

Chapter 2

Discretization and Formulation of Solution Approaches

Abstract Discretization principles as a foundation for numerical approaches and PDE solution methodologies are described, for space and time, from a bird's-eye view. Two PDE solution methodologies, based on the strong or the weak form, in mesh-based and meshless methods, are introduced briefly.

Keywords Strong form · Weak form · Discretization of time

2.1 Background

In the previous section, we showed how to formulate a simple physical model for heat transfer (see Eq. (1.4)). The formulated diffusion PDE depends on spatial and temporal variables. Now, we model a more general problem as:

$$\mathcal{L}u(\mathbf{x}) = g(\mathbf{x}), \quad \mathbf{x} \in \Omega, \quad (2.1)$$

with boundary conditions:

$$\mathcal{B}u(\mathbf{x}) = h(\mathbf{x}), \quad \mathbf{x} \in \Gamma, \quad (2.2)$$

where u is a continuous unknown solution, \mathbf{x} is a vector of continuous independent variables, \mathcal{L} and \mathcal{B} are differential operators, g and h are known functions, Ω is the problem domain, and Γ is its boundary.

We continue by describing the functions in terms with which we will express the numerical solutions of the PDEs:

- The *basis functions* p_j are the members of the *basis*, a set of functions that spans the space of the employed interpolating or approximating functions. The typical basis functions are monomials, Gaussian, splines, etc.
- The *shape functions* ϕ_i are linear combination of the basis functions that can reconstruct arbitrary field u through an interpolation or approximation. The function can also be interpreted as a nodal approximation function of value 1 in node

\mathbf{x}_i and 0 in all the other nodes from the global domain Ω . The shape functions are fully determined by the distribution of nodes and definition of basis functions. A smooth, hat-shaped weight function w_i can be applied to control the amount of nodals' impacts.

- The *nodal trial function* \hat{u}_i of a node \mathbf{x}_i is a parameterized function that approximates or interpolates the field u on the local support domain Ω_{Si} .
- The *trial function* \hat{u} of the unknown solution u is a parameterized function that approximates or interpolates the field u on the global domain Ω .

To summarize, from suitable basis functions and nodal positions, we can create shape functions that are used to construct the trial functions. The next chapter describes in more details how this is done. Please note, that we will differentiate between \hat{u} and u only where the difference must be stressed, otherwise u will be used for the approximate solution.

In most numerical methods for solving PDEs, the general strategy is to represent continuous unknown field u , e.g., temperature, pressure, stress, velocity, etc., with discrete values in a set of discretization points of independent variables, e.g., spatial coordinates, time, or others. The spatial discretization strategy relies on a distribution of N discretization points through the problem domain Ω for which a solution is sought. We will term the discretization points as nodes:

$$\mathbf{x}_i \in \Omega, \quad \text{for } i \in \{1, \dots, N\}, \quad (2.3)$$

in order to distinguish them from any other points in the domain, e.g., evaluation points for the calculation of numerical integrals, evaluation points for visualization, etc. The independent variables \mathbf{x} and the approximate solution \hat{u} are discretized using the set of nodes \mathbf{x}_i and the corresponding nodal parameters $\hat{u}(\mathbf{x}_i) = u_i$. The solution in the nodes can encompass more variables, e.g., temperature, velocities, pressure, displacements, etc., in such cases the vector notation $\mathbf{u}(\mathbf{x}_i)$ will be used.

The conceptual difference between mesh-based and meshless methods is in the way the discretization nodes are treated. In mesh-based methods, they are organized in a mesh before the solution procedure, using a priori knowledge about the neighboring nodes of \mathbf{x}_i and the relations between them. In meshless methods, no a priori knowledge about the nodal topology is required. To determine the support domains, simple algorithms like Nearest Neighbors Search (NNS) can be used, either during the simulation (SPH) or in a preprocess phase (DAM, LRBFCM).

In mesh-based methods, the discretization nodes can be organized in a mesh of polygons, traditionally called *elements*. The mesh is usually determined by a list of elements with the corresponding ordered nodes. Note that for regular meshes with an isomorphic neighborhood of nodes (for example, the FDM), the list can be generated explicitly, i.e., the neighboring nodes are determined by an explicit function. Unfortunately, in many real cases with nonregular geometries in Ω , such

an approach is not appropriate. More sophisticated methods (for example, the FEM) are based on the formation of polygons that can cover a generally shaped Ω and adaptively maintain the discretization densities.

