MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
DNIPROVSKY STATE TECHNICAL UNIVERSITY

Dmytro Redchyts

LECTURE NOTES 3
"BASIC APPROACHES TO SOLVING PROBLEMS IN MATHEMATICAL PHYSICS"

on discipline

"SPECIAL TOPICS OF APPLIED MATHEMATICS"

for first-level (bachelor's) higher education students
in specialty 113 “Applied Mathematics”

Kamyanske
2025

CONTENTS

TOPIC 3. BASIC APPROACHES TO SOLVING PROBLEMS IN MATHEMATICAL PHYSICS 3

3.1 Preliminary information about calculation techniques 3
3.2 Types of calculation grids 3
3.3 Discretization of the original differential equations. Approximation of derivatives 4
3.3.1 Converting derivatives to discrete algebraic expressions 5
3.3.2 Derivatives in space 6
3.3.3 Derivatives in time 7
3.3.4 Taylor series expansion 8
3.4 Accuracy of the discretization process 10
3.5 Consistency of differential schemes 12
3.5.1 Rounding error 13
3.5.2 Stability of finite-difference schemes 13
3.5.3 Convergence 14
3.5.4 Computational efficiency 14

LIST OF RECOMMENDED REFERENCES 15

TOPIC 3. BASIC APPROACHES TO SOLVING PROBLEMS IN MATHEMATICAL PHYSICS

3.1 Preliminary information about calculation techniques

This section will review some of the basic computational techniques used in solving thermal physics and hydroaerodynamics problems. When solving a specific problem, the original equations and the corresponding boundary conditions will be known. Computational techniques are used to obtain an approximate solution to the original equations with given boundary conditions.

The process of constructing a computational solution consisting of two stages, schematically shown in Fig. 3.1

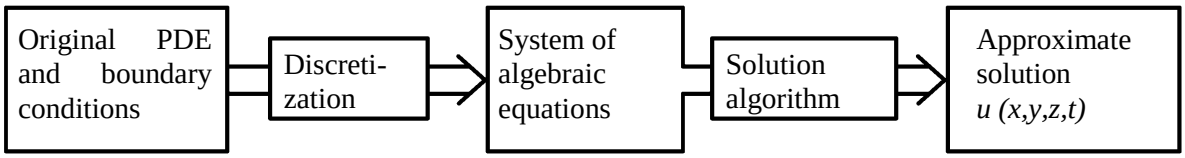

Figure 3.1 – The process of constructing a computational solution

At the first stage of discretization, the partial differential equations describing the continuous process, as well as the auxiliary (boundary and initial) conditions, are transformed into a discrete system of algebraic equations. The discretization process is easily identified if the difference method is used, but it is somewhat less obvious when finite element methods, finite volume methods, and spectral methods are used.

In the process of replacing individual terms of the original equations, which are partial derivatives, with algebraic expressions that connect the nodal values on the finite grid, some error is introduced. No less important than the error in the representation of the differential terms of the original equation is the error in the solution itself.

The second stage of the solution process requires that the equation solving algorithm provides a solution to the system of algebraic equations. Some error may also be introduced at this stage, but it is usually negligible compared to the error introduced at the discretization stage, unless the method is unstable.

3.2 Types of calculation grids

Calculation grids, depending on the shape of the grid cells, are divided into triangular, quadrilateral, etc. Depending on the type of lines or surfaces that limit the cells, grids can be curvilinear or rectilinear. If the sizes of all cells are the same, the grids are called uniform, otherwise they are called non-uniform. If the sizes of the cells do not change over time, the grids are called fixed, otherwise they are called variable. If the grid is formed by the intersection of curvilinear coordinate planes, it is

called structured (regular), otherwise they are called unstructured (irregular). Hybrid grids are possible, when the grid is structured in part of the region, and unstructured in another. Structured grids can be block, when the grid is built separately for different parts of the region, and the connection is performed using interpolation (used when solving problems for regions of complex shape). In structured grids, the cells are quadrilateral in the two-dimensional case and hexagonal in the three-dimensional with curved sides and faces in the general case. An unstructured grid consists of triangles or tetrahedra in the two- and three-dimensional cases. Grids in non-Cartesian coordinate systems are of three types: polar, triangular, and skew. And they can also be formed by elements of other shapes. Structured grids use a simpler data structure and simplify the approximation of the differential problem. Unstructured grids have advantages for problems with complex boundaries.

