Application of Runge–Kutta Methods to the Solution of the Heat Equation

JIŘÍ ZELINKA
Department of Mathematics and Statistics,
Faculty of Science, Masaryk University,
Kotlářská 2, 611 37 Brno
CZECH REPUBLIC

Abstract: This article studies the numerical solution of the one-dimensional heat equation by Runge–Kutta time discretization combined with a standard finite-difference discretization in space. Explicit and implicit Runge–Kutta methods are considered within the method-of-lines framework. For explicit methods, formulas for the stage vectors and the resulting stability restrictions are derived, showing how the stability constraint limits the benefit of higher temporal order. For implicit methods, the Runge–Kutta stages lead to a sparse block system whose blocks are tridiagonal matrices. A block Gauss–Seidel iteration is proposed for the efficient solution of this system, and sufficient convergence conditions are discussed. The convergence analysis is developed in more detail for two-stage and three-stage methods. Finally, a numerical example based on the fourth-order two-stage Gauss–Legendre method is compared with the Crank–Nicolson scheme and illustrates the advantage of higher-order implicit Runge–Kutta discretization when the temporal error is dominant.

Key-Words: heat equation, Runge–Kutta method, stage of method, order of method, stability, Crank–Nicolson scheme, Gauss–Seidel method

Received: April 2, 2026. Revised: May 17, 2026. Accepted: June 19, 2026. Published: November 10, 2026.

1 Introduction

For solving the one-dimensional heat equation, finite difference methods are commonly used, in particular the Crank–Nicolson method, which provides error estimates of order $O(\tau^2 + h^2)$ for the time step τ and the spatial step h. However, the use of the method of lines makes it possible to improve the order of the error with respect to the time step, which can be useful especially in situations where the time step is fixed and has a decisive influence on the overall error of the approximate solution. It can therefore be advantageous to use Runge–Kutta methods, which are very well known and also thoroughly studied in terms of their properties.

Although it is possible to find tutorial videos explaining how to use these methods to solve the heat equation, the author was able to find only a few scientific papers on this topic (see e.g. [1], [2]), and only one that also deals with implicit Runge–Kutta methods (see [3]). That paper contains a truly deep theoretical analysis of the general approach; however, constructing a practical and efficient algorithm for solving a specific equation based on it would be quite difficult.

Therefore, the author decided to study the topic independently, to derive computational schemes for both explicit and implicit Runge–Kutta methods, and to propose an efficient algorithm for solving the system of linear equations that arises in implicit methods. Author also presents a sufficient condition for the general convergence of this algorithm, which, however, involves a very strong assumption. In a specific situation it is possible to use the procedure presented in the article, with the help of which we can obtain better conditions for the convergence of the algorithm. The procedure is analyzed in detail for two-stage and three-stage methods.

Chapters 2 and 3 are just a summary of known facts and were included in the article mainly because of the notation used. The main results are contained in Chapter 4, especially Sections 4.2 and 4.3, devoted to implicit methods and algorithm for solving the obtained block system of equations. Here, the above-mentioned algorithm for the efficient solution of the heat conduction equation using the implicit Runge–Kutta method is derived.

2 Runge–Kutta methods very briefly indeed

As already mentioned in the introduction, Runge–Kutta methods (hereafter RK methods) are very well known and theoretically well developed, and they belong to the most widely used methods for the numerical solution of ordinary differential equations. Therefore, we will discuss them here only very briefly, mainly to establish the notation that will be used later. Readers interested in a deeper understanding of these methods may consult, for example, the classical textbook (see [4]); standard references are also include in [5], [6], while the order theory of Runge–Kutta processes goes back to works such as [7].

We describe the principle of the methods using the simple initial value problem

$$y' = f(t, y), \quad y(t_0) = y_0. \tag{1}$$

Continuity of f in t and Lipschitz continuity in y guarantee the existence of a local (and also global) solution. RK methods can also be applied to systems of differential equations $\boldsymbol{y}' = \boldsymbol{f}(t, \boldsymbol{y}), \quad \boldsymbol{y}(t_0) = \boldsymbol{y}_0$ for vector-valued functions $\boldsymbol{y}$ and $\boldsymbol{f}$.

The approximate solution of the initial problem is naturally sought at discrete points $t_0, t_1, \ldots$.

RK methods are single-step methods; however, when transitioning from t_i to t_{i+1}, values of the function f at additional auxiliary points are used. Thus, the value of the approximate solution at t_{i+1} is obtained as a linear combination of values of the vector field in the interval between t_i and t_{i+1}.

RK methods are divided into two main groups: explicit and implicit. For both groups, however, the same formula is used to compute the approximate solution at t_{i+1}:

$$y_{i+1} = y_i + b_1 k_1 + b_2 k_2 + \cdots + b_s k_s \tag{2}$$

The coefficients b_j are determined by the specific RK method, and k_j are the above-mentioned values of the vector field multiplied by the step $\tau = t_{i+1} - t_i$.

In the case of explicit methods, these values are computed sequentially using the equations

$$\begin{aligned} k_1 &= \tau f(t_i, y_i) \\ k_2 &= \tau f(t_i + c_2\tau, y_i + a_{21}k_1) \\ k_3 &= \tau f(t_i + c_3\tau, y_i + a_{31}k_1 + a_{32}k_2) \\ k_4 &= \tau f(t_i + c_4\tau, y_i + a_{41}k_1 + a_{42}k_2 + a_{43}k_3) \\ &\vdots \end{aligned} \tag{3}$$

The coefficients c_j and a_{jl} are again determined by the specific method. All coefficients are arranged in the Butcher tableau:

$$\begin{array}{c|ccccc} c_2 & a_{21} & & & & \\ c_3 & a_{31} & a_{32} & & & \\ \vdots & \vdots & & & & \\ c_s & a_{s1} & a_{s2} & \ldots & a_{s,s-1} & \\ \hline & b_1 & b_2 & \ldots & b_{s-1} & b_s \end{array} \tag{4}$$

Remark: Usually, the step τ is not included in the coefficients k_j, and the RK formulas are then written as

$$\begin{aligned} y_{i+1} &= y_i + \tau(b_1 k_1 + b_2 k_2 + \cdots + b_s k_s) \\ k_1 &= f(t_i, y_i) \\ k_2 &= f(t_i + c_2\tau, y_i + \tau a_{21} k_1) \\ &\vdots \end{aligned} \tag{5}$$

In this paper, however, we include the step τ as part of the coefficients k_j, which leads to somewhat simpler expressions in the following sections.