Alternatively, the unknown variables in the solution can be approximated with a set of unconnected nodes. Instead of a mesh, we need an appropriate algorithm for the selection of the nodes that influence the approximation. The collection of selected nodes is known as the support domain $\Omega_S \subset \Omega$. The selection of the strategy might have a major impact on the solution quality and the execution performance of the method. It is intuitively clear that with a denser discretization, higher accuracy in the approximate solution can be achieved, i.e., the approximate solution converges to the accurate solution of the PDE as the distances between \mathbf{x}_i limit to zero. This is true for a linear case if the solution methodology is *consistent*—the local truncation error goes to zero with denser discretization nodes, and *stable*—the approximate solution remains bounded for the analyzed set of independent variables in the global problem domain Ω .

We can distinguish between the two basic formulations of the PDE solution approaches:

- The strong formulation relies on the use of approximated derivatives in the PDE and the determination of one equation for each node \mathbf{x}_i that satisfies the PDE.
- The weak formulation, on the other hand, tries to reach an optimal criterion for the accuracy of the approximated solution over the entire domain Ω , not in the nodes only.

Generally, a numerical solution of a PDE is based on the spatial discretization of the simulated global domain Ω , including its boundary Γ , which converts the PDE (2.1) with boundary conditions (2.2) into a system of algebraic equations:

$$\mathbf{K}\mathbf{u} = \mathbf{f}, \quad (2.4)$$

with \mathbf{u} the vector of the unknown solutions in nodes, \mathbf{K} the global system matrix, where elements of row i in columns j represent the relation of the node \mathbf{x}_i to the nodes \mathbf{x}_j , and \mathbf{f} the vector of the discretized right-hand side (rhs) of the PDE (2.1) and its boundary conditions (2.2). The matrix \mathbf{K} and the vector \mathbf{f} are traditionally named the stiffness matrix and the load vector, respectively. The names originate from the field of mechanics [64].

In the local numerical methods, the solution values in the nodes depend only on the nearest neighboring nodes, which can be defined either by a predefined mesh of elements (the mesh-based approach) or by searching for a relatively small number of nearest nodes that belong to the support domain. In both cases, the equations of the final system can be obtained either by an approximation of the nodal derivatives from \mathcal{L} and \mathcal{B} and the satisfaction of the PDE for all the nodes (strong form) or by a global minimization of the differences between the approximate and the exact solutions over the entire domain Ω (weak form). Here, the Boundary Conditions (BC) are expressed by a general differential operator \mathcal{B} , however, commonly they

are prescribed as Dirichlet or essential BC by Eq. (2.5) or as Neumann or natural BC by Eq. (2.6) on the corresponding boundaries $\Gamma = \Gamma_e \cup \Gamma_n$:

$$u(\mathbf{x}) = \bar{u}(\mathbf{x}), \quad (\mathbf{x}) \in \Gamma_e, \quad (2.5)$$

$$\frac{\partial u(\mathbf{x})}{\partial n} = \bar{g}(\mathbf{x}), \quad (\mathbf{x}) \in \Gamma_n, \quad (2.6)$$

where $\bar{u}(\mathbf{x})$ and $\bar{g}(\mathbf{x})$ are the prescribed boundary functions and n is the normal to the boundary. A weighted combination of both the above boundary conditions is referred to as Robin BC

$$au(\mathbf{x}) + b \frac{\partial u(\mathbf{x})}{\partial n} = \bar{r}(\mathbf{x}), \quad (\mathbf{x}) \in \Gamma_r, \quad (2.7)$$

where a and b are scalars or, in more general cases, functions defined on Γ_r . Term $\bar{r}(\mathbf{x})$ is a prescribed boundary function.

