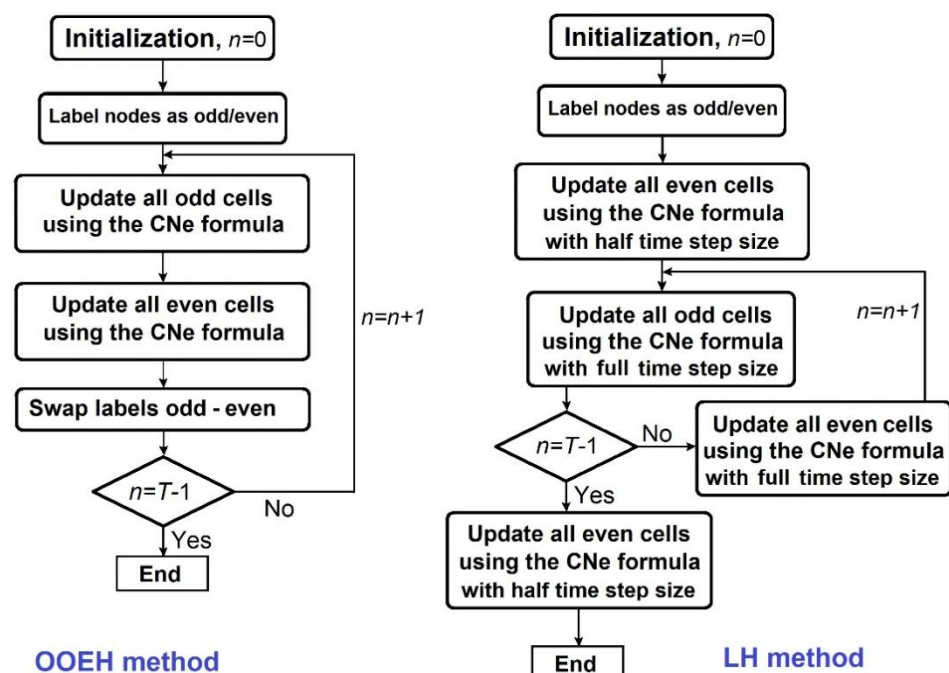


Figure 1. Flowchart for the OOEH and LH methods.

4. The LH-CNe algorithm: The leapfrog–hopscotch (LH) structure contains time steps with length not only with integer h , but with $h/2$. The solution begins with a half time step for, e.g., the odd cells, after which full-length time steps are allowed for the set of the even and the odd cells, strictly alternately. This goes until we reach the last time step, which must have the length of $h/2$ for the set of the odd cells again to precisely reach the final time t^{fin} . The LH-CNe method investigated in this paper always employs the CNe formula (11) with the appropriate full or half time-step size.

5. The CpC method [29] first calculates new predictor u values applying the CNe formula (11), but with a halved time step:

$$u_i^{\text{pred}} = u_i^n e^{-r_i/2} + \frac{A_i}{r_i} (1 - e^{-r_i/2}) \quad (12)$$

These predictor values are used to obtain the new

$$A_i^{\text{pred}} = \Delta t \sum_{j \neq i} \frac{u_j^{\text{pred}}}{C_i R_{ij}} \quad (13)$$

quantities. Then the second, corrector stage constitutes a full time step with (11) again:

$$u_i^{n+1} = u_i^n \cdot e^{-r_i} + \frac{A_i^{\text{pred}}}{r_i} (1 - e^{-r_i}) \quad (14)$$

6. The LNe (or sometimes LNe2, meaning two-stage linear-neighbor) algorithm is a predictor–corrector method [30], too. The first stage uses the above given CNe formula with full time-step size, and it gives the new u_i^{pred} predictor values. Using these, one can use Equation (13) and then calculate the corrector values:

$$u_i^{n+1} = u_i^n e^{-r_i} + \left(A_i - \frac{A_i^{\text{pred}} - A_i}{r_i} \right) \frac{1 - e^{-r_i}}{r_i} + \frac{A_i^{\text{pred}} - A_i}{r_i} \quad (15)$$

7-8-9. The corrector values given by Equation (15) are used to calculate A_i^{pred} with (13) again, and thus one repeats (15) to obtain refined u values. This increases the total number of stages in one time step to three, and hence, the seventh algorithm is called LNe3 [30]. This iteration can be continued to obtain the LNe4 (LNe5) method, which has four (five) stages altogether. These are obtained by repeating (13) and (15). These iterations, although they yield slightly larger accuracy, unfortunately do not improve the order.

10. The three-stage constant-linear-quadratic neighbor (CLQ) method [31] is a bit more complicated than the previous ones, since it also employs the quadratic-neighbor approximation. Its first two stages are similar to those of the LNe algorithm: A_i is calculated at the beginning of the first stage, and then one full-size time step with (10) is performed. After using (12), the LNe stage (15) comes with a full as well as a half-length time step, to obtain the sets u_i^L and $u_i^{L\frac{1}{2}}$, respectively. Using these, one can repeat Equation (13) to calculate $A_i^{\text{pred,L}}$ and $A_i^{\text{pred,L}\frac{1}{2}}$, both without dividing h by 2 in (13). Two new quantities, $S_i = 4A_i^{\text{pred,L}\frac{1}{2}} - A_i^{\text{pred,L}} - 3A_i$ and $W_i = 2\left(A_i^{\text{pred,L}} - 2A_i^{\text{pred,L}\frac{1}{2}} + A_i\right)$, must then be calculated. The new u values are given by

$$u_i^Q = e^{-r_i} u_i^n + \frac{1 - e^{-r_i}}{r_i} \left(\frac{2W_i}{r_i^2} - \frac{S_i}{r_i} + A_i \right) + \frac{(1 - 2/r_i)W_i + S_i}{r_i} \quad (16)$$

11-12-13. We can add one more stage by using the values in (16). In this case, besides (16) at the third stage, the midpoint values must also be calculated by a similar formula:

$$u_i^{Q\frac{1}{2}} = e^{-r_i/2} u_i^n + \frac{1 - e^{-r_i/2}}{r_i} \left(\frac{2W_i}{r_i^2} - \frac{S_i}{r_i} + A_i \right) + \frac{W_i}{4r_i} - \frac{W_i}{r_i^2} + \frac{S_i}{2r_i} \quad (17)$$

Using these, one can calculate $A_i^{\text{pred},Q}$ and $A_i^{\text{pred},Q\frac{1}{2}}$ values by (12) at the fourth stage, with which one repeats the calculations of the third stage, including those of S and W . With these, one has four stages altogether, which constitute the CLQ2 algorithm. One can further repeat this iteration by re-calculating $A_i^{\text{pred},Q}$ and $A_i^{\text{pred},Q\frac{1}{2}}$, then applying (16) and (17) again, etc. If one has three or four quadratic stages (five or six stages altogether), we call the obtained methods CLQ3 and CLQ4, respectively [31].

Properties of the methods when solving the space-discretized diffusion equation:

The orders of temporal accuracy are the following: one for the UPFD and the CNe, two for the OOEH-CNe, LH-CNe, CpC, and LNe-LNe5, three for the CLQ, and four for the CLQ2-4. All of these algorithms are unconditionally stable when applied to the linear diffusion equation; therefore, the previously mentioned CFL limit cannot confine their stability. This outstanding stability comes from the following property.

Theorem 1. *If methods 1–13 solve the spatially discretized linear diffusion Equation (4), then the new u_i^n values are the convex combinations of the initial values u_i^0 . The statement is true in the case of the general Equation (7) or (8), when the 1–9 methods are applied.*

These statements have been analytically proved in the original papers. According to all of the numerical tests, the convex combination property holds when the CLQ-CLQ4 algorithms solve the generalized Equations (7) and (8), but we have not been able to prove this analytically. Furthermore, the truncation errors of the methods for the linear diffusion equation are calculated in our previous paper [32].

4. The Operator-Splitting Treatments of the Nonlinear Term

During a time step, we independently take into account the effect of the linear diffusion term and the nonlinear Huxley term. The diffusion term is treated by the 13 methods listed in the previous subsection. After fully considering the diffusion term by those schemes, we obtain a concentration value u_i^{diff} , which we temporarily denote as p for brevity. The reaction term is local and can be written into the form $\beta u^2 - \beta u^3$. A very efficient way to take into account this nonlinear term is to solve the initial value problem

$$\begin{cases} \frac{du}{dt} = \beta u^2 - \beta u^3, \\ u(t=0) = u_i^{\text{diff}} = p, \quad t \in [0, h] \end{cases} \quad (18)$$

analytically, as in [26]. However, we were not able to solve this, even with the help of Maple 18 software. Hence, we choose an approach which starts with a selective replacement of u by the p values in all twelve possible ways:

$$\begin{array}{lll} \beta u^2 - \beta u^3 & \beta u p - \beta u^3 & \beta p^2 - \beta u^3 \\ \beta u^2 - \beta u^2 p & \beta u p - \beta u^2 p & \beta p^2 - \beta u^2 p \\ \beta u^2 - \beta u p^2 & \beta u p - \beta u p^2 & \beta p^2 - \beta u p^2 \\ \beta u^2 - \beta p^3 & \beta u p - \beta p^3 & \beta p^2 - \beta p^3 \end{array} \quad (19)$$

Three of these ways, namely, $\beta u p - \beta u p^2$, $\beta u p - \beta p^3$, and $\beta p^2 - \beta u p^2$, can be considered as a kind of linearization, while the last one without u is the equivalent to the explicit

Euler treatment. After inserting each of the expressions (19) into the right-hand side of (18), we try to solve the obtained initial value problems, for example,

$$\begin{cases} \frac{du}{dt} = \beta pu - \beta pu^2, \\ u(t=0) = u_i^{\text{diff}} = p, \quad t \in [0, h] \end{cases}$$

again analytically. If the solution is found, we first check whether the obtained analytical solutions guarantee dynamical consistency by plotting it as a function of p and $b := \beta h$. We observed that there are two cases when there is no analytical solution, six cases (including the linearization $\beta up - \beta up^2$) where the solution exists but it can be larger than one, and four cases when the solution is in the unit interval. These latter four cases will be used to treat the nonlinear term, and they are listed in the following Table 1.

We have an extra treatment, where the effect of the Huxley term is calculated using an approach which we call the “pseudo-implicit” trick [31]. In this case, we replace u on the right-hand side of (17) with not only the u_i^{diff} values but also the new, unknown value u_i^{n+1} with the goal of eliminating the negative terms from the right-hand side of this formula. In this way, we obtain

$$\frac{u_i^{n+1} - u_i^{\text{diff}}}{h} = \beta (u_i^{\text{diff}})^2 - \beta (u_i^{\text{diff}})^2 u_i^{n+1} \quad (24)$$

Table 1. Four treatments of the nonlinear term.