For implicit methods, the values k_j are given by a system of nonlinear equations:

$$\begin{aligned} k_1 &= \tau f(t_i + c_1\tau, y_i + \sum_{j=1}^{s} a_{1j}k_j) \\ k_2 &= \tau f(t_i + c_2\tau, y_i + \sum_{j=1}^{s} a_{2j}k_j) \\ &\vdots \end{aligned} \tag{6}$$

Solving this system may appear complicated in general; however, the equations directly define an iterative method for their computation. We determine initial values for k_j, for example using an explicit RK method, and then by substituting into the right-hand sides of the equations we obtain new iterates, which are subsequently used in the iterative process. It can be shown that if the step τ is sufficiently small, the iterative process converges.

The Butcher tableau for an implicit method has the form

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \ldots & a_{1,s} \\ c_2 & a_{21} & a_{22} & \ldots & a_{2,s} \\ \vdots & \vdots & \vdots & & \vdots \\ c_s & a_{s1} & a_{s2} & \ldots & a_{s,s} \\ \hline & b_1 & b_2 & \ldots & b_s \end{array} \tag{7}$$

or, more compactly,

$$\begin{array}{c|c} \boldsymbol{c} & A \\ \hline & \boldsymbol{b}^T \end{array} \tag{8}$$

The Butcher tableau for explicit methods can of course also be written in this form, with the coefficient c_1 and the entries of the matrix A on and above the diagonal equal to zero.

The number s, which denotes the number of summands used in the sums, is called the number of stages. Another important criterion of RK methods is their order. Roughly speaking, a method is said to be of order p if the error of the approximation is $O(\tau^p)$.

The equations that the coefficients must satisfy for the method to achieve a given order are called order conditions. Here we present the order conditions for $p \leq 4$:

$$p = 1\text{: } \sum_{j=1}^{s} b_j = 1 \text{ (consistency condition)} \tag{9}$$

$$p = 2\text{: } \sum_{j=1}^{s} b_j = 1, \sum_{j=1}^{s} b_j c_j = \tfrac{1}{2} \tag{10}$$

$$p = 3\text{: previous conditions and additionally } \sum_{j=1}^{s} b_j c_j^2 = \tfrac{1}{3}, \sum_{j=1}^{s}\sum_{l=1}^{s} b_j a_{jl} c_l = \tfrac{1}{6} \tag{11}$$

$p = 4$: previous conditions and additionally

$$\sum_{i=1}^{s} b_i c_i^3 = \frac{1}{4} \tag{12}$$

$$\sum_{i,j=1}^{s} b_i c_i a_{ij} c_j = \frac{1}{8} \tag{13}$$

$$\sum_{i,j=1}^{s} b_i a_{ij} c_j^2 = \frac{1}{12} \tag{14}$$

$$\sum_{i,j,k=1}^{s} b_i a_{ij} a_{jk} c_k = \frac{1}{24} \tag{15}$$

Furthermore, all commonly used methods satisfy the relation $c_j = \sum_{l=1} a_{jl}$. The Table 1 shows the number of conditions for a given order.

Table 1: Number of conditions

order	1	2	3	4	5	6	7	8
number of cond.	1	2	4	8	17	37	85	200

Source: [4].

A very important property of all methods for the numerical solution of differential equations is their stability. A key role is played by the stability region, which is the region in the complex plane where the method exhibits correct behaviour on a certain test problem. If this region contains the entire half-plane of complex numbers with negative real part, the method is called absolutely stable, or A-stable.

A stronger property is so-called L-stability; L-stable methods are, for example, suitable for stiff problems. Unfortunately, all explicit methods have a limited stability region, while implicit methods may be A-stable and even L-stable.

As an example, we present two fourth-order methods. The first is the well-known classical Runge–Kutta method, which is an explicit fourth-stage method given by the following Butcher tableau:

$$\begin{array}{c|cccc} 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 1/3 & 1/3 & 1/6 \end{array} \tag{16}$$

The second example is the Gauss–Legendre method, which is a two-stage implicit method that is A-stable. It is the simplest implicit fourth-order method and will also be used in the following part of this article:

$$\begin{array}{c|cc} \frac{1}{2}-\frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4}-\frac{\sqrt{3}}{6} \\ \frac{1}{2}+\frac{\sqrt{3}}{6} & \frac{1}{4}+\frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \tag{17}$$

3 Heat equation – again very briefly

The physical interpretation of the one-dimensional heat equation consists in modelling thermal processes in a thin rod or wire, but it also has applications, for example, in financial mathematics.

Let u and q be functions of two variables, $u = u(x,t)$, $q = q(x,t)$, where $x \in [0,1]$, $t \geq 0$. The heat equation then has the form

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} + q. \tag{18}$$

The solution of this equation is a function $u(x,t)$ describing the temperature of the rod at a given point and time. The function q represents external heat sources, and the coefficient $\alpha > 0$ is typically a constant describing the material properties of the rod.

To solve the problem, we need the initial condition $u(x,0) = u^0(x)$, $x \in [0,1]$, and boundary conditions; in our case we use Dirichlet conditions of the form $u(i,t) = g_i(t)$, $t \geq 0$, $i \in \{0,1\}$. For $q \equiv 0$ we obtain the homogeneous equation.

We seek an approximate solution on $[0,1] \times [0,T]$, $T > 0$, at grid points defined as follows: let $h = 1/N$ be the spatial step and $\tau = T/M$ the time step. We set $x_0 = 0$, $x_N = 1$, $x_i = i\,h$, $t_0 = 0$, $t_M = T$, $t_k = k\,\tau$.

We denote the approximate solution by $u_i^k \approx u(x_i, t_k)$. The source terms are known exactly: $q_i^k = q(x_i, t_k)$, $g_0^k = g_0(t_k)$, $g_1^k = g_1(t_k)$.

The solution is computed layer by layer in time using standard finite difference approximations of derivatives. The most common methods are the explicit method, the implicit method, and the Crank–Nicolson method (see [8], [9]).

3.1 Explicit method

We approximate the equation at the point (x_i, t_k):

$$\frac{1}{\tau}(u_i^{k+1} - u_i^k) = \frac{\alpha}{h^2}\left(u_{i-1}^k - 2u_i^k + u_{i+1}^k\right) + q_i^k. \tag{19}$$

From this we obtain

$$u_i^{k+1} = \frac{\alpha\tau}{h^2}\left(u_{i-1}^k - 2u_i^k + u_{i+1}^k\right) + u_i^k + \tau q_i^k. \tag{20}$$

Letting $r = \frac{\alpha\tau}{h^2}$, the explicit scheme can be written as

$$u_i^{k+1} = r\,u_{i-1}^k + (1-2r)u_i^k + r\,u_{i+1}^k + \tau q_i^k. \tag{21}$$

This gives an explicit update formula for the values $u_1^{k+1}, \ldots, u_{N-1}^{k+1}$. The truncation error is $O(\tau + h^2)$, and it can be shown that for $r = \frac{1}{6}$ the error becomes $O(\tau^2 + h^4)$.