If a PDE is time dependent, a separate initial condition must be prescribed explicitly by:

$$u(\mathbf{x}, t) = u_0, \quad t = t_0, \quad (2.8)$$

where u_0 is the known initial condition at t_0 . The goal is to calculate the numerical solution $u(\mathbf{x}, t)$ for any time $t > t_0$. A common solution approach is to first apply the spatial discretization, to formulate an intermediate system of Ordinary Differential Equations (ODEs), which the subsequent temporal discretization transforms into a system of algebraic equations. Temporal discretization can be implemented by explicit or implicit numerical methods. The explicit method computes the next time step from the current state. Therefore, only a matrix vector multiplication is necessary for the evaluation of the approximate solution in each time step. On the other hand, the implicit methods seek the next time step solution by satisfying the PDE at hand. Consequently, a global system must be solved with a different rhs in each time step.

The PDE (2.1) and corresponding BCs can be written in an alternative way as a function of independent variables and their derivatives:

$$F(\mathbf{x}, u(\mathbf{x}), u_{,x_1}(\mathbf{x}), u_{,x_2}(\mathbf{x}), \dots, u_{,x_1x_2}(\mathbf{x}), \dots) = 0, \quad (2.9)$$

where the partial derivatives, for example a derivative on x_1 , and a mixed derivative on x_1 and x_2 , are expressed by the part of subscripts following the comma:

$$u_{,x_1}(\mathbf{x}) = \frac{\partial u(\mathbf{x})}{\partial x_1}, \quad u_{,x_1x_2}(\mathbf{x}) = \frac{\partial^2 u(\mathbf{x})}{\partial x_1 \partial x_2}. \quad (2.10)$$

More details, explanations, and derivations of the above-mentioned approaches are given in the following sections.

2.2 Strong Form

An intuitive approach for the solution of the PDE (2.1) is based on its original, strong form. The methodology is, in general, simple and straightforward. First, the domain Ω is discretized with nodes \mathbf{x}_i . Then, a numerical approximation of the derivatives appearing in the PDE (2.9) is expressed as a function of \mathbf{x} . For each node, including the boundary nodes, the unknown solution and its derivatives in the PDE are replaced by their numerical approximations:

$$r(\mathbf{x}) = F(\mathbf{x}, \hat{u}(\mathbf{x}), \hat{u}_{,x_1}(\mathbf{x}), \hat{u}_{,x_2}(\mathbf{x}), \dots, \hat{u}_{,x_1x_2}(\mathbf{x}), \dots). \quad (2.11)$$

The obtained function $r(\mathbf{x})$ is the residual of the PDE (2.9), since it reflects the inaccuracy or the error in the solution. Forcing the residual to be zero in each node, we obtain a system of the kind $\mathbf{Ku} = \mathbf{f}$ (2.4). The resulting global system (2.4) is linear when F is linear. Moreover, when only a few neighboring nodes influence the solution at each node, the system becomes sparse, and the method is considered a local strong form method. There are many approaches for the approximation of the derivatives, based on the Taylor series or on an analytical derivation of the interpolated or approximated solutions with nodal trial functions. Note that differentiation is, by its nature, an ill-conditioned problem and therefore the accuracy, in particular of higher order derivatives, might be quite poor. The order of the accuracy can be increased, besides by increasing the number of discretization nodes and also by incorporating more neighboring nodes, which results in more complex processing and more nonzero elements in the final system.

A mesh-based representative of the strong form approach is the well-known FDM, where the Taylor expansion is used to approximate the derivatives in the nodes. The meshless variants of the strong form approaches are, for example, the DAM and the LRBFCM, which could also be understood as a generalization of the FDM. It has been proved long ago in the Lax Richtmyer equivalence theorem [65] that the FDM-based numerical solutions of well-posed PDEs converge to the true solutions. An important characteristic of the strong form approach is that the solution values between the nodes are not known; also the derivatives exist only in the discretization nodes.

2.3 Weak Form

An alternative technique for the numerical solution of the PDE (2.1) is based on its integral, weak form, which requires weaker consistency of the unknown solution. The weak form can be obtained by variation methods [18] or by the weighted residual method [52]. We will use the former, since it is often used in the meshless context.