In some cases, in some parts of the area it is necessary to make more accurate calculations. Then, even for simple calculation areas, a non-uniform grid is used, thickening the nodes in such areas. Adaptive grids are those that are automatically rebuilt over time depending on the behaviour of the function to achieve a predetermined accuracy. They give a particularly significant effect if the area of large gradients changes its position over time.

There are two types of simplest variable grids:

− the variable is uniform when the variable is the time step t , and the spatial steps are the same in each coordinate direction for a fixed time t and changes when moving to the next time layer;

− the variable is non-uniform when, for spatial variables, the distances between neighboring points in each direction are not the same.

After selecting the grid, it is necessary to approximate the derivatives that enter the equations and the boundary and initial conditions by finite differences. The grid nodes that are used in the formulas for the difference approximation of derivatives are called the template.

3.3 Discretization of the original differential equations. Approximation of derivatives

To transform the original partial differential equation (or system of such equations) into a system of algebraic equations (or ordinary differential equations), one can choose one of several options. The most common are the finite difference method, the finite element method, and the finite volume method. The way in which the discretization is performed also depends on whether the time derivatives are considered (as applied to time-dependent problems) or equations containing only spatial derivatives.

In fact, the discretization of time derivatives is almost always done using the finite-difference method. When discretizing spatial derivatives, the finite-difference, finite-element, or finite-volume methods are usually used.

3.3.1 Converting derivatives to discrete algebraic expressions

To illustrate the discretization process, consider the equation that defines the non-stationary process of thermal conductivity in one dimension

$$\frac{\partial \overline{T}}{\partial t} = a \frac{\partial^2 \overline{T}}{\partial x^2}. \tag{3.1}$$

where $\overline{T}$ is the temperature a is the thermal conductivity coefficient. The dash on top symbolizes the exact solution. The characteristic boundary and initial conditions corresponding to the equation have the form

$$\overline{T}(0,t) = b,\ \overline{T}(1,t) = d \tag{3.2}$$

$$\overline{T}(x,0) = T_0(x),\ 0 \le x \le 1 \tag{3.3}$$

The most direct way to discretize is to replace the derivatives with their equivalent finite-difference expressions. For example, the equation (3.1) can be replaced by the following equation, which is algebraic as opposed to a differential equation (3.1)

$$\frac{T_j^{n+1} - T_j^n}{\Delta t} = \frac{a\left(T_{j-1}^n - 2T_j^n + T_{j+1}^n\right)}{\Delta x^2}. \tag{3.4}$$

Step sizes $\Delta t, \Delta x$, as well as the value of the subscript j and superscript n are indicated in Fig. 3.2. In the equation, (3.4) the symbol T_j^n corresponds to the value T at the node (j,n). As follows from the discretization process itself, which transforms the equation (3.1) into the equation (3.4), the problem of finding an exact (continuous) solution $\overline{T}(x,t)$ was replaced by the problem of finding a set of discrete values T_j^n, that is, an approximate solution at each of the nodes (j,n). It is clear that two related errors arise. That is an approximation error and a solution error.

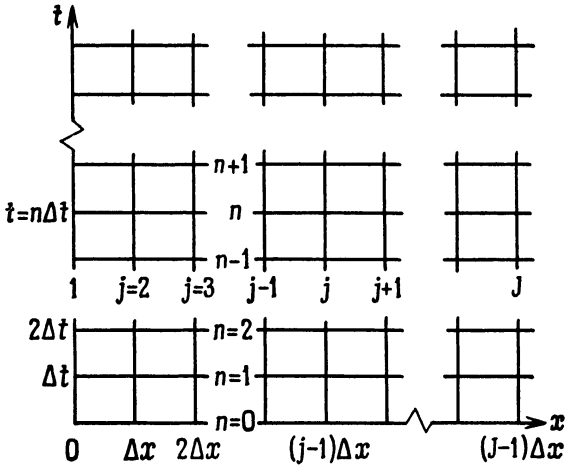

Figure 3.2 – Discrete grid

The true value of the approximate solution at an intermediate point between grid nodes is not at all obvious. According to the intuitive view, the solution should change smoothly in the intervals between the nodal points. In principle, the solution at some point (x_r, t_r) that does not coincide with a node can be constructed by interpolating the values corresponding to the solution for the surrounding nodal points.