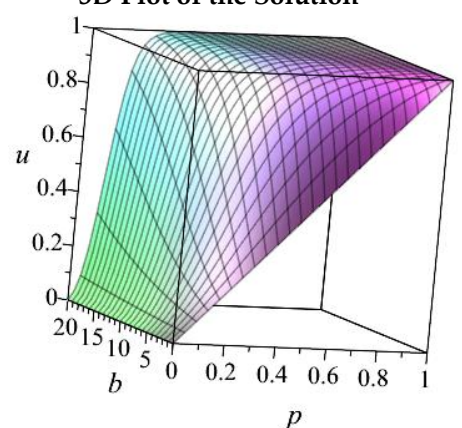
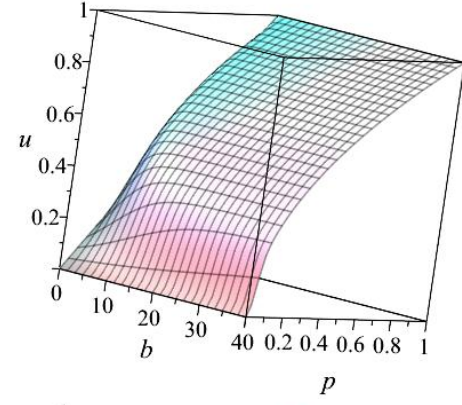
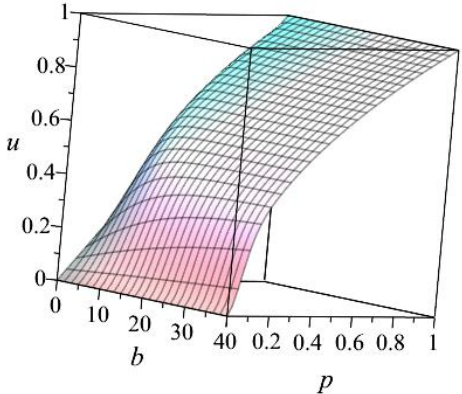
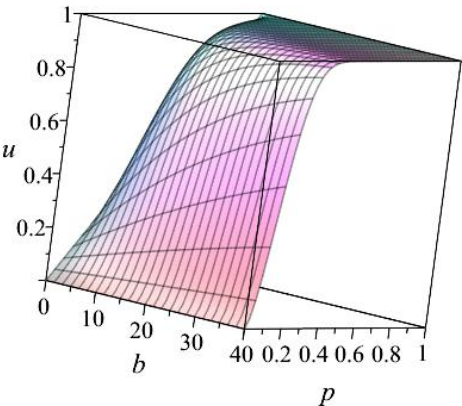
Treatment	Substitution	Solution	3D Plot of the Solution
t1	$\beta up - \beta u^2 p$	$u(h) = \frac{p}{p + (1-p)e^{-pb}} \quad \text{or} \quad u_i^{n+1} = \frac{u_i^{\text{diff}}}{u_i^{\text{diff}} + (1 - u_i^{\text{diff}})e^{-u_i^{\text{diff}} \beta t}} \quad (20)$	3D Plot of the Solution 
t2	$\beta up - \beta u^3$	$u(h) = \frac{p}{\sqrt{p + (1-p)e^{-2pb}}} \quad \text{or} \quad u_i^{n+1} = \frac{u_i^{\text{diff}}}{\sqrt{u_i^{\text{diff}} + (1 - u_i^{\text{diff}})e^{-2u_i^{\text{diff}} \beta t}}} \quad (21)$	

Table 1. Cont.

Treatment	Substitution	Solution	3D Plot of the Solution
t3	$\beta p^2 - \beta u^2 p$	$u(h) = \sqrt{p} \tanh\left(bp^{3/2} + \operatorname{arctanh}(\sqrt{p})\right)$ or $u_i^{n+1} = \sqrt{u_i^{\text{diff}}} \tanh\left(\beta t \left(u_i^{\text{diff}}\right)^{3/2} + \operatorname{arctanh}\left(\sqrt{u_i^{\text{diff}}}\right)\right) \quad (22)$	
t4	$\beta p^2 - \beta u p^2$	$u(h) = 1 - (1 - p)e^{-p^2 b}$ or $u_i^{n+1} = 1 - \left(1 - u_i^{\text{diff}}\right)e^{-(u_i^{\text{diff}})^2 \beta t} \quad (23)$	

This can be simply rearranged to take on a completely explicit form:

$$u_i^{n+1} = \frac{1 + \beta h u_i^{\text{diff}}}{1 + \beta h (u_i^{\text{diff}})^2} u_i^{\text{diff}}. \quad (25)$$

This will be the fifth treatment of the Huxley term, denoted by PI. Using the short, temporary notations, it has the form

$$u(h) = \frac{1 + bp}{1 + bp^2} p.$$

Lemma 1. If $0 \leq p \leq 1$, then $p \leq u \leq 1$ holds for the following five different $u(h)$ functions

1. $u = \frac{p}{p + (1 - p)e^{-pb}}$
2. $u = \frac{p}{\sqrt{p + (1 - p)e^{-2pb}}}$
3. $u = \sqrt{p} \tanh\left(bp^{3/2} + \operatorname{arctanh}(\sqrt{p})\right)$
4. $u = 1 - (1 - p)e^{-p^2 b}$
5. $u = \frac{1 + bp}{1 + bp^2} p$

for arbitrary $b > 0$. Furthermore, if $b = 0$, then $p = u$, and u is monotonously increasing with increasing b .

Proof. We clearly have $0 \leq p^2 \leq p \leq \sqrt{p} \leq 1$, $0 \leq 1 - p \leq 1$, and $0 \leq e^{-2pb} \leq e^{-pb} \leq 1$, which will be used below, when the five functions are individually examined, as follows.

1. The term $(1 - p)e^{-pb}$ is non-negative, and thus the denominator $p + (1 - p)e^{-pb}$ cannot be smaller than p ; hence, $u \leq 1$. On the other hand, the denominator is majorated by unity:

$$p + (1 - p)e^{-pb} = e^{-pb} + p(1 - e^{-pb}) \leq e^{-pb} + (1 - e^{-pb}) = 1.$$

Therefore, the denominator cannot be larger than one, and thus $p \leq u$.

2. Since $\sqrt{p} \geq p$, the same argument as in point 1 can be repeated for this denominator $\sqrt{p + (1 - p)e^{-2pb}}$ as well.

3. Both $\sqrt{p}$ and the tanh function are majorated by one, and thus also their product; hence, $u \leq 1$. Moreover, the tanh is a monotonously increasing function, and thus $\sqrt{p} \cdot \tanh(bp^{3/2} + \operatorname{arctanh}(\sqrt{p})) \geq \sqrt{p} \cdot \tanh(\operatorname{arctanh}(\sqrt{p})) = \sqrt{p} \cdot \sqrt{p} = p$, which implies $p \leq u$.

4. Since $(1 - p)e^{-p^2b}$ is non-negative, $u \leq 1$. On the other hand,

$$u = 1 - (1 - p)e^{-p^2b} = pe^{-p^2b} + (1 - e^{-p^2b}) \geq pe^{-p^2b} + p(1 - e^{-p^2b}) = p,$$

and thus, we obtain $p \leq u \leq 1$.

5. The denominator $1 + bp^2$ cannot be greater than the numerator $1 + bp$, and thus $\frac{1+bp}{1+bp^2} \geq 1 \Rightarrow u \geq p$. Additionally,

$$1 + bp^2 \geq p + bp^2 \Rightarrow u \leq 1.$$

The last statements of the lemma can be immediately obtained by the $b = 0$ substitution. $\square$

Simple operator-splitting means that a time step starts with the treatment of the diffusion by any of the algorithms displayed in Section 3, and after that, the treatment of the nonlinear term following one of the five ways above is conducted as a substitutional step. On the other hand, Strang-splitting means that one of the operations (20)–(25) is performed once before and a second time after the calculation of the effect of diffusion, both with halved time-step sizes. This means that the substitution $h \rightarrow h/2$ must be performed in (20)–(25). Note that in the case of Strang-splitting, the first and the second formulas for the nonlinear term are not necessarily the same; i.e., the treatments (20)–(25) can be hybridized. For example, it is possible to perform (20) with a halved h before and (22) after any of the methods 1–13 for the diffusion term.

Theorem 2. (The methods' dynamical consistency) Suppose one applies Methods 1–13 combined with treatments (20)–(25) with or without Strang-splitting to Huxley's Equation (1). If the initial and boundary values of the concentration are in the unit interval $u_i^0, u_0^n, u_N^n \in [0, 1]$, then the calculated concentration values will also be in this interval, i.e., $u_i^n \in [0, 1]$ for any values of the coefficients $\alpha > 0$, $\beta > 0$ and time-step size $h > 0$. Furthermore, increasing β can yield only the increase, and never the decrease, of the concentration values.

Proof of Theorem 2. Due to the operator-splitting strategy, the procedure of the numerical solution consists of performing steps which treat the diffusion term and the non-

linear term subsequently. Theorem 1 and Lemma 1 imply that none of these steps can yield any concentration value outside of the unit interval, which immediately gives the first statement.

The last statement of Lemma 1 means that increasing β can yield only the increase in the concentration values calculated by any of the nonlinear steps given in Table 1. The convex combination property in Theorem 1 means that the steps which treat the diffusion term calculate linear combinations of the u_i^n values with coefficients in the unit interval $[0, 1]$. The non-negativity of these coefficients implies that if the u values are increased in the nonlinear step due to increasing β , that can yield only increased values after the diffusion step as well. $\square$

The first statement of Theorem 2 is analogous to the maximum and minimum principles being valid for the pure (linear) diffusion equation.

Corollary 1. *If the conditions of Theorem 2 hold, the numerical solution is unconditionally stable, since the difference between the true and the numerical solutions cannot be larger than 1.*

We note that a small fraction of the statements of Theorem 2 have been published in our paper [31].

5. Verification in 1D

Now PDE (1) with $\alpha = 1$ is going to be solved in the domain $(x, t) \in [x_0, x_N] \times [t^{\text{ini}}, t^{\text{fin}}]$. The analytical solution [1] (p. 34)

$$u^{\text{exact}}(x, t) = \frac{e^{\sqrt{\frac{\beta}{2}}x + \frac{\beta}{2}t}}{e^{\sqrt{\frac{\beta}{2}}x + \frac{\beta}{2}t} + c} \quad (26)$$

is used as the reference. The initial and the Dirichlet boundary conditions are obtained by evaluating this function at the time and space points belonging to the initial time $u(x, 0)$, and with the boundaries $u(x_0, t)$, $u(x_N, t)$, respectively. The maximum numerical errors are calculated by comparing the numerical solutions u_j^{num} produced using the examined method with the analytical reference solution u_j^{ref} at the final time t^{fin} using the formula

$$\text{Error}(L_\infty) = \max_{0 \leq j \leq N} |u_j^{\text{ref}}(t^{\text{fin}}) - u_j^{\text{num}}(t^{\text{fin}})|. \quad (27)$$

The average error is very similar:

$$\text{Error}(L_1) = \frac{1}{N} \sum_{0 \leq j \leq N} |u_j^{\text{ref}}(t^{\text{fin}}) - u_j^{\text{num}}(t^{\text{fin}})|.$$

The third type of error has an energy dimension in the case of the heat conduction equation, and hence we call it the energy error:

$$\text{Error}(\text{Energy}) = \sum_{1 \leq j \leq N} C_j |u_j^{\text{ref}}(t^{\text{fin}}) - u_j^{\text{num}}(t^{\text{fin}})|.$$

We would like to assess the error not only for a specific time-step size but also for a wide range of time-step sizes. Hence, we start using the maximum possible time-step size $h = t^{\text{fin}} - t^{\text{ini}}$, calculate the errors defined above, and then repeat this calculation for subsequently decreasing time-step sizes. The time-step size is always divided by two, and the total number of different time-step sizes is denoted by P . We calculate the average of

the logarithm of the obtained errors, which will be called the aggregated error (AgE). In the case of the maximum errors, it is given by the following formula

$$\text{AgE}(L_\infty) = \frac{1}{P} \sum_{k=1}^P \log(\text{Error}(L_\infty)), \quad (28)$$

where the summation is going for the different time-step sizes. The final step is to calculate the average of the three different AgE errors:

$$\text{AgE} = \frac{1}{3} (\text{AgE}(L_\infty) + \text{AgE}(L_1) + \text{AgE}(\text{Energy})). \quad (29)$$

Note that when the AgE is negative with a large absolute value, the method is very accurate. In this section, we calculated the aggregated error for $P = 13$ different time-step sizes for all the studied methods.