3.2 Implicit method

We approximate the equation at the point (x_i, t_{k+1}):

$$\frac{1}{\tau}(u_i^{k+1} - u_i^k) = \frac{\alpha}{h^2}\left(u_{i-1}^{k+1} - 2u_i^{k+1} + u_{i+1}^{k+1}\right) + q_i^{k+1}. \tag{22}$$

After rearrangement we obtain

$$-r\, u_{i-1}^{k+1} + (1+2r)u_i^{k+1} - r\, u_{i+1}^{k+1} = u_i^k + \tau q_i^{k+1}. \tag{23}$$

This leads to a system of linear equations for the unknown values $u_1^{k+1}, \ldots, u_{N-1}^{k+1}$. The truncation error is again $O(\tau + h^2)$.

3.3 Crank–Nicolson method

We approximate the equation at the midpoint $(x_i, t_k + \tau/2)$:

$$\begin{aligned}\frac{1}{\tau}(u_i^{k+1} - u_i^k) = \frac{1}{2}\Bigg[&\frac{\alpha}{h^2}\left(u_{i-1}^k - 2u_i^k + u_{i+1}^k\right) \\ &+ \frac{\alpha}{h^2}\left(u_{i-1}^{k+1} - 2u_i^{k+1} + u_{i+1}^{k+1}\right)\Bigg] \\ &+ q_i^{k+1/2}.\end{aligned} \tag{24}$$

This yields

$$\begin{aligned}&-\frac{r}{2}u_{i-1}^{k+1} + (1+r)u_i^{k+1} - \frac{r}{2}u_{i+1}^{k+1} = \\ &\frac{r}{2}u_{i-1}^k + (1-r)u_i^k + \frac{r}{2}u_{i+1}^k + \tau q_i^{k+1/2}.\end{aligned} \tag{25}$$

The truncation error in this case is $O(\tau^2 + h^2)$, so the temporal accuracy is one order higher than in the previous methods.

3.4 Matrix formulation and stability

We start with the explicit method and denote U^k as the values of the approximate solution in the k-th time layer, i.e. $U^k = (u_1^k, u_2^k, \ldots, u_{N-1}^k)^T$. For $i = 1$ we have the equation

$$u_1^{k+1} = r\, u_0^k + (1-2r)u_1^k + r\, u_2^k + \tau q_i^k, \tag{26}$$

but the value u_0^k is given by the boundary condition; similarly, the value u_N^k is given. These values can therefore be incorporated into the function q, and we define $Q^k = (\tau q_1^k + r g_0^k, \tau q_2^k, \ldots, \tau q_{N-2}^k, \tau q_{N-1}^k + r g_1^k)^T$. Let I denote the identity matrix and let M be a tridiagonal matrix of order $N-1$ with 2 on the main diagonal and -1 on the sub- and super-diagonals:

$$M = \begin{pmatrix} 2 & -1 & 0 & 0 & \cdots & 0 \\ -1 & 2 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 2 & -1 & \cdots & 0 \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ 0 & 0 & \cdots & 0 & -1 & 2 \end{pmatrix} \tag{27}$$

Then the explicit method can be written in matrix form as

$$U^{k+1} = (I - rM)\, U^k + Q^k. \tag{28}$$

For the implicit method and the Crank–Nicolson method we can use the same vector and matrix notation, obtaining systems that can be written as

$$(I + rM)U^{k+1} = U^k + Q^{k+1} \tag{29}$$

for the implicit method, and

$$(I + \frac{r}{2}M)U^{k+1} = (I - \frac{r}{2}M)U^k + Q^{k+1/2} \tag{30}$$

for the Crank–Nicolson method. In both cases we obtain systems with a tridiagonal coefficient matrix, which can be solved using, for example, LU decomposition or the Thomas algorithm (see [10] or [11]).

When studying the stability of these methods, we are interested in the propagation of errors between time levels. In general, all methods can be written as

$$U^{k+1} = T\, U^k + S^k. \tag{31}$$

If the initial condition is given with some error, even a very small one, the error may grow during computation depending on the properties of the matrix T. It is clear that this will not happen if its spectral radius does not exceed one, i.e. if for all eigenvalues λ it holds $|\lambda| \leq 1$. Therefore, we first focus on the eigenvalues and eigenvectors of the matrix M.

The matrix M is a tridiagonal Toeplitz matrix of order $N-1$. A direct computation shows that its eigenvectors are

$$\boldsymbol{v}_k = \left(\sin\frac{k\pi}{N}, \sin\frac{2k\pi}{N}, \ldots, \sin\frac{(N-1)k\pi}{N}\right)^T, \quad k = 1, \ldots, N-1, \tag{32}$$

with corresponding eigenvalues $\lambda_k = 2(1 - \cos\frac{k\pi}{N})$. From this it is clear that the eigenvalues of M lie in the interval $(0, 4)$, and for large N they can lie arbitrarily close to the endpoints of this interval. Since M is

symmetric with positive eigenvalues, it is positive definite.

Let T_E denote the time-stepping matrix for the explicit method, i.e. $T_E = I - rM$. From the above discussion it follows that the eigenvalues of T_E lie in the interval $(1-4r, 1)$, so its spectral radius is less than or equal to one for $r \leq \frac{1}{2}$. This is therefore the stability condition for the explicit method.

For the implicit method we obtain a linear system with coefficient matrix $I+rM$. Since the eigenvalues of M are positive, the eigenvalues of $I + rM$ are greater than one, and this matrix is regular (invertible). The transition matrix is therefore $T_I = (I + rM)^{-1}$ and its eigenvalues are always less than one. Hence, the implicit method is unconditionally stable, meaning that no restriction is imposed on the parameter r.

The transition matrix for the Crank–Nicolson method can be written as $T_{CN} = (I + \frac{r}{2}M)^{-1}(I - \frac{r}{2}M)$, since the matrix $I + \frac{r}{2}M$ is invertible. If λ is an eigenvalue of M, then the corresponding eigenvalue of T_{CN} is

$$\lambda_{CN} = \frac{1 - \frac{r}{2}\lambda}{1 + \frac{r}{2}\lambda} = \frac{2 - r\lambda}{2 + r\lambda} = \frac{4}{2 + r\lambda} - 1. \quad (33)$$

Since λ lies in $(0, 4)$, the fraction is positive and less than 2, hence $|\lambda_{CN}| < 1$. Thus, the Crank–Nicolson method is also unconditionally stable.