If you look at Fig. 3.2, you can get a formula for determining the unknown value T_j^{n+1} depending on the known values T_j^n at the n-th time layer, that is, get the expression

$$T_j^{n+1} = T_j^n + \frac{a\Delta t}{\Delta x^2}\left(T_{j-1}^n - 2T_j^n + T_j^n + T_{j+1}^n\right) \tag{3.5}$$

To construct a complete numerical solution on the time layer $(n+1)$, one should apply the formula (3.5) to all nodes $j = 2..., j-1$, assuming that the Dirichlet boundary conditions provide data on the values of T_1^{n+1} and T_j^{n+1}.

3.3.2 Derivatives in space

We have already seen how the finite-difference method discretizes derivatives in space, for example as the quantity $\partial^2 \overline{T} / \partial x^2$ in the equation (3.1) is transformed into an expression $\left(T_{j-1}^n - 2T_j^n + T_{j+1}^n\right) / \Delta x^2$ that enters the equation (3.4).

The finite element method allows to achieve discretization due to the initial assumption that the local solution T allows interpolation. Then this local solution is substituted into the integral of all terms of the original equation, taken with appropriate weight. A typical result of such an action when using linear elements on a uniform grid has the following form

$$\begin{aligned} &\frac{(\Delta x/6)\left(T_{j-1}^{n+1} - T_{j-1}^n\right)}{\Delta t} + \frac{(2\Delta x/3)\left(T_j^{n+1} - T_j^n\right)}{\Delta t} + \frac{(\Delta x/6)\left(T_{j+1}^{n+1} - T_{j+1}^n\right)}{\Delta t} = \\ &= \frac{a\left(T_{j-1}^n - 2T_j^n + T_{j+1}^n\right)}{\Delta x} \end{aligned} \tag{3.6}$$

If we divide both sides of the equation (3.6) by Δx, the result will be similar in structure to the expression (3.4).

The spectral method uses assumptions similar to those of the finite element method, except that the expected form of the solution T is

$$T = \sum_{j=1}^{J} a_j(t)\varphi_j(x) \tag{3.7}$$

where $a_j(t)$ are the unknown coefficients determined in the process of constructing the solution, and $\varphi_j(x)$ are the known functions x.

The final form of the discrete representation of the equation using the spectral method can be written as

$$\frac{a_j^{n+1} - a_j^n}{\Delta t} = \sum_{j=1}^{J} p_j a_j^n \tag{3.8}$$

where p_j are the known algebraic coefficients.

Whatever method is used to perform the discretization, the process of further solving the equations, for example using the expression (3.8), applies directly to the algebraic equations and does not depend on the method of discretization.

3.3.3 Derivatives in time

When replacing $\partial \overline{T} / \partial t$ in equation (3.1) for the one-way difference expression $\left(T_j^{n+1} - T_j^n\right) / \Delta t$ uses information only from the time layers n and $n+1$. Because time changes only in the positive direction, information from the time layers with numbers $n+2$ and is no longer available to us. In the equation, (3.4) the spatial derivative $\partial^2 \overline{T} / \partial x^2$ was approximated on the time layer n, resulting in an explicit algorithm for determining T_j^{n+1}. If the spatial terms were approximated on the time layer $n+1$, the following implicit algorithm would result

$$-sT_{j-1}^{n+1} + (1+2s)T_j^{n+1} - sT_{j+1}^{n+1} = T_j^n \tag{3.9}$$

where $s = a\Delta t / \Delta x^2$.