We perform two case studies using the analytical solution (26), with completely different parameter values.

Case Study 1: Medium nonlinear coefficient, equidistant mesh

PDE (1) is discretized on the $x_0 = 0$, $x_{N=200} = 4$, $t^{\text{ini}} = 0.00$, $t^{\text{fin}} = 1.3$ domain using $\Delta x = 0.02$ space step size. The nonlinear coefficient is $\beta = 4.3$, while $c = 1$ in Equation (26).

Case Study 2: Large nonlinear coefficient, non-equidistant mesh

The equation is solved on the $[0, 1.4416]$ interval with $t^{\text{ini}} = 0.1$, $t^{\text{fin}} = 0.29$. A non-equidistant spatial mesh is used with slowly increasing cell diameters from $\Delta x_1 = 0.002$ to $\Delta x_{400} = 0.0052$. The nonlinear coefficient is quite large, $\beta = 23.7$, and $c = 3.6$.

In order to verify the algorithm combinations, we set $P = 18$, and plot the errors as a function of the time-step size. However, instead of displaying the $5 \times 13 = 65$ curves for each case study, we select those using the PI reaction-term treatment in Case Study 1 with all the diffusion-solvers in Figure 2, while choosing three different diffusion-solvers combined with all the reaction-term treatments in Figure 3. Very similarly, for Case Study 2, we present the error curves for all diffusion-solvers using the t2 reaction-treatment, and then the five reaction-treatments combined with only three diffusion-solvers, in Figures 4 and 5, respectively.

The numerical order of convergence has been calculated for the two cases displayed in Figures 2 and 4 and presented in Tables 2 and 3, respectively. For this, we chose two values of the errors belonging to two subsequent time-step sizes using the formula

$$\text{Error} \sim h^{\text{NO}} \Rightarrow \text{NO} = \frac{\log \frac{\text{Error}(L_\infty)(h_k)}{\text{Error}(L_\infty)(h_{k+1})}}{\log(h_k/h_{k+1})} = \frac{\log \frac{\text{Error}(L_\infty)(h_k)}{\text{Error}(L_\infty)(h_{k+1})}}{\log 2}.$$

The values of the first time-step size h_k and the subsequent time-step size $h_{k+1} = h_k/2$ have been chosen in such a way as to obtain the maximal value for NO. The exception is the first (and largest) time-step size, where the observed significant drop of the error does not reflect the order of the methods (see the top-right corner of Figures 2 and 4), and hence this $k = 1$ case is not sampled. One can observe that the numerical orders are not far from the theoretical orders which were calculated for the pure diffusion equation.

Very similar error curves have been obtained for other values of the parameters β , t^{fin} , Δx and for the remaining algorithm combinations as well. The methods converge smoothly, and unstable behavior or unphysical oscillations are not produced. The only exception is treatment 3, which does not converge; therefore, we omit it from further investigations. The remaining four reaction-treatments are organized with halved time steps by Strang-splitting as well. In this way, 12 treatments are constructed and tested with the 13 diffusion-solvers, i.e., $12 \times 13 = 156$ algorithm combinations are tested. Consider-

ing that we have two test problems and $P = 13$ different time-step sizes, we performed 4056 runs with error calculations. The AgE values are tabulated in Tables 4 and 5 and visualized in Figures 6 and 7 for Case Study 1 and 2, respectively. We note that the errors of the different treatments are sometimes very close to each other, so the difference between them is only in the third or fourth decimals.

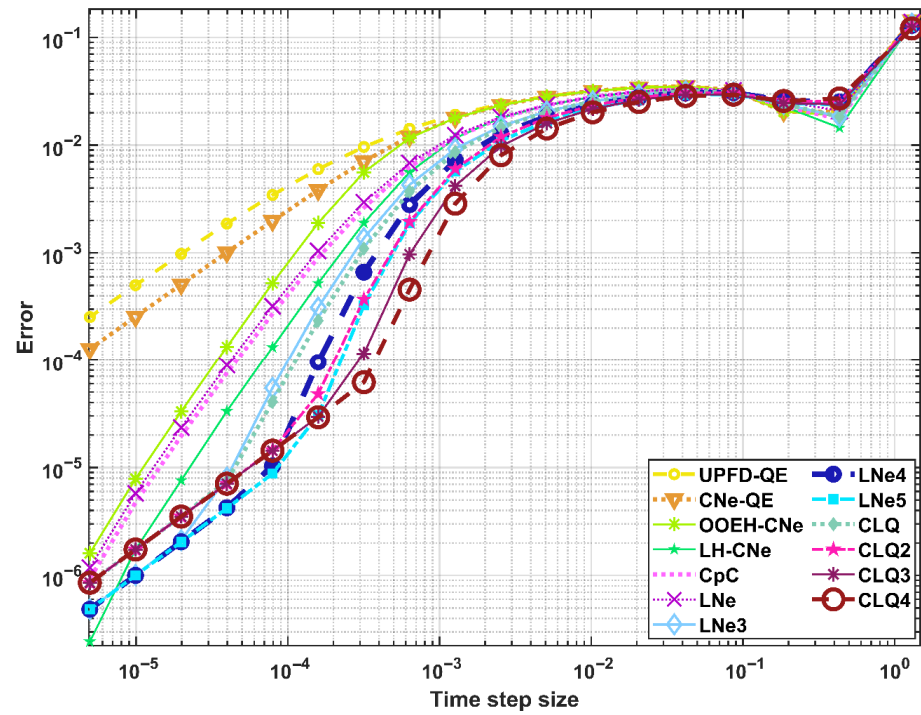


Figure 2. Maximum errors as a function of the time-step size for the PI treatment in Case Study 1.

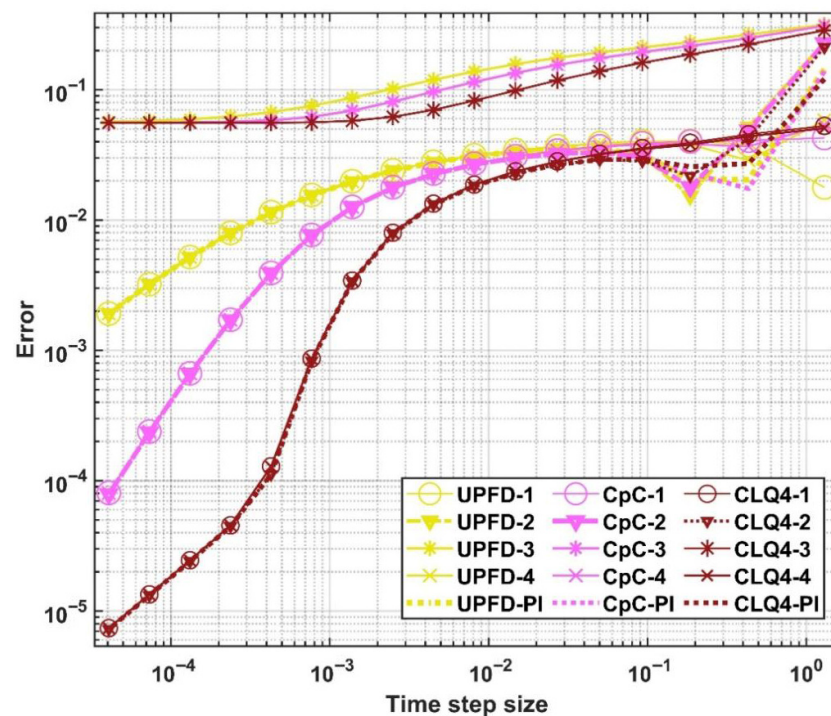


Figure 3. Maximum errors as a function of the time-step size for the five reaction-treatments in the case of three diffusion-solver schemes in Case Study 1.

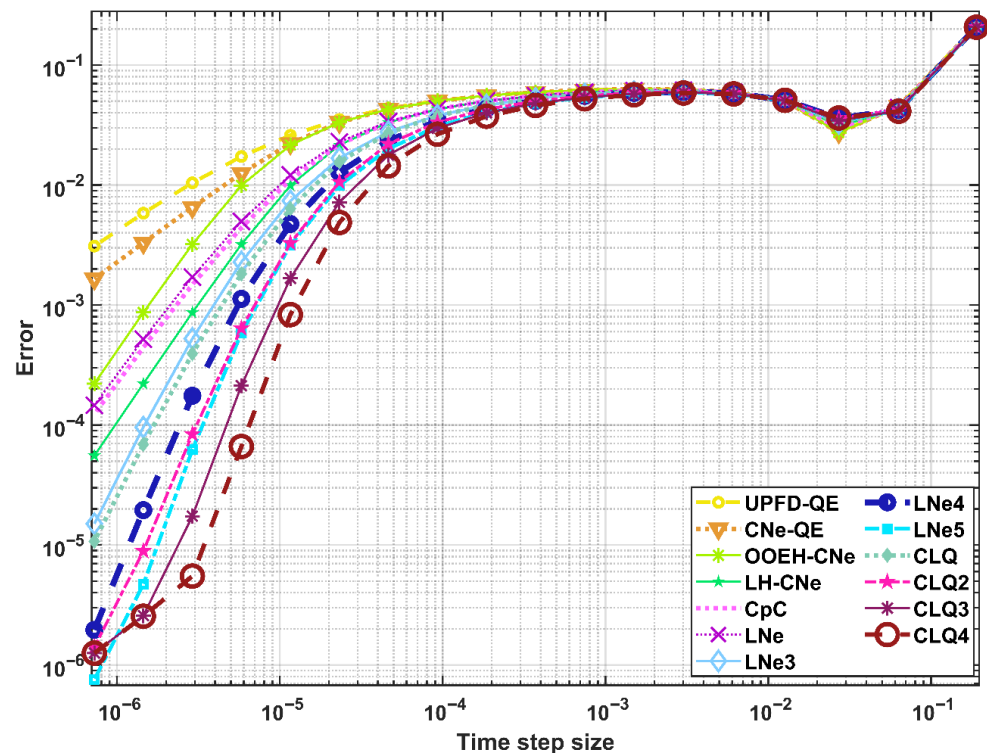


Figure 4. Maximum errors as a function of the time-step size for the t2 treatment in Case Study 2.