3.5 Method of lines

The idea of the method of lines is to discretize only the spatial variable, which yields a system of ordinary differential equations in time. The method can also be applied to nonlinear equations of the form

$$u_t = f(x, t, u_x, u_{xx}), \quad 0 < x < 1, \quad t > 0. \quad (34)$$

with initial condition $u(x, 0) = u^0(x)$, $x \in [0, 1]$ and Dirichlet boundary conditions $u(i, t) = g_i(t)$, $t \geq 0$, $i \in \{0, 1\}$.

We define the spatial grid as before and denote $u_i(t) = u(x_i, t)$, i.e. the restriction of u to the line $x = x_i$.

We use standard finite difference approximations of the spatial derivatives of u, while time derivatives of u_i are ordinary derivatives in time. This leads to a system of first-order ordinary differential equations

$$\begin{aligned} u_i' &= f(x_i, t, \frac{u_{i+1} - u_{i-1}}{2h}, \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}) \\ &= f_i(t, u_{i-1}, u_i, u_{i+1}),\ i = 1, \ldots, N-1 \end{aligned} \quad (35)$$

with initial conditions $u_i(0) = u^0(x_i)$, while u_0 and u_N are given by boundary conditions.

For the heat equation we obtain

$$u_i' = f_i(t, u_{i-1}, u_i, u_{i+1}), \quad \text{where} \quad (36)$$

$$f_i = \frac{\alpha}{h^2}(u_{i-1} - 2u_i + u_{i+1}) + q_i, \quad (37)$$

$$q_i = q(x_i, t) \quad (38)$$

Due the matrix notation we introduce:

$$\boldsymbol{u} = \boldsymbol{u}(t) = (u_1, \ldots, u_{N-1})^T, \quad (39)$$

$$\boldsymbol{q} = \boldsymbol{q}(t) = (q_1 + \tfrac{\alpha}{h^2} g_0, q_2, \ldots, q_{N-2}, q_{N-1} + \tfrac{\alpha}{h^2} g_1)^T, \quad (40)$$

and we can write the system as

$$\boldsymbol{u}' = -\frac{\alpha}{h^2} M\boldsymbol{u} + \boldsymbol{q} = \boldsymbol{f}(t, \boldsymbol{u}) \quad (41)$$

with initial condition $\boldsymbol{u}_{(0)=(u^0(x_1),\ldots,u^0(x_{N-1}))^T}$. Moreover, the variable t appears explicitly only in the functions q_i, so for the homogeneous heat equation we obtain an autonomous system.

To obtain the numerical solution we discretize time with step τ, i.e. $t_j = j\tau$, and denote by $\boldsymbol{U}^j$ the vector of values of u_i at time t_j, i.e. $\boldsymbol{U}^j = \boldsymbol{u}(t_j)$.

The simplest numerical methods for ODEs are the well-known Euler methods (see [5]), including explicit Euler, implicit Euler, and the trapezoidal rule. Applying these methods to system (41) yields the explicit, implicit, and Crank–Nicolson methods introduced above. We now focus on how Runge–Kutta methods can be used for solving (41).

4 Runge–Kutta methods for the heat equation

4.1 Explicit methods

Let us consider the Butcher tableau

$$\begin{array}{c|ccccc} c_2 & a_{21} & & & & \\ c_3 & a_{31} & a_{32} & & & \\ \vdots & \vdots & & & & \\ c_s & a_{s1} & a_{s2} & \ldots & a_{s,s-1} & \\ \hline & b_1 & b_2 & \ldots & b_{s-1} & b_s \end{array} \quad (42)$$

The RK method applied to equation (41) takes the form

$$\boldsymbol{U}^{j+1} = \boldsymbol{U}^j + \sum_{l=1}^{s} b_l k_l, \quad (43)$$

$$k_l = \tau \boldsymbol{f}(t_j + c_l\tau, \boldsymbol{U}^j + \sum_{n=1}^{l-1} a_{ln} k_n). \quad (44)$$

Furthermore we have

$$\begin{aligned} \tau \boldsymbol{f}(t_j, \boldsymbol{U}^j) &= -\tfrac{\alpha\tau}{h^2} M\boldsymbol{U}^j + \tau \boldsymbol{q}(t_j) \\ &= -rM\boldsymbol{U}^j + \tau \boldsymbol{q}(t_j). \end{aligned} \quad (45)$$

Let $M_1 = -rM$, $\boldsymbol{q}_l = \tau \boldsymbol{q}(t_j + c_l\tau)$. Then

$$\begin{aligned} k_1 &= \tau \boldsymbol{f}(t_j, \boldsymbol{U}^j) \\ &= -rM\boldsymbol{U}^j + \tau \boldsymbol{q}(t_j) = M_1\boldsymbol{U}^j + \boldsymbol{q}_1 \end{aligned} \tag{46}$$

$$\begin{aligned} k_2 &= \tau \boldsymbol{f}(t_j + c_2\tau, \boldsymbol{U}^j + a_{21}k_1) \\ &= M_1(\boldsymbol{U}^j + a_{21}k_1) + \boldsymbol{q}_2 \\ &= [M_1 + a_{21}M_1^2]\boldsymbol{U}^j + a_{21}M_1\boldsymbol{q}_1 + \boldsymbol{q}_2 \end{aligned} \tag{47}$$

$$\begin{aligned} k_3 &= \tau \boldsymbol{f}(t_j + c_3\tau, \boldsymbol{U}^j + a_{31}k_1 + a_{32}k_2) \\ &= [M_1 + (a_{31} + a_{32})M_1^2 + a_{32}a_{21}M_1^3]\boldsymbol{U}^j \\ &\quad +(a_{31}M_1 + a_{32}a_{21}M_1^2)\boldsymbol{q}_1 \\ &\quad +a_{32}M_1\boldsymbol{q}_2 + \boldsymbol{q}_3 \end{aligned} \tag{48}$$

$$\begin{aligned} k_4 &= \tau \boldsymbol{f}(t_j + c_4\tau, \boldsymbol{U}^j + a_{41}k_1 + a_{42}k_2 + a_{43}k_3) \\ &= [M_1 + (a_{41} + a_{42} + a_{43})M_1^2 \\ &\quad +(a_{42}a_{21} + a_{43}a_{31} + a_{43}a_{32})M_1^3 \\ &\quad +a_{43}a_{32}a_{21}M_1^4]\boldsymbol{U}^j \\ &\quad +(a_{41}M_1 + (a_{42}a_{21} + a_{43}a_{31})M_1^2 \\ &\quad +a_{43}a_{32}a_{21}M_1^3)\boldsymbol{q}_1 \\ &\quad +(a_{42}M_1 + a_{43}a_{32}M_1^2)\boldsymbol{q}_2 \\ &\quad +a_{43}M_1\boldsymbol{q}_3 + \boldsymbol{q}_4 \\ &\vdots \end{aligned} \tag{49}$$