The equation (3.9) can be solved if we consider it as part of a system of equations (3.8) obtained by writing this equation for all nodes $j = 2..., j-1$. If (3.1) we substitute the central difference formula into the equation $\left(T_j^{n+1} - T_j^{n-1}\right) / 2\Delta t$, then we can construct the following explicit algorithm for determining T_j^{n+1}

$$T_j^{n+1} = T_j^{n-1} + \left(2a\Delta t / \Delta x^2\right)\left(T_{j-1}^n - 2T_j^n + T_{j+1}^n\right) \tag{3.10}$$

The algorithm (3.10) is more accurate than the formula (3.8), but also more complex, since it covers data from two layers, and from three layers: $n-1$, n, $n+1$. This particular version of the algorithm turns out to be impractical, because it is unstable. However, when applied to other equations, such as the convection equation, the use of central differences over time leads to stable algorithms.

There is an alternative approach to discretizing time derivatives, built in connection with ordinary differential equations. The equation (3.1) can be written in the form

$$\frac{\partial \overline{T}}{\partial t} = L\overline{T} \tag{3.11}$$

where L is a differential operator $a\partial^2 / \partial x^2$. After discretization over space, the equation (3.11) takes the form

$$\frac{\partial T_j}{\partial t} = L_a T_j \tag{3.12}$$

where L_a is an algebraic operator obtained as a result of spatial discretization. The set of equations (3.12) written in each of the nodes is a system of ordinary differential equations in time. It follows that (3.12) any method of integrating ordinary differential equations can be applied to the equations. In general, the result of integration can be written in the form

$$T_j^{n+1} = T_j^n + \int_{t_n}^{t_{n+1}} L_a T_j dt' . \tag{3.13}$$

Calculating the integral according to (3.13) Euler's scheme gives an expression identical to the expression (3.5), if L_a is the finite-difference operator appearing in the equation (3.4)

$$T_j^{n+1} = T_j^n + \left[L_a T_j \right]^n \Delta t . \tag{3.14}$$

Due to errors associated with the introduction of the spatial discretization operator L_a, using a very high-order integration formula for the substitution (3.13) usually does not provide any advantages.

3.3.4 Taylor series expansion

As a first step towards developing an algorithm for calculating the values $\overline{T}$ that may appear in the equation (3.1), we will express the derivatives in space and time at the node (j,n) through value $\overline{T}$ in adjacent nodes. To implement this process, we will use Taylor series expansions of the type

$$\overline{T}_{j+1}^n = \sum_{m=0}^{\infty} \frac{\Delta x^m}{m!} \left[\frac{\partial^m \overline{T}}{\partial x^m} \right]_j^n , \tag{3.15}$$

$$\overline{T}_j^{n+1} = \sum_{m=0}^{\infty} \frac{\Delta t^m}{m!} \left[\frac{\partial^m \overline{T}}{\partial t^m} \right]_j^n \tag{3.16}$$

These series can be truncated after any number of terms, and the resulting error (approximation error) is determined mainly by the next term of the expansion, if only $\Delta x \le 1$ in the expansion (3.15) or if $\Delta t \le 1$ in the expansion (3.16). Therefore, we can write

$$\overline{T}_{j+1}^{n} = \overline{T}_{j}^{n} + \Delta x\left[\frac{\partial \overline{T}}{\partial x}\right]_{j}^{n} + \frac{\Delta x^2}{2}\left[\frac{\partial^2 \overline{T}}{\partial x^2}\right]_{j}^{n} + O\left(\Delta x^3\right). \tag{3.17}$$

The interpretation of the residual term $O\left(\Delta x^3\right)$ is that it is assumed that there is some positive constant K, which depends on $\overline{T}$, such that the difference between the value $\overline{T}$ in the node$\left(j+1,n\right)$ and the first three terms of the right-hand side (3.17), calculated at the node $\left(j,n\right)$ turns out to be numerically smaller than the value $K\Delta x^3$ for any sufficiently small Δx. It is clear that the error associated with such an approximation will rapidly decrease in magnitude as Δx.