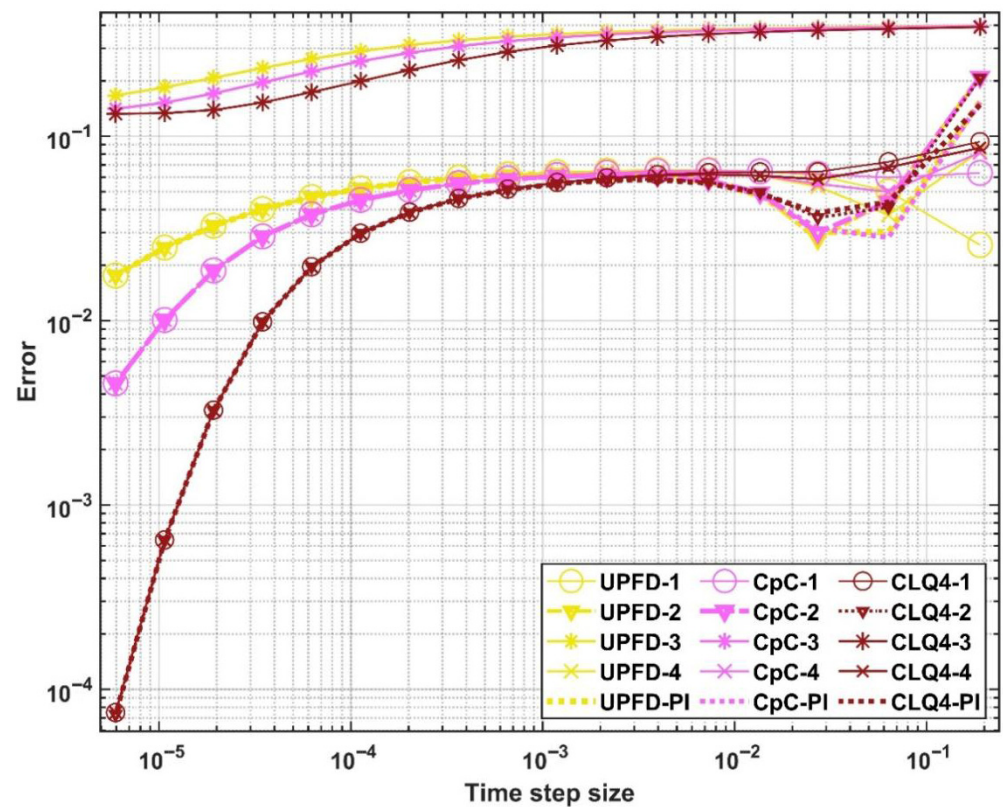


Figure 5. Maximum errors as a function of the time-step size for the five reaction-treatments in the case of three diffusion-solver schemes in Case Study 2.

Table 2. The theoretical order for the diffusion equation and numerical orders of the methods for the PI treatment in Case Study 1.

Num. Method (Theor. Order)	Time-Step Size	Error (L_∞)	NO
UPFD (1)	$h_{17} = 9.9 \times 10^{-6}$	5.0×10^{-4}	0.98
	$h_{18} = 4.9 \times 10^{-6}$	2.5×10^{-4}	
CNe (1)	$h_{17} = 9.9 \times 10^{-6}$	2.5×10^{-4}	1.0
	$h_{18} = 4.9 \times 10^{-6}$	1.3×10^{-4}	
OOEH-CNe (2)	$h_{17} = 9.9 \times 10^{-6}$	7.8×10^{-6}	2.3
	$h_{18} = 4.9 \times 10^{-6}$	1.6×10^{-6}	
LH-CNe (2)	$h_{17} = 9.9 \times 10^{-6}$	1.7×10^{-6}	2.8
	$h_{18} = 4.9 \times 10^{-6}$	2.4×10^{-7}	
CpC (2)	$h_{17} = 9.9 \times 10^{-6}$	4.7×10^{-6}	2.3
	$h_{18} = 4.9 \times 10^{-6}$	9.3×10^{-7}	
LNe (2)	$h_{17} = 9.9 \times 10^{-6}$	5.7×10^{-6}	2.2
	$h_{18} = 4.9 \times 10^{-6}$	1.2×10^{-6}	
LNe3 (2)	$h_{14} = 1.5 \times 10^{-4}$	5.4×10^{-5}	2.7
	$h_{15} = 7.9 \times 10^{-5}$	8.1×10^{-6}	
LNe4 (2)	$h_{13} = 3.1 \times 10^{-4}$	9.5×10^{-5}	3.2
	$h_{14} = 1.5 \times 10^{-4}$	1.1×10^{-5}	
LNe5 (2)	$h_{12} = 6.3 \times 10^{-4}$	3.2×10^{-4}	3.4
	$h_{13} = 3.1 \times 10^{-4}$	3.1×10^{-5}	
CLQ (3)	$h_{14} = 1.5 \times 10^{-4}$	4.1×10^{-5}	2.5
	$h_{15} = 7.9 \times 10^{-5}$	7.4×10^{-6}	
CLQ2 (4)	$h_{12} = 6.3 \times 10^{-4}$	3.6×10^{-4}	2.9
	$h_{13} = 3.1 \times 10^{-4}$	4.8×10^{-5}	
CLQ3 (4)	$h_{11} = 1.2 \times 10^{-3}$	9.7×10^{-4}	3.0
	$h_{12} = 6.3 \times 10^{-4}$	1.1×10^{-4}	
CLQ4 (4)	$h_{11} = 1.2 \times 10^{-3}$	4.5×10^{-4}	2.8
	$h_{12} = 6.3 \times 10^{-4}$	6.1×10^{-5}	

Table 3. The theoretical order for the diffusion equation and numerical orders of the methods for the t2 treatment in Case Study 2.

Num. Method (Theor. Order)	Time-Step Size	Error (L_∞)	NO
UPFD (1)	$h_{17} = 1.4 \times 10^{-6}$	5.8×10^{-3}	0.91
	$h_{18} = 7.2 \times 10^{-7}$	3.1×10^{-3}	
CNe (1)	$h_{17} = 1.4 \times 10^{-6}$	3.3×10^{-3}	0.99
	$h_{18} = 7.2 \times 10^{-7}$	1.7×10^{-3}	
OOEH-CNe (2)	$h_{17} = 1.4 \times 10^{-6}$	8.6×10^{-4}	1.9
	$h_{18} = 7.2 \times 10^{-7}$	2.2×10^{-4}	

Table 3. Cont.

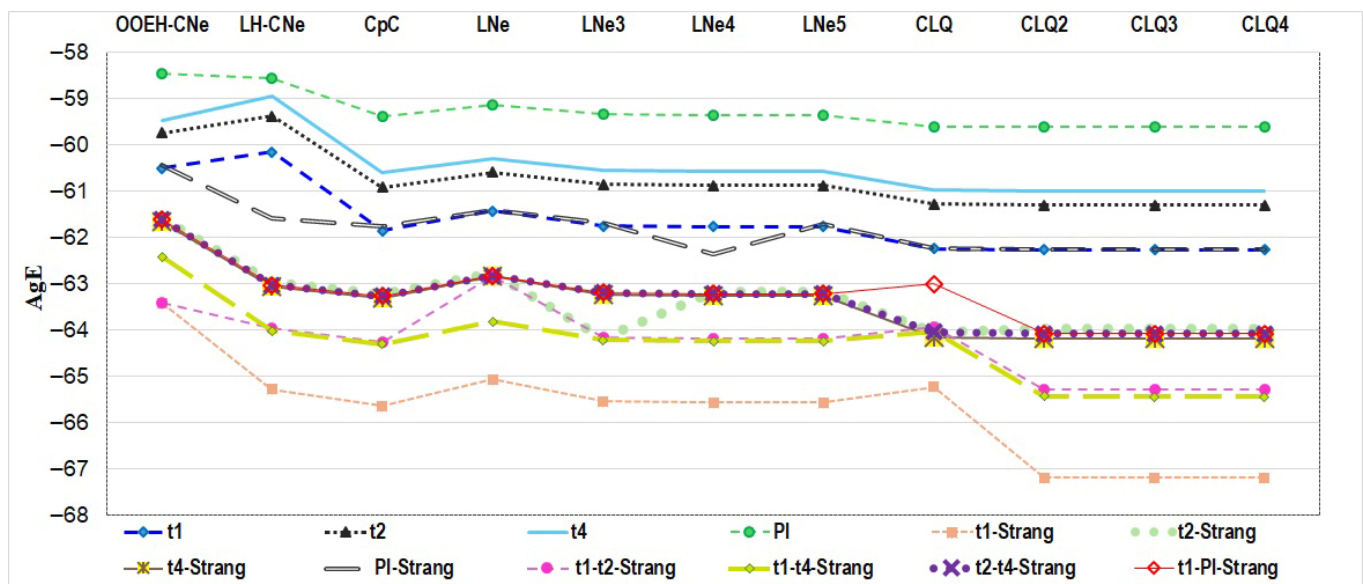
Num. Method (Theor. Order)	Time-Step Size	Error (L_∞)	NO
LH-CNe (2)	$h_{17} = 1.4 \times 10^{-6}$	2.2×10^{-4}	2
	$h_{18} = 7.2 \times 10^{-7}$	5.5×10^{-5}	
CpC (2)	$h_{17} = 1.4 \times 10^{-6}$	4.4×10^{-4}	1.8
	$h_{18} = 7.2 \times 10^{-7}$	1.2×10^{-4}	
LNe (2)	$h_{17} = 1.4 \times 10^{-6}$	5.2×10^{-4}	1.8
	$h_{18} = 7.2 \times 10^{-7}$	1.4×10^{-4}	
LNe3 (2)	$h_{17} = 1.4 \times 10^{-6}$	9.5×10^{-5}	2.6
	$h_{18} = 7.2 \times 10^{-7}$	1.5×10^{-5}	
LNe4 (2)	$h_{17} = 1.4 \times 10^{-6}$	1.9×10^{-5}	3.3
	$h_{18} = 7.2 \times 10^{-7}$	1.9×10^{-6}	
LNe5 (2)	$h_{17} = 1.4 \times 10^{-6}$	4.6×10^{-6}	3.7
	$h_{18} = 7.2 \times 10^{-7}$	7.5×10^{-7}	
CLQ (3)	$h_{17} = 1.4 \times 10^{-6}$	6.8×10^{-5}	2.6
	$h_{18} = 7.2 \times 10^{-7}$	1.1×10^{-5}	
CLQ2 (4)	$h_{16} = 2.8 \times 10^{-6}$	8.3×10^{-5}	3.2
	$h_{17} = 1.4 \times 10^{-6}$	8.9×10^{-6}	
CLQ3 (4)	$h_{15} = 5.7 \times 10^{-6}$	1.7×10^{-5}	3.6
	$h_{16} = 2.8 \times 10^{-6}$	2.5×10^{-6}	
CLQ4 (4)	$h_{15} = 5.7 \times 10^{-6}$	5.5×10^{-6}	3.6
	$h_{16} = 2.8 \times 10^{-6}$	2.5×10^{-6}	

Table 4. The AgE errors for different treatments of the diffusion and the nonlinear term in Case Study 1.