These expressions can of course be simplified using order conditions. Then, for instance, the classical Runge–Kutta method is given by

$$\begin{aligned} \boldsymbol{U}^{j+1} &= (I + M_1 + \frac{1}{2}M_1^2 + \frac{1}{6}M_1^3 + \frac{1}{24}M_1^4)\boldsymbol{U}^j \\ &\quad +(\frac{1}{6}I + \frac{1}{6}M_1 + \frac{1}{12}M_1^2 + \frac{1}{24}M_1^3)\boldsymbol{q}_1 \\ &\quad +(\frac{1}{3}I + \frac{1}{6}M_1 + \frac{1}{12}M_1^2)\boldsymbol{q}_2 \\ &\quad +(\frac{1}{3}I + \frac{1}{6}M_1)\boldsymbol{q}_3 + \frac{1}{6}\boldsymbol{q}_4 \end{aligned} \tag{50}$$

The homogeneous equations the expressions can be simplified further. For methods up to order 4 with the same number of stages we obtain

$$\begin{aligned} \boldsymbol{U}^{j+1} &= (I + M_1 + \tfrac{1}{2}M_1^2)\boldsymbol{U}^j \\ &\text{for } p = 2 \\ \boldsymbol{U}^{j+1} &= (I + M_1 + \tfrac{1}{2}M_1^2 + \tfrac{1}{6}M_1^3)\boldsymbol{U}^j \\ &\text{for } p = 3 \\ \boldsymbol{U}^{j+1} &= (I + M_1 + \tfrac{1}{2}M_1^2 + \tfrac{1}{6}M_1^3 + \tfrac{1}{24}M_1^4)\boldsymbol{U}^j \\ &\text{for } p = 4 \end{aligned} \tag{51}$$

Stability condition:
The eigenvalues of matrix M lie in $(0, 4)$, hence the eigenvalues of $M_1 = -rM$ lie in $(-4r, 0)$. The amplification matrix is given by a polynomial P, and its spectral radius is less than one if all eigenvalues lie in the region where $|P| < 1$, which defines the stability intervals for explicit methods up to order 4. Stability intervals and conditions for r are presented in Table 2.

Table 2: Stability intervals and conditions for r

$s(= p)$	1	2
interval	$(-2, 0)$	$(-2, 0)$
$r \leq$	0.5	0.5

$s(= p)$	3	4
interval	$(-2.513, 0)$	$(-2.785, 0)$
$r \leq$	0.628	0.696

Source: created by the author.

Computational remark
The matrix M_1 is tridiagonal, but its powers are banded. Therefore it is not efficient to compute these powers explicitly; instead, one should repeatedly multiply the vector by M_1, preserving the tridiagonal structure.

Truncation error
If an RK method of order p is used, the truncation error is $O(\tau^p + h^2)$. However, due to the stability restriction, we have $\tau = Ch^2$, as follows from the constraint on r. Thus the dominant term in the error is h^2, since $O(\tau^p + h^2) = O(h^{2p} + h^2) = O(h^2)$. Hence explicit RK methods cannot provide a higher order of accuracy on the same grid than the basic methods used for the heat equation.

4.2 Implicit methods

Let us consider a method given by the Butcher tableau

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \dots & a_{1,s} \\ c_2 & a_{21} & a_{22} & \dots & a_{2,s} \\ \vdots & \vdots & \vdots & & \vdots \\ c_s & a_{s1} & a_{s2} & \dots & a_{s,s} \\ \hline & b_1 & b_2 & \dots & b_s \end{array} \tag{52}$$

For values k_l we have the equation

$$k_l = \tau f(t_j + c_l\tau, \boldsymbol{U}^j + \sum_{n=1}^{s} a_{ln}k_n). \tag{53}$$

Furthermore,

$$\tau \boldsymbol{f}(t_j + c_l\tau, \boldsymbol{V}) = M_1\boldsymbol{V} + \boldsymbol{q}_l, \tag{54}$$

so that

$$k_l = \sum_{n=1}^{s} a_{ln} M_1 k_n + M_1 \boldsymbol{U}^j + \boldsymbol{q}_l. \qquad (55)$$

The system of equations for all stage vectors k_l can be written in block form:

$$\begin{pmatrix} k_1 \\ k_2 \\ \vdots \\ k_s \end{pmatrix} = \begin{pmatrix} a_{11}M_1 & a_{12}M_1 & \cdots & a_{1s}M_1 \\ a_{21}M_1 & a_{22}M_1 & \cdots & a_{2s}M_1 \\ \vdots & \vdots & & \vdots \\ a_{s1}M_1 & a_{s2}M_1 & \cdots & a_{ss}M_1 \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ \vdots \\ k_s \end{pmatrix} + \boldsymbol{Q} \qquad (56)$$

where

$$\boldsymbol{Q} = \begin{pmatrix} \boldsymbol{q}_1 + M_1\boldsymbol{U}^j \\ \boldsymbol{q}_2 + M_1\boldsymbol{U}^j \\ \vdots \\ \boldsymbol{q}_s + M_1\boldsymbol{U}^j \end{pmatrix} = \begin{pmatrix} \boldsymbol{q}_1 - rM\boldsymbol{U}^j \\ \boldsymbol{q}_2 - rM\boldsymbol{U}^j \\ \vdots \\ \boldsymbol{q}_s - rM\boldsymbol{U}^j \end{pmatrix}. \qquad (57)$$

Thus, the vectors $k_1, \ldots, k_s$ satisfy the block system

$$\begin{pmatrix} I - a_{11}M_1 & -a_{12}M_1 & \cdots & -a_{1s}M_1 \\ -a_{21}M_1 & I - a_{22}M_1 & \cdots & -a_{2s}M_1 \\ \vdots & \vdots & & \vdots \\ -a_{s1}M_1 & -a_{s2}M_1 & \cdots & I - a_{ss}M_1 \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ \vdots \\ k_s \end{pmatrix} = \boldsymbol{Q} \qquad (58)$$

or equivalently

$$\begin{pmatrix} I + ra_{11}M & ra_{12}M & \cdots & ra_{1s}M \\ ra_{21}M & I + ra_{22}M & \cdots & ra_{2s}M \\ \vdots & \vdots & & \vdots \\ ra_{s1}M & ra_{s2}M & \cdots & I + ra_{ss}M \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ \vdots \\ k_s \end{pmatrix} = \boldsymbol{Q} \qquad (59)$$

We thus obtain a block system of equations in which all blocks are tridiagonal matrices, so the system matrix is sparse.