Referring to expression (3.17), it is not difficult to see that the finite-difference representation $\partial \overline{T} / \partial x$ can be obtained directly. Indeed, rearranging the terms in (3.17) results

$$\left[\frac{\partial \overline{T}}{\partial x}\right]_{j}^{n} = \left(T_{j+1}^{n} - T_{j}^{n}\right) / \Delta x - 0.5\Delta x\left[\frac{\partial^2 \overline{T}}{\partial x^2}\right]_{j}^{n} + ... \tag{3.18}$$

It is obvious that the use of finite-difference substitution ensures accuracy $O\left(\Delta x\right)$.

$$\left[\frac{\partial \overline{T}}{\partial x}\right]_{j}^{n} \approx \frac{T_{j+1}^{n} - T_{j}^{n}}{\Delta x} \tag{3.19}$$

The additional terms appearing on the right-hand side of (3.18) are referred to as the approximation error. The expression on the right-hand side of the formula (3.19) is called the forward difference approximation. If we expand the quantity T_{j-1}^{n} in a Taylor series at the node j,n and regroup the terms, then we can construct a back-difference approximation

$$\left[\frac{\partial \overline{T}}{\partial x}\right]_{j}^{n} \approx \frac{T_{j}^{n} - T_{j-1}^{n}}{\Delta x}. \tag{3.20}$$

Like (3.19), this approximation introduces an error $O\left(\Delta x\right)$. Formulas (3.19) and (3.20) were obtained using the Taylor series expansion in space. Taylor series expansion in time is the formula (3.16) which can be used to construct the forward difference approximation

$$\left[\frac{\partial \overline{T}}{\partial t}\right]_{j}^{n} \approx \frac{T_{j}^{n+1} - T_{j}^{n}}{\Delta t}, \tag{3.21}$$

which introduces an error $O\left(\Delta t\right)$ if we assume that $\Delta t << 1$ and that higher-order derivatives are bounded.

3.4 Accuracy of the discretization process

Discretization is necessary to transform the original differential equations into an equivalent system of algebraic equations, for the solution of which a computer can be used. If we exclude the case when the corresponding exact solution has a simple analytical form, the discretization process necessarily introduces some error. Thus, one-sided formulas (3.19) are (3.20) accurate only for linear polynomials. The accuracy can be inferred from the fact that the approximation error terms are equal to zero for polynomials of sufficiently low order.

In the general case, the error of the finite-difference representation of the derivative can be determined by means of a Taylor series expansion in the vicinity of the node where the derivative is evaluated, while the evaluation of the leading term of the residue gives a fairly good approximation of the error, provided that the grid element size is small. However, the final evaluation of the Taylor series terms is only possible when the exact solution is known.

A more direct way to compare the accuracy of different algebraic formulas is to consider a simple analytic function of type $\overline{T} = \exp x$ and compare the value of the derivative obtained analytically with the value found by the discretization formula. Table 3.1 shows a similar comparison for the values of $\partial\overline{T} / \partial x$, evaluated at the point $x = 1$ for an analytic function of the form $\overline{T} = \exp x$, the step size is $\Delta x = 0{,}1$. In the general case, three-point formulas, whether symmetric or asymmetric, turn out to be much more accurate than two-point formulas with forward or backward difference, but much less accurate than a five-point symmetric formula. As is evident from Table 3.1, if the value Δx is quite small, the leading term of the Taylor series gives a good estimate of the error.

Table 3.1 Comparison of formulas for calculation $d\overline{T} / dx$ at $x = 1{,}0$