Method	Treatment of the Nonlinear Term											
	t1	t2	t4	PI	t1-Strang	t2-Strang	t4-Strang	PI-Strang	t1-t2-Strang	t1-t4-Strang	t2-t4-Strang	t1-PI-Strang
UPFD	−41.98	−41.99	−41.98	−41.99	−42.00	−42.00	−42.00	−42.00	−42.00	−41.996	−41.99	−41.99
CNe	−45.99	−46.00	−45.97	−45.98	−45.99	−46.00	−45.99	−45.99	−46.00	−45.99	−45.99	−45.99
OOEH-CNe	−60.50	−59.73	−59.47	−58.46	−63.41	−61.58	−61.64	−60.42	−63.41	−62.41	−61.61	−61.62
LH-CNe	−60.15	−59.37	−58.93	−58.55	−65.26	−62.96	−63.06	−61.58	−63.95	−64.01	−63.03	−63.02
CpC	−61.85	−60.91	−60.60	−59.38	−65.63	−63.21	−63.31	−61.75	−64.24	−64.29	−63.25	−63.25
LNe	−61.42	−60.59	−60.30	−59.13	−65.05	−62.76	−62.82	−61.39	−62.82	−63.81	−62.82	−62.82
LNe3	−61.74	−60.85	−60.55	−59.33	−65.53	−64.16	−63.22	−61.68	−64.16	−64.21	−63.18	−63.19
LNe4	−61.76	−60.87	−60.57	−59.34	−65.56	−63.16	−63.24	−62.36	−64.18	−64.23	−63.20	−63.21
LNe5	−61.77	−60.88	−60.57	−59.35	−65.57	−63.17	−63.25	−61.70	−64.19	−64.24	−63.21	−63.22
CLQ	−62.23	−61.28	−60.98	−59.60	−65.22	−64.03	−64.15	−62.24	−63.92	−64.03	−64.03	−62.99
CLQ2	−62.26	−61.29	−60.99	−59.61	−67.18	−63.95	−64.18	−62.26	−65.27	−65.42	−64.07	−64.07
CLQ3	−62.26	−61.29	−60.99	−59.61	−67.18	−63.96	−64.18	−62.26	−65.27	−65.42	−64.07	−64.07
CLQ4	−62.26	−61.29	−60.99	−59.61	−67.18	−63.96	−64.18	−62.26	−65.27	−65.42	−64.07	−64.07

Table 5. The AgE errors for different treatments of the diffusion and the nonlinear term in Case Study 2.

Methods	t1	t2	t4	PI	t1-Strang	t2-Strang	t4-Strang	PI-Strang	t1-t2-Strang	t1-t4-Strang	t2-t4-Strang	t1-PI-Strang
UPFD	−22.37	−22.58	−22.65	−22.49	−22.8	−22.73	−22.76	−22.69	−22.76	−22.78	−22.74	−22.74
CNe	−24.97	−24.67	−24.82	−24.54	−25.07	−24.93	−25	−24.86	−25.01	−25.04	−24.7	−24.99
OOEH-CNe	−31.79	−30.91	−31.34	−30.66	−32.82	−31.97	−32.38	−31.75	−32.32	−32.58	−32.17	−32.18
LH-CNe	−36.16	−35.15	−35.62	−34.85	−38.93	−37.60	−38.26	−37.29	−38.12	−38.55	−37.90	−37.91
CpC	−35.26	−34.00	−34.60	−33.68	−36.89	−35.54	−36.19	−35.22	−36.07	−36.49	−35.83	−35.84
LNe	−34.05	−32.94	−33.48	−32.66	−35.60	−34.41	−34.99	−34.12	−34.88	−35.26	−34.67	−34.68
LNe3	−36.58	−35.52	−36.03	−35.24	−39.60	−38.35	−39.00	−38.03	−38.84	−39.26	−38.64	−38.65
LNe4	−37.09	−36.03	−36.54	−35.75	−40.21	−38.97	−39.61	−38.64	−39.46	−39.87	−39.26	−39.27
LNe5	−37.30	−36.24	−36.75	−35.96	−40.44	−39.19	−39.84	−38.86	−39.68	−40.10	−39.48	−39.49
CLQ	−38.57	−37.22	−37.85	−36.90	−44.69	−41.80	−43.16	−41.18	−42.89	−43.81	−42.43	−42.44
CLQ2	−39.71	−38.36	−38.99	−38.05	−47.38	−43.97	−45.59	−43.26	−45.27	−46.35	−44.70	−44.71
CLQ3	−40.42	−39.08	−39.70	−38.77	−48.68	−45.16	−46.84	−44.43	−46.50	−47.62	−45.92	−45.93
CLQ4	−40.65	−39.31	−39.93	−39.00	−49.08	−45.52	−47.23	−44.79	−46.89	−48.02	−46.30	−46.31

**Figure 6.** The AgE errors for different treatments of the diffusion and the nonlinear term in Case Study 1.

Based on Figures 6 and 7 and Tables 4 and 5, and further results we do not present here for brevity, we intend to remove those combinations which were clearly outperformed by others. First, we can conclude that the t1 treatment with Strang-splitting is the most accurate, while the PI treatment is the least accurate. Strang-splitting yields a significant increase in accuracy, typically for the higher-order schemes, e.g., for the CLQ3 and CLQ4 schemes, which are more accurate for the diffusion equation. For the first-order UPFD and CNe solvers, the difference in accuracy between the reaction-treatments is very small. The mixing of the treatments by Strang-splitting does not yield any substantial improvement. Moreover, they require more effort to code, so we exclude them from future investigations. However, when we select the methods for further investigation, including running time tests, we have to be careful: not only the accuracy but also the speed has to be considered. The multi-stage diffusion-solvers are more accurate, but slower as well, and thus we keep both low- and high-order solvers. On the other hand, the formula of the t1-t4 treatments contains the calculation of the exponential function in each time step for each cell, e.g.,

$e^{-u_i^{\text{diff}}\beta t}$, while the PI treatment contains cheaper operations. Furthermore, Strang-splitting also consumes time. Therefore, although we select mainly the t1 and t1-Strang treatments, we keep the PI treatment combined with the simplest UPFD and CNe solver, hoping that the speed gain can dominate the loss in the accuracy, at least in some cases. To summarize, we decided to keep the following 28 combinations:

UPFD-PI	OOEH-CNe-t1-Strang	LNe3-t1	CLQ-t1-Strang
UPFD-t1	LH-CNe-t1	LNe3-t1-Strang	CLQ2-t1
UPFD-t1-Strang	LH-CNe-t1-Strang	LNe4-t1	CLQ2-t1-Strang
CNe-PI	CpC-t1	LNe4-t1-Strang	CLQ3-t1
CNe-t1	CpC-t1-Strang	LNe5-t1	CLQ3-t1-Strang
CNe-t1-Strang	LNe-t1	LNe5-t1-Strang	CLQ4-t1
OOEH-CNe-t1	LNe-t1-Strang	CLQ-t1	CLQ4-t1-Strang

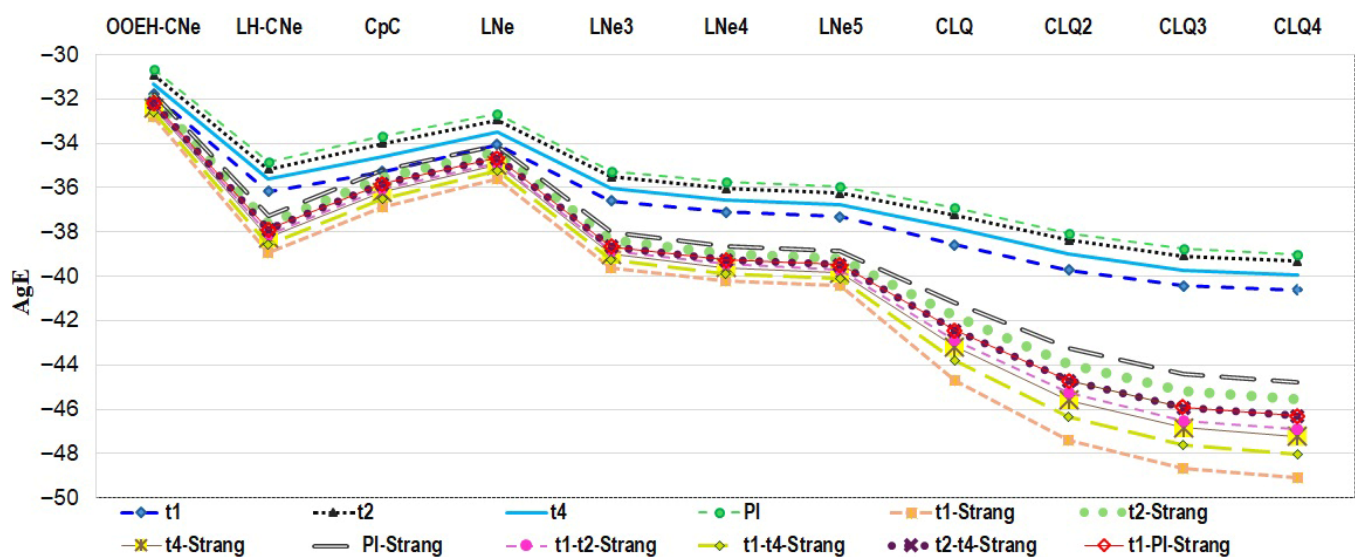


Figure 7. The AgeE errors for different treatments of the diffusion and the nonlinear term in Case Study 2.

6. Testing the Performance with Running Time Measurements in 2D

The capacity-resistivity model, introduced in Section 2, will be used for the simulations in two spatial dimensions here and in the next section. This means that the ODE system (8), supplemented with the Huxley reaction term, is solved by the selected 28 algorithm combinations. The CFL limit and the stiffness ratio can be adjusted by generating random values for the C and R quantities following a log-uniform distribution:

$$C_i = 10^{(a_C - b_C \cdot \text{rand})}, R_{x,i} = 10^{(a_{R_x} - b_{R_x} \cdot \text{rand})}, R_{y,i} = 10^{(a_{R_y} - b_{R_y} \cdot \text{rand})}. \quad (30)$$

Here, *rand* are random numbers, uniformly distributed in the unit interval by MATLAB R2021b. We assume that if our methods work in the case of random systems, they will work in other cases as well. We construct two completely different systems and via running-time measurements, try to obtain the methods with the best performance. In all 2D cases, the reference solutions are calculated by the MATLAB built-in ode15 solver using a very strict absolute and relative error tolerance. The tic-toc command of MATLAB is utilized to measure the running times, which are good approximations of the CPU times if no other tasks are given to the computer, which has a 13th Gen Intel Core i7-13700 CPU and 64GB RAM. We tried to minimize the running-time measurement errors appearing as random

fluctuations by repeating the calculations more times subsequently, and then taking the average of the measured running times.