4.3 Block solution algorithm

Let us now focus on methods for solving system (59). Computing the inverse matrix is very inefficient, since it would destroy the sparsity structure. If the Runge–Kutta coefficient matrix A (see the corresponding Butcher tableau) were symmetric, it would be possible in particular cases to determine whether the system matrix is positive definite, and then apply the conjugate gradient method.

In the general case, we consider a block Gauss–Seidel iteration method. Its derivation can proceed as follows.

We set the initial iteration for k_j, $j > 1$, to zero vectors, i.e. $k_2^{(0)} = k_3^{(0)} = \cdots = k_s^{(0)} = \boldsymbol{0}$. The first block equation then has the form

$$(I + ra_{11}M)k_1 = \boldsymbol{q}_1 - rM\boldsymbol{U}^j, \qquad (60)$$

from which we compute the first iterate $k_1^{(1)}$, for example using the Thomas algorithm. For $a_{11} \geq 0$, the matrix $I + ra_{11}M$ is nonsingular, so $k_1^{(1)}$ is uniquely determined.

The next equation has the form

$$ra_{21}Mk_1 + (I + ra_{22}M)k_2 = \boldsymbol{q}_2 - rM\boldsymbol{U}^j. \qquad (61)$$

We substitute the already computed iterate for k_1 and compute $k_2^{(1)}$, and proceed similarly for the remaining components. After obtaining $k_1^{(1)}, \ldots, k_s^{(1)}$, we return to the first equation and continue similarly to obtain $k_1^{(2)}$, etc.

For a block formulation, we split the system matrix into two block parts: the lower block-triangular part, denoted L, and the strictly upper block part, denoted U. All remaining entries are zero. Thus

$$L = \begin{pmatrix} I + ra_{11}M & 0 & 0 & \cdots & 0 \\ ra_{21}M & I + ra_{22}M & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ ra_{s1}M & ra_{s2}M & ra_{s3}M & \cdots & I + ra_{ss}M \end{pmatrix}, \qquad (62)$$

$$U = \begin{pmatrix} 0 & ra_{12}M & ra_{13}M & \cdots & ra_{1s}M \\ 0 & 0 & ra_{23}M & \cdots & ra_{2s}M \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}. \qquad (63)$$

The upper-triangular part U is moved to the right-hand side, where previous iterates are substituted. On the left-hand side we obtain new iterates, which can be written as

$$L \begin{pmatrix} k_1^{(m+1)} \\ k_2^{(m+1)} \\ \vdots \\ k_s^{(m+1)} \end{pmatrix} = \boldsymbol{Q} - U \begin{pmatrix} k_1^{(m)} \\ k_2^{(m)} \\ \vdots \\ k_s^{(m)} \end{pmatrix}. \qquad (64)$$

If all diagonal entries of A are nonnegative, then L is nonsingular, and the iteration system (64) can be written in explicit form as

$$\begin{pmatrix} k_1^{(m+1)} \\ k_2^{(m+1)} \\ \vdots \\ k_s^{(m+1)} \end{pmatrix} = W \begin{pmatrix} k_1^{(m)} \\ k_2^{(m)} \\ \vdots \\ k_s^{(m)} \end{pmatrix} + L^{-1}\boldsymbol{Q}, \qquad (65)$$

where the iteration matrix is $W = -L^{-1}U$.

This formulation is mainly of theoretical importance, since L^{-1} is no longer sparse. It is

known that the iteration converges if and only if all eigenvalues of W have modulus less than one. However, in the general case this is difficult to verify. Although the eigenvalues of U are known (they are zero), and the eigenvalues of M are also known (hence those of L and L^{-1} as well), this is not sufficient to determine the eigenvalues of the product $L^{-1}U$ unless the matrices share eigenvectors, which need not be the case.

A sufficient condition for convergence is the strict diagonal dominance of the original matrix of the system, i.e. the matrix

$$\begin{pmatrix} I+ra_{11}M & ra_{12}M & \cdots & ra_{1s}M \\ ra_{21}M & I+ra_{22}M & \cdots & ra_{2s}M \\ \vdots & \vdots & & \vdots \\ ra_{s1}M & ra_{s2}M & \cdots & I+ra_{ss}M \end{pmatrix}. \tag{66}$$

Row or column diagonal dominance of this matrix yields the following inequalities that can be easily derived:

$$4r\sum_{j\neq i}|a_{ij}|<1, \text{ for all } i, \tag{67}$$

or

$$4r\sum_{i\neq j}|a_{ij}|<1, \text{ for all } j. \tag{68}$$

We will now show a procedure for determining the convergence of the mentioned iterative method applicable to a specific RK method. This procedure shows the block form of the inversion of the matrix L and therefore of the matrix W, too. The procedure is demonstrated on a two-stage and three-stage methods as in this cases the iterative matrix W doesn't take complicated expression. The key property is that all blocks have the same eigenvectors as the matrix M, which allows us to determine the eigenvalues of the matrix W.

For two-stage method we have

$$L = \begin{pmatrix} I+ra_{11}M & 0 \\ ra_{21}M & I+ra_{22}M \end{pmatrix}, \tag{69}$$

$$U = \begin{pmatrix} 0 & ra_{12}M \\ 0 & 0 \end{pmatrix}, \tag{70}$$

for brevity we use the notation

$$L=\begin{pmatrix} L_{11} & 0 \\ L_{21} & L_{22} \end{pmatrix}, \quad U=\begin{pmatrix} 0 & U_{12} \\ 0 & 0 \end{pmatrix}. \tag{71}$$

It is important that all the above blocks, as well as their inverses, have the same eigenvectors as the matrix M. As for eigenvalues, they transform similarly as for matrix M, so for example if λ is an eigenvalue of matrix M, then $1+ra_{11}\lambda$ is an eigenvalue of matrix L_{11} and $\frac{1}{1+ra_{11}\lambda}$ is an eigenvalue of its inverse.