Formula variant	Algebraic formula	$\left[\frac{d\overline{T}}{dx}\right]_j$	Error	Leading term of Taylor series
Accurate	-	2.7183	-	-
Three-point symmetrical	$\left(\overline{T}_{j+1} - \overline{T}_{j-1}\right) / 2\Delta x$	2.7183	$0{,}4533\times10^{-2}$	$0{,}4531\times10^{-2}$
Differences ahead	$\left(\overline{T}_{j+1} - \overline{T}_{j}\right) / \Delta x$	2.7183	$0{,}1406\times10^{-0}$	$0{,}1359\times10^{-0}$
Differences back	$\left(\overline{T}_{j} - \overline{T}_{j-1}\right) / \Delta x$	2.7183	$-0{,}1359\times10^{-0}$	$-0{,}1359\times10^{-0}$
Three-point asymmetric	$\left(-1{,}5\overline{T}_{j} + 2\overline{T}_{j+1} - 0{,}5\overline{T}_{j+2}\right) / \Delta x$	2.7183	$-0{,}9773\times10^{-2}$	$-0{,}9061\times10^{-2}$
Five-point symmetrical	$\left(\overline{T}_{j-2} - 8\overline{T}_{j-1} + 8\overline{T}_{j+1} - \overline{T}_{j+2}\right) / 12\Delta x$	2.7183	$-0{,}9072\times10^{-5}$	$-0{,}9061\times10^{-5}$

Typical algebra formulas for estimating the function values $d^2\overline{T}/dx^2$ at x = 1.0 $\overline{T} = \exp x$ are given in Table 3.2. The function values are estimated with an interval $\Delta x = 0{,}1$. As before, the three-point symmetric formula is exact, but now the three-point asymmetric formula turns out to be inaccurate. As in the evaluation of the quality of the formulas for the first derivative, the leading term of the Taylor series provides a fairly accurate estimate of the error.

Table 3.2 Comparison of formulas for calculation $d^2\overline{T}/dx^2$ at $x = 1{,}0$

Formula variant	Algebraic formula	$\left[\frac{d^2\overline{T}}{dx^2}\right]$	Errors	Leading term of Taylor series
Accurate	-	2,7183	-	-
Three-point symmetrical	$\left(\overline{T}_{j-1} - 2\overline{T}_j + \overline{T}_{j+1}\right)/\Delta x^2$	2,7205	$0{,}2266\times10^{-2}$	$0{,}2265\times10^{-2}$
Three-point asymmetric	$\left(\overline{T}_j - 2\overline{T}_{j+1} + \overline{T}_{j+2}\right)/\Delta x^2$	3,0067	$0{,}2884\times10^{-0}$	$0{,}2718\times10^{-0}$
Five-point symmetrical	$\left(\overline{T}_{j-2} + 16\overline{T}_{j-1} - 30\overline{T}_j + 16\overline{T}_{j+1} - -\overline{T}_{j+2}\right)/12\Delta x^2$	2,7183	$0{,}3023\times10^{-5}$	$0{,}3020\times10^{-5}$

Table 3.3 Leading term of approximation error (algebraic representation) $d\overline{T}/dx$

Formula variant	Algebraic formula	Leading term of approximation error
Three-point symmetrical	$\left(\overline{T}_{j+1} - \overline{T}_{j-1}\right)/2\Delta x$	$\Delta x^2 T_{xxx}/6$
Differences ahead	$\left(\overline{T}_{j+1} - \overline{T}_j\right)/\Delta x$	$\Delta x T_{xx}/12$
Differences back	$\left(\overline{T}_j - \overline{T}_{j-1}\right)/\Delta x$	$-\Delta x T_{xx}/2$
Three-point asymmetric	$\left(-1{,}5\overline{T}_j + 2\overline{T}_{j+1} - 0{,}5\overline{T}_{j+2}\right)/\Delta x$	$-\Delta x^2 T_{xxx}/3$
Five-point symmetrical	$\left(\overline{T}_{j-2} - 8\overline{T}_{j-1} + 8\overline{T}_{j+1} - -\overline{T}_{j+2}\right)/12\Delta x^2$	$-\Delta x^4 T_{xxxxx}/30$

Algebraic formulas for the main terms of the expressions that determine the approximation error are given in Tables 3.3 and 3.4. These formulas are obtained by expanding in a Taylor series in the vicinity of the j-th node. In Table 3.3, the notation is adopted $\overline{T}_{xxx} \equiv d^3\overline{T}/dx^3$ and etc.

For this specific example, $\left(\overline{T} = \exp x\right)$ we have $\overline{T}_{xxx} = \overline{T}_{xxxx}$ etc. Therefore, the magnitude of the error primarily depends on the degrees Δx. And from this it follows that as it decreases, Δx it should be expected that when using the five-point formula, the approximation error will decrease much faster than when using two-point formulas with forward or backward difference.