Case Study 3: A moderately stiff system with stronger nonlinearity

The mesh size and final time used are $N_x = 51$, $N_y = 60$, $t^{\text{fin}} = 0.02$, while $\beta = 11$, and $a_C = 1$, $b_C = 2$, $a_{Rx} = 1$, $b_{Rx} = 2$, $a_{Rz} = 1$, $b_{Rz} = 2$. These parameters yield $SR = 1.9 \cdot 10^5$ and $h_{\text{MAX}}^{\text{FTCS}} = 8.5 \cdot 10^{-3}$, which means that the system is mildly stiff. The initial concentrations are random numbers: $u_i^0 = \text{rand}$. The errors as a function of the time-step size and running times are presented in Figures 8 and 9, respectively.

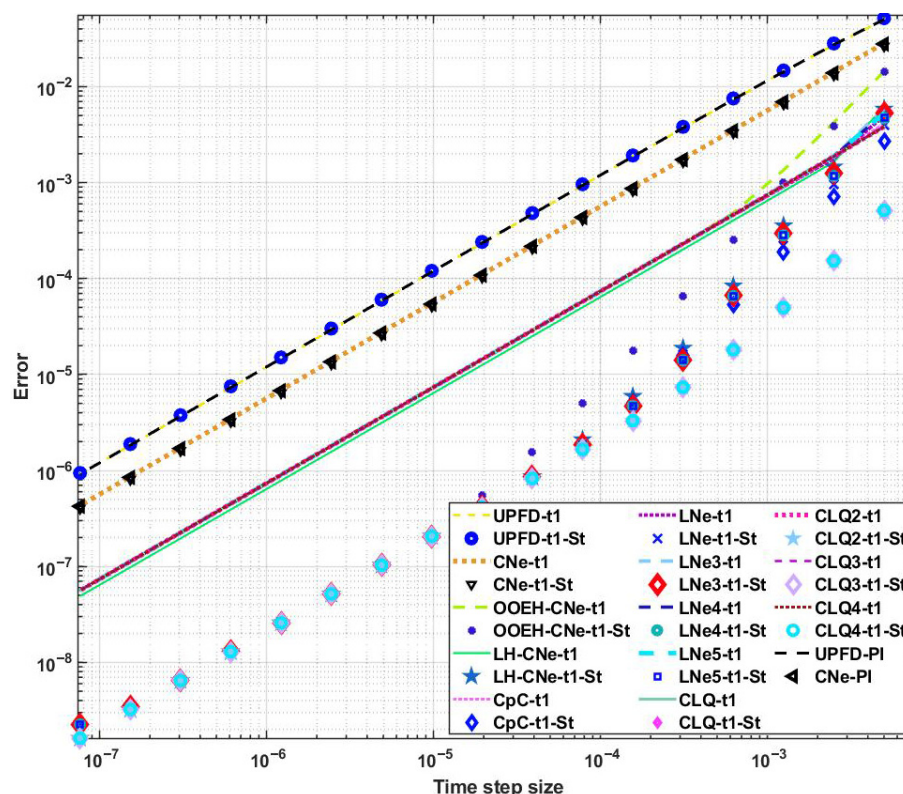


Figure 8. Maximum errors as a function of the time-step size in Case Study 3.

Case Study 4: A stiff system with weaker nonlinearity

The size of the system is $N_x = 51$, $N_y = 60$, the final time is $t^{\text{fin}} = 0.01$, the coefficient of the reaction term is $\beta = 2.9$, while the exponents are $a_C = 3$, $b_C = 6$, $a_{Rx} = 1$, $b_{Rx} = 2$, $a_{Ry} = 2$, $b_{Ry} = 4$. The problem we obtained using these parameters is characterized by the $SR = 2.68 \cdot 10^9$ and $h_{\text{MAX}}^{\text{FTCS}} = 2.1 \cdot 10^{-5}$ numbers, and hence the system is rather stiff. The initial concentration function is smooth: $u_i^0 = \frac{1}{\sqrt{1+i}} + 0.001$. The errors as a function of the time-step size and running times are presented in Figures 10 and 11, respectively.

Using the results accumulated up to this point, we further reduce the number of methods, and we keep only the top nine of them. These are the OOEh-CNe-t1-Strang, LH-CNe-t1, LH-CNe-t1-Strang, CpC-t1-Strang, and LNe-t1-Strang, as well as four higher-order combinations: the CLQ-CLQ4 series with the t1 treatment and Strang-splitting.

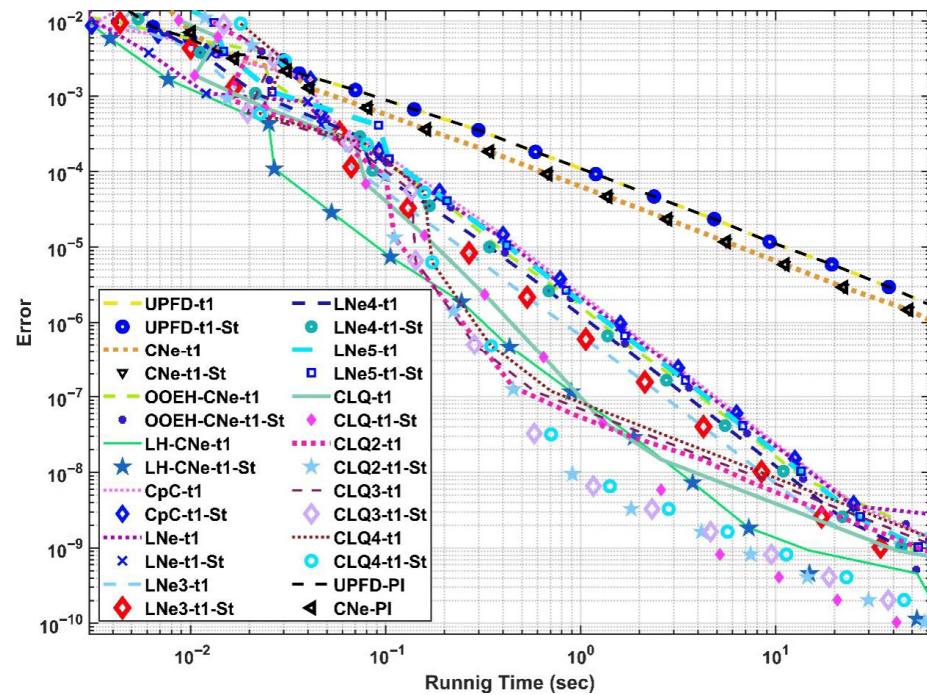


Figure 9. Maximum errors as a function of the running time in Case Study 3.

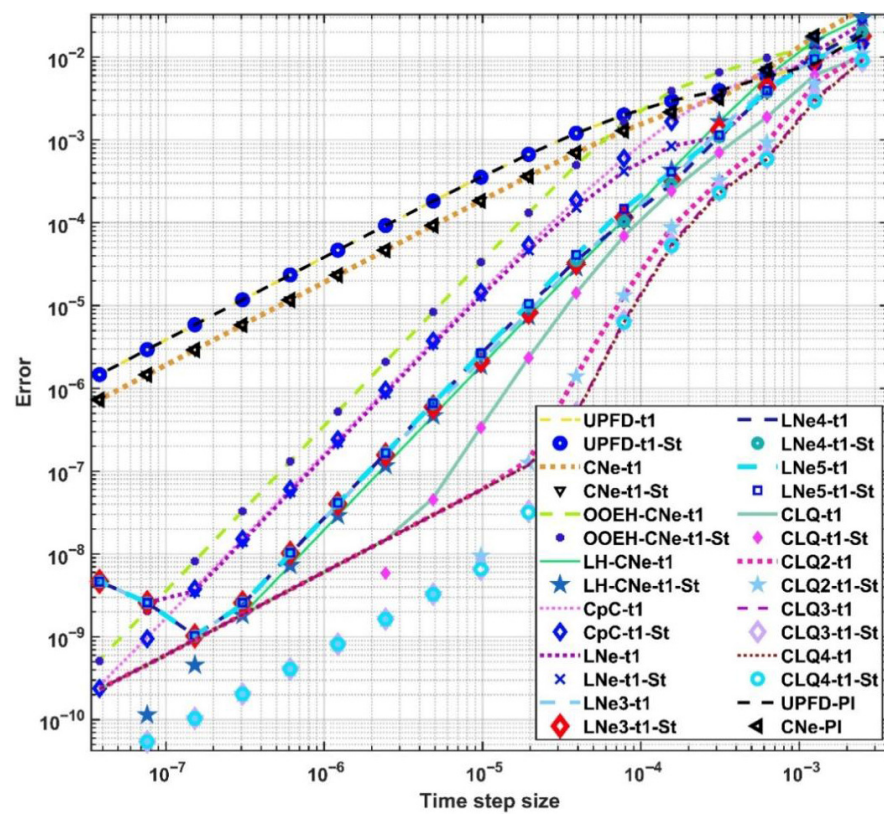


Figure 10. Maximum errors as a function of the time-step size in Case Study 4.

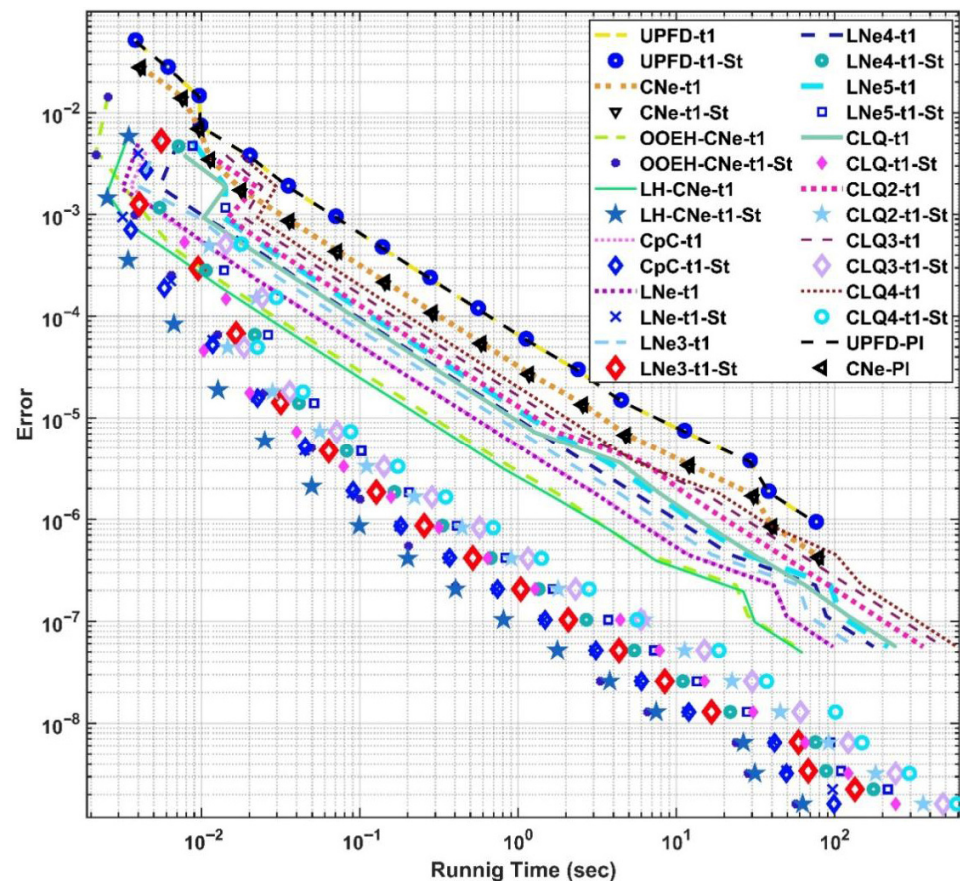


Figure 11. Maximum errors as a function of the running time in Case Study 4.