Let us assume again that the entries a_{11} and a_{22} are nonnegative, so that the matrix L is nonsingular, as well as the matrices L_{11} and L_{22}. By direct computation one can verify that

$$L^{-1}=\begin{pmatrix} L_{11}^{-1} & 0 \\ -L_{22}^{-1}L_{21}L_{11}^{-1} & L_{22}^{-1} \end{pmatrix} \tag{72}$$

and then

$$W=-L^{-1}U=\begin{pmatrix} 0 & -L_{11}^{-1}U_{12} \\ 0 & L_{22}^{-1}L_{21}L_{11}^{-1}U_{12} \end{pmatrix}. \tag{73}$$

The matrix W is of order $2(N-1)$, $N-1$ of its eigenvalues are equal to zero, and the corresponding eigenvectors are $(\boldsymbol{v},\mathbf{0})^T$, where $\boldsymbol{v}$ can be taken as eigenvectors of matrix M. To determine the remaining eigenvalues of matrix W, we must first find its eigenvectors. Let us use the vector $(\boldsymbol{v}, x\cdot\boldsymbol{v})^T$, where $\boldsymbol{v}$ is an eigenvector of matrix M and x is a suitable constant. For λ being the eigenvalue corresponding to $\boldsymbol{v}$ we have

$$\begin{aligned} W\cdot\begin{pmatrix} \boldsymbol{v} \\ x\boldsymbol{v} \end{pmatrix} &= \begin{pmatrix} -x\frac{ra_{12}\lambda}{1+ra_{11}\lambda}\boldsymbol{v} \\ \frac{r^2a_{12}a_{21}\lambda^2}{(1+ra_{11}\lambda)(1+ra_{22}\lambda)}x\boldsymbol{v} \end{pmatrix} \\ &= \frac{r^2a_{12}a_{21}\lambda^2}{(1+ra_{11}\lambda)(1+ra_{22}\lambda)}\begin{pmatrix} \boldsymbol{v} \\ x\boldsymbol{v} \end{pmatrix} \end{aligned} \tag{74}$$

for $x=-\frac{ra_{21}\lambda}{1+ra_{22}\lambda}$. The eigenvalues of matrix W are therefore given by

$$\begin{aligned} \mu &= \frac{r^2a_{12}a_{21}\lambda^2}{(1+ra_{11}\lambda)(1+ra_{22}\lambda)} \\ &= \frac{r^2\lambda^2a_{12}a_{21}}{1+r\lambda(a_{11}+a_{22})+r^2\lambda^2a_{11}a_{22}}, \end{aligned} \tag{75}$$

where $\lambda\in(0,4)$ are eigenvalues of matrix M. Since we assumed that a_{11} and a_{22} are nonnegative, a sufficient condition for $|\mu|<1$ is, for example, the inequality $|a_{12}a_{21}|\leq a_{11}a_{22}$. If the product $a_{12}a_{21}$ is also nonnegative, then the inequality $\mu<1$ is equivalent to

$$1+r\lambda\cdot\mathrm{tr}(A)+r^2\lambda^2\cdot\det(A)>0, \tag{76}$$

from which further sufficient conditions (for example $\det(A)\geq 0$) can be derived that guarantee the above condition for μ, and thus also convergence of the iterative process.

Now let us focus on the three-stage method. Here

$$L=\begin{pmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{pmatrix}, \tag{77}$$

$$U=\begin{pmatrix} 0 & U_{12} & U_{13} \\ 0 & 0 & U_{23} \\ 0 & 0 & 0 \end{pmatrix}. \tag{78}$$

The positivity of the diagonal members of the matrix A again causes the non-singularity of the matrix L.

$$L^{-1} = \begin{pmatrix} L_{11}^{-1} & 0 & 0 \\ -L_{22}^{-1}L_{21}L_{11}^{-1} & L_{22}^{-1} & 0 \\ -L_{33}^{-1}L_{31}L_{11}^{-1}+ & -L_{33}^{-1}L_{32}L_{22}^{-1} & L_{33}^{-1} \\ +L_{33}^{-1}L_{32}L_{22}^{-1}L_{21}L_{11}^{-1} & & \end{pmatrix} \tag{79}$$

and

$$W = \begin{pmatrix} 0 & -L_{11}^{-1}U_{12} & -L_{11}^{-1}U_{13} \\ 0 & L_{22}^{-1}L_{21}L_{11}^{-1}U_{12} & L_{22}^{-1}L_{21}L_{11}^{-1}U_{13}- \\ & & -L_{22}^{-1}U_{23} \\ 0 & L_{33}^{-1}L_{31}L_{11}^{-1}U_{12}- & L_{33}^{-1}L_{31}L_{11}^{-1}U_{13}- \\ & -L_{33}^{-1}L_{32}L_{22}^{-1}L_{21}L_{11}^{-1}U_{12} & -L_{33}^{-1}L_{32}L_{22}^{-1}L_{21}L_{11}^{-1}U_{13}+ \\ & & +L_{33}^{-1}L_{32}L_{22}^{-1}U_{23} \end{pmatrix}. \tag{80}$$

We have again $N-1$ zeros eigenvalues corresponding eigenvectors $(\boldsymbol{v}, \mathbf{0}, \mathbf{0})^T$ for $\boldsymbol{v}$ being the eigenvectors of matrix M.

The other eigenvectors of W are $(\boldsymbol{v}, x \cdot \boldsymbol{v}, y \cdot \boldsymbol{v})^T$ for suitable constants x and y. Solving the equation

$$W \cdot \begin{pmatrix} \boldsymbol{v} \\ x\boldsymbol{v} \\ y\boldsymbol{v} \end{pmatrix} = \mu \begin{pmatrix} \boldsymbol{v} \\ x\boldsymbol{v} \\ y\boldsymbol{v} \end{pmatrix} \tag{81}$$

is very difficult, so the library for symbolic calculations in MATLAB was used, but it would certainly be possible to use other suitable software.

The equation has two triples of solutions μ, x, y depending on eigenvalues λ of the matrix M (and of course also on r and a_{ij}), so it is possible to determine all the eigenvalues of matrix W. But the general expression is too complex, for example, it contains the square root of the sum of 37 terms. Therefore, it is not possible to formulate a general convergence condition for the proposed algorithm in this case.

4.4 Example

As an example we consider the equation $u_t = \frac{1}{3\pi^2}u_{xx}$ with homogeneous boundary conditions and initial condition $u^0(x) = \sin(\pi x)$. Since the exact solution is known to be $u(x,t) = \sin(\pi x)e^{-\frac{t}{3}}$, we can compute the error of the numerical solution. The equation will be solved using the Gauss–Legendre method. Let us recall the Butcher tableau of this method:

$$\begin{array}{c|cc} \frac{1}{2}-\frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4}-\frac{\sqrt{3}}{6} \\ \frac{1}{2}+\frac{\sqrt{3}}{6} & \frac{1}{4}+\frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array} \tag{82}$$

It holds that $|a_{12}a_{21}| = \frac{1}{48} < \frac{1}{16} = a_{11}a_{22}$, so the proposed iterative method will converge.