Table 3.4 Leading term of approximation error (algebraic representation) $d^2\overline{T}/dx^2$

Formula variant	Algebraic formula	Leading term of approximation error
Three-point symmetrical	$\left(\overline{T}_{j-1}-2\overline{T}_j+\overline{T}_{j+1}\right)/\Delta x^2$	$\Delta x^2 T_{xxxx}/12$
Three-point asymmetric	$\left(\overline{T}_j-2\overline{T}_{j+1}+\overline{T}_{j+2}\right)/\Delta x^2$	$\Delta x T_{xxx}$
Five-point symmetrical	$\left(\overline{T}_{j-2}+16\overline{T}_{j-1}-30\overline{T}_j+16\overline{T}_{j+1}-\right.$ $\left.-\overline{T}_{j+2}\right)/12\Delta x^2$	$-\Delta x^4 T_{xxxxxx}/90$

The reason for the large error associated with the three-point asymmetric formula and shown in Table 3.2 becomes apparent from Table 3.4, which shows that the main term of the approximation error has in this case only the first order of accuracy. As is evident from Tables 3.1 and 3.2, there is a close correlation between the directly calculated error and the approximation error. On this basis, it should be expected that the directly calculated error will decrease along with Δx the same law as shown in Tables 3.3 and 3.4. This conclusion is confirmed by the results shown in Fig. 3.4 and 3.5. If we plot the graphs according to the logarithmic scale, the gradient of the results concerning the convergence, i.e. the convergence rate, corresponds to a certain degree Δx. As is clear from the data presented in the graphs, in the various cases we have considered, the expected rate of convergence is suggested by the expressions of the main terms of the approximation error, which are given in Tables 3.3 and 3.4. The estimate of the rate of convergence can, as before, be based on the data on the approximation error, even if the exact solution is unknown.

3.5 Consistency of differential schemes

The system of algebraic equations obtained as a result of the discretization process ***is consistent*** with the initial partial differential equations (PDEs); when the grid step sizes approach zero, the system of algebraic equations is equivalent to the PDEs at each of the grid nodes.

It is clear that the presence ***of consistency*** is necessary if the approximate solution is to converge to the solution of the PDEs under consideration. However, this condition is not sufficient, since the system of algebraic equations will be equivalent to the PDEs when the grid step size approaches zero, hence it cannot be concluded that the solution of this system of algebraic equations will approach the solution of the original PDEs.

Consistent difference scheme is one that approximates a partial differential equation. Recall that the approximation error is the difference between a differential equation and its finite-difference analogue, so the condition for consistency of a difference scheme is that the approximation error tends to zero as the grid is refined. This condition is certainly satisfied if the approximation error decreases as the grid is refined, i.e. if the approximation error has order $O(\Delta t),\ O(\Delta x)$, etc.

However, if the order of approximation error is, for example, $O(\Delta t/\Delta x)$, then the scheme will be consistent only if the mesh refinement is performed in accordance with the condition $\Delta t/\Delta x \to 0$.

3.5.1 Rounding error

Any numerically obtained solution, even a so-called exact analytical solution of a partial differential equation, depends on rounding errors associated with the finite number of signs used in arithmetic operations.

The error that occurs in this case is called rounding error. It can significantly affect the solution of finite-difference equations, since obtaining this solution is usually associated with performing a large number of the same type of arithmetic operations. In some cases, the rounding error is proportional to the number of nodes of a different grid, so refining the grid, while reducing the approximation error, can increase the rounding error.

The approximation error is equal to the difference between the exact (excluding rounding errors) solutions of the original differential equation and its finite-difference analogue. Therefore, the error of the solution of the partial differential equation obtained on the computer is equal to the sum of the approximation and rounding errors. The accuracy of the numerical solution of the partial differential equation is determined by the approximation error of both the equation itself and the boundary conditions.