7. Parameter-Sweep for Further Testing of the Performance of the Top 9 Methods

Case Study 5: Errors as a function of the β parameter, non-stiff case

The following series is used for the values of the nonlinear coefficient: $\beta \in \{0, 0.25, 1, 2, 5, 10, 15, 20, 30, 50\}$. To construct a system with the lowest possible stiffness ratio, all a and b exponents are set to zero, i.e., $C = 1$, $R_x = R_y = 1$ for all cells. The system size is set to $N_x = 21$ and $N_y = 12$, and the final time is $t^{fin} = 2.1$. The initial concentration function is given by $u_i^0 = \frac{2}{i+2} + 0.2 \text{ rand}$. Figure 12 shows the AgE errors as a function of the square root of the β coefficient. One can observe that the errors increase when the level of nonlinearity is increasing until $\beta \approx 5$. If β surpasses this region, the solution function is extremely close to 1; therefore, the errors become smaller and smaller again. In this region, the diffusion-solver has no significance, while the Strang-splitting yields a slight but noticeable advantage.

Case Study 6: Errors as a function of the β parameter, stiff case

The following series is used for the values of the nonlinear coefficient: $\beta \in \{0, 0.1, 0.25, 1, 4, 9, 16, 25, 36, 64, 81, 100\}$. The a and b exponents are larger: $a_C = 3$, $b_C = 6$, $a_{Rx} = 1$, $b_{Rx} = 2$, $a_{Ry} = 2$, $b_{Ry} = 4$. The system size is set to $N_x = 15$ and $N_y = 20$, and the final time is $t^{fin} = 0.2$. The initial concentration function is given by $u_i^0 = \frac{1}{i+2} + 0.4 \text{ rand}$. Figure 13 shows the AgE errors as a function of $\sqrt{\beta}$. One can see that the accuracy of the methods deteriorates when the β parameter is increasing, but the higher-order diffusion-solvers lose their advantage at a much slower rate than for a non-stiff system.

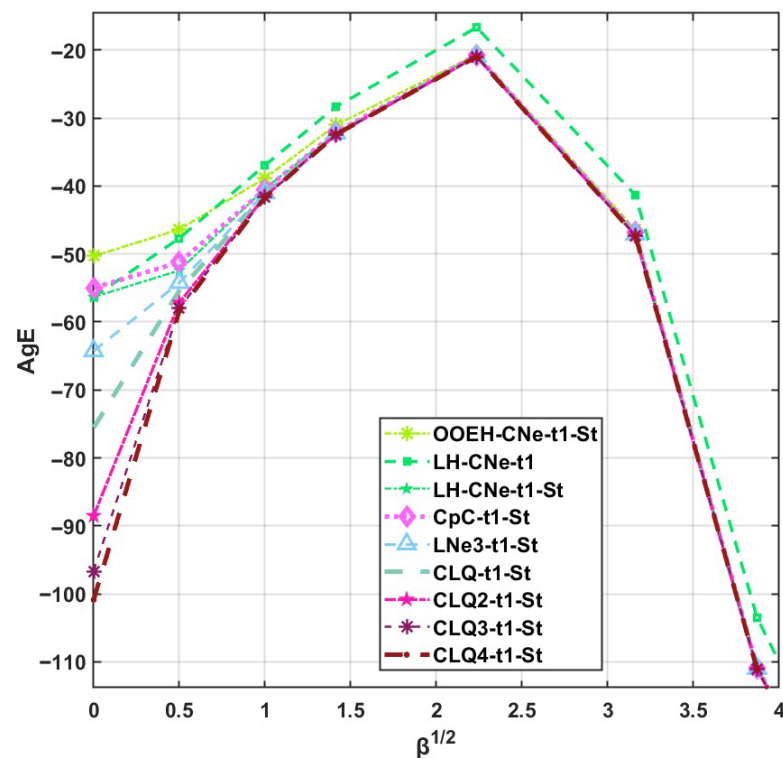


Figure 12. The AgE errors as a function of the square root of β in Case Study 5.

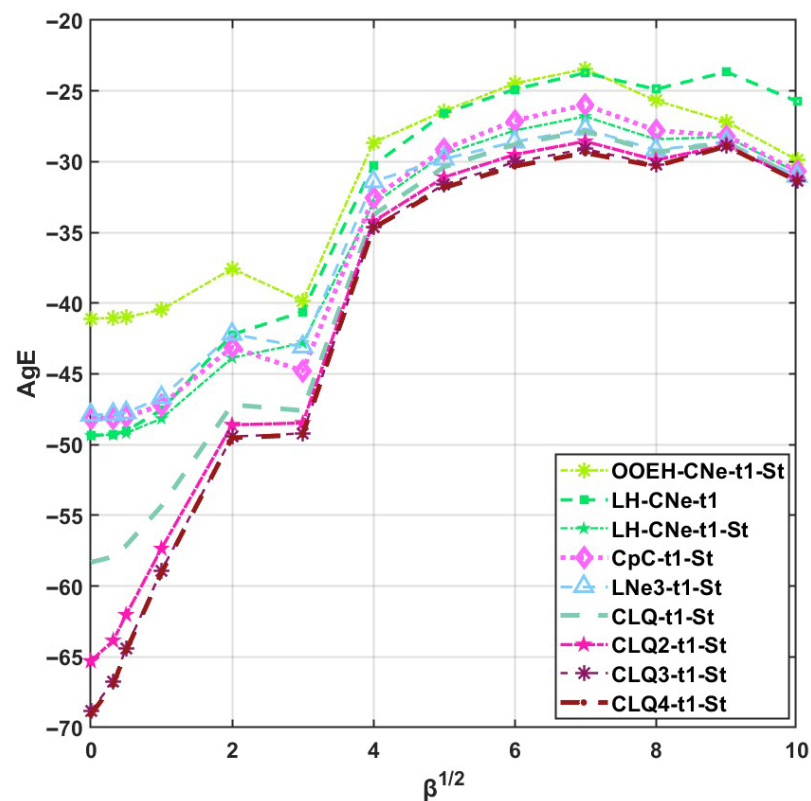


Figure 13. The AgE errors as a function of the square root of β in Case Study 6.

Case Study 7: Errors as a function of the CFL limit

A series of test problems with a decreasing CFL limit and increasing stiffness ratio has been set up by gradually increasing the a and b exponents in (30), as Table 6 shows. All other parameters are kept fixed, such as $\beta = 4.3$, $N_x = 25$ and $N_y = 22$, and $t^{fin} = 0.2$. The initial

values were generated only once for all the cases using the formula $u_i^0 = \frac{3}{i+5} + 0.5 \text{ rand.}$ $P = 15$ different time-step sizes have been used in the definitions (28) and (29) of the aggregated error.

Table 6. Data of the 10 random systems produced with a decreasing CFL limit (increasing stiffness ratio).

Number	a_C	b_C	a_{Rx}	b_{Rx}	a_{Ry}	b_{Ry}	h_{MAX}^{RK1}	SR
1	0	0	0	0	0	0	0.25	5.04×10^2
2	−1	1	0	0	0	0	0.04	7.3×10^3
3	−1	1	−1	1	0	0	0.015	1.8×10^4
4	−1	1	−1	1	−1	1	6.5×10^{-3}	5.0×10^4
5	−2	2	−1	1	−1	1	7.9×10^{-4}	1.9×10^6
6	−2	2	−2	2	−1	1	2.5×10^{-4}	5.3×10^6
7	−2	2	−2	2	−2	2	1.0×10^{-4}	1.8×10^7
8	−3	3	−2	2	−2	2	2.5×10^{-5}	5.9×10^8
9	−3	3	−3	3	−2	2	4.7×10^{-6}	3.5×10^9
10	−3	3	−3	3	−3	−3	1.4×10^{-6}	9.1×10^9

The AgE errors for different CFL limits are presented in Figure 14 and Table 7. As was expected, the accuracy of the methods deteriorates with a decreasing CFL limit and increasing stiffness ratio. The high-order CLQ family has a large relative advantage for a small CFL limit. The error of the OOEH scheme increases faster than those of the other schemes with a decreasing CFL limit and exceeds the error of the LH-CNe without Strang-splitting.

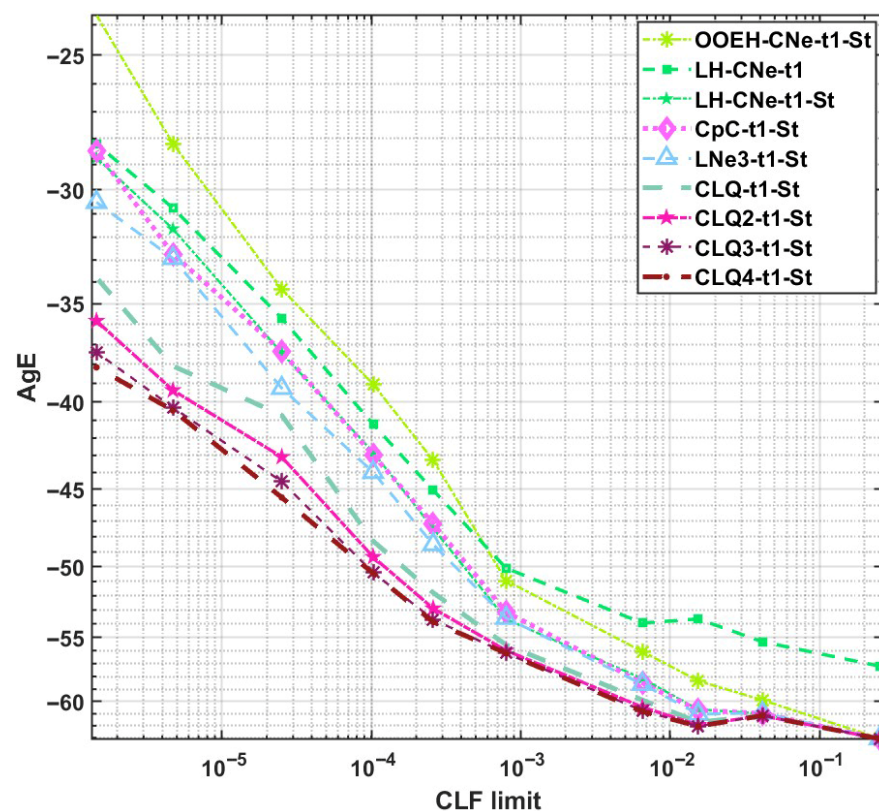


Figure 14. The AgE errors as the function of the CFL limit in Case Study 7.