We compare the proposed method with the Crank–Nicolson method. We compute an approximate solution up to time $T = 5$. We gradually decrease the step size h and for each value we determine the maximum error for decreasing time steps. The results are summarized in the Table 3, Table 4, 5 and 6.

Table 3: Accuracy of the approximate solution depending on the time step for spatial step $h = 0.05$

τ	GL	CN
0.5	$7.57 \cdot 10^{-4}$	$9.41 \cdot 10^{-5}$
0.1	$7.57 \cdot 10^{-4}$	$7.23 \cdot 10^{-4}$
0.05	$7.57 \cdot 10^{-4}$	$7.48 \cdot 10^{-4}$
0.01	$7.57 \cdot 10^{-4}$	$7.56 \cdot 10^{-4}$

Source: created by the author.

Table 4: Accuracy of the approximate solution depending on the time step for spatial step $h = 0.01$

τ	GL	CN
0.5	$3.07 \cdot 10^{-5}$	$8.24 \cdot 10^{-4}$
0.1	$3.03 \cdot 10^{-5}$	$3.80 \cdot 10^{-6}$
0.05	$3.03 \cdot 10^{-5}$	$2.17 \cdot 10^{-5}$
0.01	$3.03 \cdot 10^{-5}$	$2.99 \cdot 10^{-5}$

Source: created by the author.

Table 5: Accuracy of the approximate solution depending on the time step for spatial step $h = 0.005$

τ	GL	CN
0.5	$7.96 \cdot 10^{-6}$	$8.47 \cdot 10^{-4}$
0.1	$7.57 \cdot 10^{-6}$	$2.65 \cdot 10^{-5}$
0.05	$7.56 \cdot 10^{-6}$	$9.51 \cdot 10^{-7}$
0.01	$7.56 \cdot 10^{-6}$	$7.22 \cdot 10^{-6}$

Source: created by the author.

From the results it is clearly seen that for relatively large values of h the Crank–Nicolson method is as good as the proposed method, in some cases even slightly better. The reason is that the error is dominated by h. When its value is decreased, the time step τ becomes dominant, especially its larger values. In this case the proposed method gives better results, even by several orders of magnitude. The results of both methods again coincide for small time steps, when the spatial step again dominates the error.

5 Conclusion

Although general conditions for the convergence of the proposed algorithm have not been derived,

Table 6: Accuracy of the approximate solution depending on the time step for spatial step $h = 0.001$

$h = 0.001$

τ	GL	CN
0.5	$6.97 \cdot 10^{-7}$	$8.54 \cdot 10^{-4}$
0.1	$3.03 \cdot 10^{-7}$	$3.38 \cdot 10^{-5}$
0.05	$3.03 \cdot 10^{-7}$	$8.21 \cdot 10^{-6}$
0.01	$3.03 \cdot 10^{-7}$	$3.81 \cdot 10^{-8}$

Source: created by the author.

the presented procedure allows in specific cases to determine the eigenvalues of the iteration matrix and thus reliably determine whether the process will converge.

The numerical experiments confirm the theoretical conclusions derived in this paper regarding the use of Runge–Kutta methods for solving the heat equation. This may be useful in situations where the time step cannot be changed, or in computationally expensive calculations where increasing τ can significantly speed up the computation while preserving accuracy when using a suitable RK method of appropriate order. It is, however, essential to use A-stable methods and suitable algorithms for tridiagonal matrices.

References:

[1] Manshoor, B., Salleh, H., Khalid, A., Sayed Abdelaal, M.A.: Method of Lines and Runge–Kutta Method in Solving Partial Differential Equation for Heat Equation. *Journal of Complex Flow* 3(1), 21–25, 2021. `http://eprints.uthm.edu.my/2702/` [Access Date: 18/09/2026]

[2] Tsai, A.Y.J., Chan, R.P.K., Wang, S.: Two-derivative Runge–Kutta methods for PDEs using a novel discretization approach. *Numerical Algorithms* 65(3), 687–703, 2014. doi:10.1007/s11075-014-9823-2

[3] Lubich, Ch., Ostermann, A.: Runge–Kutta methods for parabolic equations and convolution quadrature. *Mathematics of Computation* 60(201), 105–131, 1993. doi:10.1090/S0025-5718-1993-1153166-7

[4] Butcher, J.C.: *Numerical Methods for Ordinary Differential Equations*, 2nd ed., Wiley, 2008. ISBN: 978-0470723357, doi:10.1002/9780470753767

[5] Hairer, E., Nørsett, S.P., Wanner, G.: *Solving Ordinary Differential Equations I: Nonstiff Problems*, 2nd ed., Springer, Berlin, 1993. ISBN: 978-3-540-56670-0, doi:10.1007/978-3-540-78862-1

[6] Hairer, E., Wanner, G.: *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*, 2nd ed., Springer, Berlin, 1996. ISBN 978-3-540-60452-5, doi:10.1007/978-3-642-05221-7

[7] Butcher, J.C.: On Runge–Kutta processes of high order. *Journal of the Australian Mathematical Society* 4(2), 179–194, 1964. doi:10.1017/S1446788700023387

[8] Crank, J., Nicolson, P.: A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Proceedings of the Cambridge Philosophical Society* 43(1), 50–67, 1947. doi:10.1017/S0305004100023197

[9] LeVeque, R.J.: *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems*. SIAM, Philadelphia, 2007. ISBN 978-0898717839, doi:10.1137/1.9780898717839

[10] Lee W.T.: Tridiagonal matrices: Thomas algorithm, MS6021, Scientific Computation, University of Limerick, 2011 `https://sites.nd.edu/gfu/files/2019/10/ms6021_thomas.pdf` [Access Date: 18/09/2026]

[11] Sultanian, B. K.: Gas Turbines: Internal Flow Systems Modeling, Appendix E: Thomas Algorithm for Solving a Tridiagonal System of Linear Algebraic Equations, 337–339 (Cambridge Aerospace Series). Cambridge: Cambridge University Press, 2018. ISBN 978-1107170094, doi:10.1017/9781316755686.012

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The author contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The author has no conflict of interest to declare that is relevant to the content of this article.

Creative Commons Attribution License 4.0 (Attribution 4.0 International, CC BY 4.0)

This article is published under the terms of the Creative Commons Attribution License 4.0
https://creativecommons.org/licenses/by/4.0/deed.en_US