3.5.2 Stability of finite-difference schemes

If, when solving a system of algebraic equations, some disturbances arise that are not subject to control (rounding errors), and such disturbances tend to decay, then the solution will be ***stable***. Conversely, an unstable solution will give an amplitude of oscillations that increases with time. Therefore, the concept of stability is associated with the growth or decay of errors that are introduced into the calculation at any stage. These are precisely those errors that are associated with rounding directly in the process of calculation by a computer. A specific method and solution are considered stable if the cumulative effect of all rounding errors is negligibly small.

Similarly, the stability of a finite-difference scheme is determined by the change in error during the calculation. O'Brien et al. [O'Brien et al., 1950] proposed the following classification of the stability of difference schemes:

1. If the total rounding error increases (does not increase), then the difference scheme is called highly unstable (stable).

2. If a separate rounding error increases (does not increase), then the difference scheme is called weakly unstable (stable).

Usually, only weak stability is studied, since for its analysis it is possible to use the method of Fourier series expansion of the solution, which in terms of computational mathematics is called the Neumann method. It is assumed that if the condition of weak stability is met, then the condition of strong stability is met.

3.5.3 Convergence

The solution of algebraic equations that approximate a given PDEs is called ***convergent*** if this approximate solution of the PDEs approaches the exact solution of the PDEs for any value of the independent variable as the grid step size approaches zero.

Thus, we require that $T_j^n \to \bar{T}\left(x_{i,}\, t_n\right)$ when $\Delta x \to 0, \Delta t \to 0$.

The difference between the exact solution of a partial differential equation and the exact solution of a system of algebraic equations is called the solution error and is denoted by the symbol e_j^n

$$e_j^n = \bar{T}\left(x_j, t_n\right) - T_j^n . \tag{3.22}$$

The exact solution of a system of algebraic equations is an approximate solution of the original partial differential equation. The solution of a system of algebraic equations is exact if no numerical errors, such as rounding errors, were introduced into it during the calculation process.

As a rule, the magnitude of the error e_j^n at $(j,\ n)$-th node depends on the sizes of the grid cells Δx and Δt, as well as on the values of those higher derivatives at this node that were not taken into account when formulating finite-difference approximations of the derivatives in the considered differential equation.

3.5.4 Computational efficiency

The question of what accuracy can be achieved with a given solution algorithm is related to the computational efficiency of that particular algorithm. Computational efficiency can be defined as the accuracy achieved per unit of execution time. Thus, an algorithm that achieves moderate accuracy on a coarse grid and takes a short time to execute may be as efficient as another algorithm that achieves high accuracy on a fine grid and takes a longer time to execute.

Computational efficiency (CE) can be determined by the formula $CE = k / \left(\varepsilon ET\right)$, where ε is the error of the approximate solution, given in the corresponding norm (for example, in the root mean square), ET is the execution time.

Computational efficiency is a useful criterion that allows us to clarify the feasibility of introducing high-order finite-difference schemes or higher-order finite elements.

For most thermal physics and hydroaerodynamics problems, the use of high-order (e.g., fourth-order or higher) finite-difference schemes or the use of higher-order finite elements (e.g., cubic or higher) is not justified in terms of increased computational efficiency, especially for three-dimensional problems.

When developing different computational schemes, it is preferable to compare them in terms of computational efficiency, and not only in terms of accuracy or only in terms of economy. If computational schemes are compared that are calculated on the same computer, then the execution time can be measured directly. If it is not possible to set the CPU time, attention should be paid to the operating mode of the computer.

LIST OF RECOMMENDED REFERENCES

1. Blazek J. Computational Fluid Dynamics: Principles and Applications. Elsevier. 2001. 470 p.
2. Chung T. Computational Fluid Dynamics. Cambridge University Press. 2010. 1058 p.
3. Saad Y. Iterative Methods for Sparse Linear Systems. Philadelphia: PA, USA, Society for Industrial and Applied Mathematics. Philadelphia PA. 2003. 184 p.
4. Van der Vorst H. Iterative Methods for Large Linear Systems. Utrecht, Netherlands . 2002. 195 p.
5. Wesseling P. Elements of Computational Fluid Dynamics. TU Delft. 2001. 147 p.