Table 7. The AgE errors for a decreasing CFL limit (increasing stiffness) in Case Study 7. The serial numbers in the first row are explained in Table 6.

Method	1	2	3	4	5	6	7	8	9	10
OOEH-CNe-t1-St	−63.06	−59.88	−58.34	−56.07	−50.93	−43.27	−39.04	−34.32	−28.19	−23.69
LH-CNe-t1	−57.18	−55.33	−53.65	−53.94	−50.08	−45.07	−41.22	−35.71	−30.75	−28.20
LH-CNe-t1-St	−63.15	−60.98	−60.63	−58.15	−53.67	−47.46	−42.81	−37.42	−31.63	−28.74
CpC-t1-St	−63.10	−60.97	−60.71	−58.50	−53.15	−47.15	−42.97	−37.34	−32.73	−28.47
LNe3-t1-St	−63.08	−60.93	−61.03	−58.59	−53.58	−48.50	−43.98	−39.25	−32.92	−30.51
CLQ-t1-St	−63.09	−61.16	−61.69	−59.87	−55.57	−51.77	−48.28	−40.72	−38.08	−33.79
CLQ2-t1-St	−63.09	−61.15	−61.93	−60.45	−55.99	−52.89	−49.33	−43.08	−39.36	−35.83
CLQ3-t1-St	−63.10	−61.15	−62.04	−60.74	−56.19	−53.68	−50.39	−44.55	−40.29	−37.40
CLQ4-t1-St	−63.11	−61.16	−62.04	−60.81	−56.18	−53.87	−50.43	−45.51	−40.44	−38.16

8. Discussion and Summary

The numerical solution of a special form of the nonlinear Huxley equation is examined, for which the true solution remains in the unit interval. We constructed numerous algorithm combinations with operator-splitting and showed that they preserve this property for arbitrary spatial mesh, the time-step size, and values of the nonlinear coefficient β . The numerical experiments confirmed these results and helped us to gradually tighten the large set of the examined methods to find the best choices for different circumstances. We can propose the LH-CNe diffusion-solver combined with t1 treatments and Strang-splitting if any of the following conditions are fulfilled:

1. Speed is more important than the accuracy of the result.
2. The CFL limit is moderate or the stiffness is not high.
3. The nonlinear parameter β is large: larger than 1 for an equidistant mesh or larger than 5 for a stiff case.

However, if high accuracy is required in a stiff case, the CLQ family combined with t1 treatments and Strang-splitting is the best choice. Furthermore, if the nonlinearity is very low, Strang-splitting is redundant, but in all other cases, it is worth coding. If one has a physical system with a complicated geometry, the odd–even hopscotch structure may be hard to construct. In those cases, the CpC-t1-Strang solver can be recommended if high speed is required, since the diffusion-solver CpC contains only two stages, and it usually yields an acceptable accuracy.

Author Contributions: Conceptualization, methodology, project administration, supervision, and resources, E.K.; software, H.K. and E.K.; validation, visualization, and investigation, H.K.; formal analysis, E.K.; writing, E.K. and H.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the authors.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Bradshaw-Hajek, B. Reaction-Diffusion Equations for Population Genetics. Ph.D. Dissertation, University of Wollongong, Wollongong, Australia, 2004.

2. Mohan, M.T.; Khan, A. On the generalized Burgers-Huxley equation: Existence, uniqueness, regularity, global attractors and numerical studies. *Discret. Contin. Dyn. Syst.—B* **2021**, *26*, 3943–3988. [\[CrossRef\]](#)
3. Agbavon, K.M.; Appadu, A.R. Construction and analysis of some nonstandard finite difference methods for the FitzHugh–Nagumo equation. *Numer. Methods Partial Differ. Equ.* **2020**, *36*, 1145–1169. [\[CrossRef\]](#)
4. Wang, X. Nerve propagation and wall in liquid crystals. *Phys. Lett. A* **1985**, *112*, 402–406. [\[CrossRef\]](#)
5. Koroğlu, C.; Aydin, A. Exact and nonstandard finite difference schemes for the Burgers equation B(2,2). *Turk. J. Math.* **2021**, *45*, 647–660. [\[CrossRef\]](#)
6. Jejenywa, O.A.; Gidey, H.H.; Appadu, A.R. Numerical Modeling of Pollutant Transport: Results and Optimal Parameters. *Symmetry* **2022**, *14*, 2616. [\[CrossRef\]](#)
7. Kumar, S.; Ramcharan, B.; Yadav, V.S.; Singh, A.; Bhumkar, Y.G.; Rajpoot, M.K. Exploring Acoustic and Sound Wave Propagation Simulations: A Novel Time Advancement Method for Homogeneous and Heterogeneous Media. *Adv. Appl. Math. Mech.* **2025**, *17*, 315–349. [\[CrossRef\]](#)
8. Ketcheson, D.I. Highly Efficient Strong Stability-Preserving Runge–Kutta Methods with Low-Storage Implementations. *SIAM J. Sci. Comput.* **2008**, *30*, 2113–2136. [\[CrossRef\]](#)
9. Qin, X.; Jiang, Z.; Yan, C. Strong Stability Preserving Two-Derivative Two-Step Runge-Kutta Methods. *Mathematics* **2024**, *12*, 2465. [\[CrossRef\]](#)
10. Anjuman; Leung, A.Y.T.; Das, S. Two-Dimensional Time-Fractional Nonlinear Drift Reaction–Diffusion Equation Arising in Electrical Field. *Fractal Fract.* **2024**, *8*, 456. [\[CrossRef\]](#)
11. Bansal, S.; Natesan, S. A novel higher-order efficient computational method for pricing European and Asian options. *Numer. Algorithms* **2024**, *99*, 1127–1159. [\[CrossRef\]](#)
12. Chou, C.S.; Zhang, Y.T.; Zhao, R.; Nie, Q. Numerical methods for stiff reaction-diffusion systems. *Discret. Contin. Dyn. Syst.—Ser. B* **2007**, *7*, 515–525. [\[CrossRef\]](#)
13. El-Sayed El-Danaf, T.; Ali Zaki, M.; Moenaaem, W. New numerical technique for solving the fractional Huxley equation. *Int. J. Numer. Methods Heat Fluid Flow* **2014**, *24*, 1736–1754. [\[CrossRef\]](#)
14. Patel, Y.F.; Dhodiya, J.M. Efficient algorithm to study the class of Burger’s Fisher equation. *Int. J. Appl. Nonlinear Sci.* **2022**, *3*, 242–266. [\[CrossRef\]](#)
15. Bai, Y.; Chaolu, T.; Bilige, S. Solving Huxley equation using an improved PINN method. *Nonlinear Dyn.* **2021**, *105*, 3439–3450. [\[CrossRef\]](#)
16. Manaa, S.; Sabawi, M. Numerical Solution and Stability Analysis of Huxley Equation. *AL-Rafidain J. Comput. Sci. Math.* **2005**, *2*, 85–97. [\[CrossRef\]](#)
17. Chen-Charpentier, B.M.; Kojouharov, H. V An unconditionally positivity preserving scheme for advection-diffusion reaction equations. *Math. Comput. Model.* **2013**, *57*, 2177–2185. [\[CrossRef\]](#)
18. Appadu, A.R. Performance of UPFD scheme under some different regimes of advection, diffusion and reaction. *Int. J. Numer. Methods Heat Fluid Flow* **2017**, *27*, 1412–1429. [\[CrossRef\]](#)
19. Drljača, B.; Savović, S. Unconditionally positive finite difference and standard explicit finite difference schemes for power flow equation. *Univ. Thought—Publ. Nat. Sci.* **2019**, *9*, 75–78. [\[CrossRef\]](#)
20. Macías-Díaz, J.E. On an exact numerical simulation of solitary-wave solutions of the Burgers–Huxley equation through Cardano’s method. *BIT Numer. Math.* **2014**, *54*, 763–776. [\[CrossRef\]](#)
21. Chertock, A.; Kurganov, A. A second-order positivity preserving central-upwind scheme for chemotaxis and haptotaxis models. *Numer. Math.* **2008**, *111*, 169–205. [\[CrossRef\]](#)
22. Agbavon, K.M.; Appadu, A.R.; Khumalo, M. On the numerical solution of Fisher’s equation with coefficient of diffusion term much smaller than coefficient of reaction term. *Adv. Differ. Equ.* **2019**, *2019*, 146. [\[CrossRef\]](#)
23. Appadu, A.R.; Inan, B.; Tijani, Y.O. Comparative study of some numerical methods for the Burgers-Huxley equation. *Symmetry* **2019**, *11*, 1333. [\[CrossRef\]](#)
24. Songolo, M.E. A Positivity-Preserving Nonstandard Finite Difference Scheme for Parabolic System with Cross-Diffusion Equations and Nonlocal Initial Conditions. *Am. Sci. Res. J. Eng. Technol. Sci.* **2016**, *18*, 252–258.
25. Chapwanya, M.; Lubuma, J.M.S.; Mickens, R.E. Positivity-preserving nonstandard finite difference schemes for cross-diffusion equations in biosciences. *Comput. Math. Appl.* **2014**, *68*, 1071–1082. [\[CrossRef\]](#)
26. Khayrullaev, H.; Omle, I.; Kovács, E. Systematic Investigation of the Explicit, Dynamically Consistent Methods for Fisher’s Equation. *Computation* **2024**, *12*, 49. [\[CrossRef\]](#)
27. Khayrullaev, H.; Omle, I.; Kovács, E. Exploring the Performance of Some Efficient Explicit Numerical Methods with Good Stability Properties for Huxley’s Equation. *Mathematics* **2025**, *13*, 207. [\[CrossRef\]](#)
28. Munka, M.; Pápay, J. *4D Numerical Modeling of Petroleum Reservoir Recovery*; Akadémiai Kiadó: Budapest, Hungary, 2001; ISBN 963 05 7843 3.

29. Kovács, E.; Nagy, Á.; Saleh, M. A set of new stable, explicit, second order schemes for the non-stationary heat conduction equation. *Mathematics* **2021**, *9*, 2284. [[CrossRef](#)]
30. Kovács, E. A class of new stable, explicit methods to solve the non-stationary heat equation. *Numer. Methods Partial. Differ. Equ.* **2020**, *37*, 2469–2489. [[CrossRef](#)]
31. Kovács, E.; Majár, J.; Saleh, M. Unconditionally Positive, Explicit, Fourth Order Method for the Diffusion- and Nagumo-Type Diffusion–Reaction Equations. *J. Sci. Comput.* **2024**, *98*, 39. [[CrossRef](#)]
32. Nagy, Á.; Majár, J.; Kovács, E. Consistency and Convergence Properties of 20 Recent and Old Numerical Schemes for the Diffusion Equation. *Algorithms* **2022**, *15*, 425. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.