Replacing the unknown solution u with its approximation \hat{u} in Eqs. (2.1) and (2.2) enables an alternative formulation of the residuals:

$$r(\mathbf{x}) = \mathcal{L}\hat{u}(\mathbf{x}) - g(\mathbf{x}) \quad \text{and} \quad \bar{r}(\mathbf{x}) = \mathcal{B}\hat{u}(\mathbf{x}) - h(\mathbf{x}). \quad (2.12)$$

The weighted residual method forces the residuals (2.12) to be orthogonal to each of the given sets of test functions Ψ . Such an approach leads to a weak form of Eq. (2.1), expressed as:

$$\int_{\Omega} r(\mathbf{x})\Psi(\mathbf{x}) d\Omega + \int_{\Gamma} \bar{r}(\mathbf{x})\bar{\Psi}(\mathbf{x}) d\Gamma = 0, \quad (2.13)$$

Test functions Ψ and $\bar{\Psi}$ can, in principle, be any nonzero functions that lead to a convenient formulation, although the choice affects the implementation methodology and its efficiency, and to some extent also the accuracy of the solution [66]. The test functions must be sufficiently smooth to enable the calculation of the integrals in (2.13).

The principle of orthogonality is schematically illustrated for a 2D case in Fig. 2.1. The true solution \mathbf{u} of a PDE does not usually lie in the subspace spanned by the basis or shape functions ϕ , because it cannot be exactly reconstructed by them. The approximate solution $\hat{\mathbf{u}}$, which is approximated by ϕ , will be optimal in some sense if the residual $\mathbf{r} = \mathbf{u} - \hat{\mathbf{u}}$ is orthogonal to the spanned subspace $\text{span}(\phi)$ (see the bold, dotted vector \mathbf{r} in Fig. 2.1). The principle of orthogonality is formally applied in Eq. (2.13).

By inserting (2.12) into (2.13), first term of Eq. (2.13) for Ω becomes:

$$\int_{\Omega} \mathcal{L}\hat{u}(\mathbf{x})\Psi(\mathbf{x}) d\Omega - \int_{\Omega} g(\mathbf{x})\Psi(\mathbf{x}) d\Omega = 0 \quad (2.14)$$

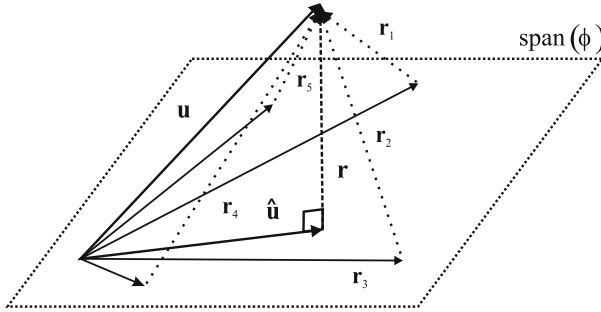


Fig. 2.1 The approximate solution is optimal when the residual $r(\mathbf{x})$ is the “shortest,” which happens when it is orthogonal to the plane $\text{span}(\phi)$

For the sake of simplicity, let us consider for a moment only the first term of Eq. (2.14). After applying the divergence rule we obtain:

$$\int_{\Omega} \mathcal{L}^{(r)} \hat{u}(\mathbf{x}) \Psi(\mathbf{x}) d\Omega = \int_{\Gamma} \mathcal{L}^{(r-1)} \hat{u}(\mathbf{x}) \mathbf{n}_{,x} \Psi(\mathbf{x}) d\Gamma - \int_{\Omega} \mathcal{L}^{(r-1)} \hat{u}(\mathbf{x}) \Psi_{,x}(\mathbf{x}) d\Omega, \quad (2.15)$$

where the maximal degree of derivatives in \hat{u} is denoted by a superscript of the differential operator and $\mathbf{n}_{,x}$ is the derivative of the boundary normal. Note that the required order of the derivatives of \hat{u} is reduced by one, but the required order of the derivatives of Ψ is increased by one. In an analogous way, the same is valid for consecutive applications of the divergence rule. Such a transfer of the differentiation from the unknown solution to the test functions enables a reduction of the required highest derivatives in the weak form, which can be considered as an advantage of the weak form methodology, in spite of its complex formulation.

Equation (2.14) must be valid for all internal nodes, therefore it leads to a system of linear equations:

$$\int_{\Gamma} \mathcal{L}^{(r-1)} \hat{u}(\mathbf{x}) \mathbf{n}_{,x} \Psi_i(\mathbf{x}) d\Gamma - \int_{\Omega} \mathcal{L}^{(r-1)} \hat{u}(\mathbf{x}) \Psi_{i,x}(\mathbf{x}) d\Omega - \int_{\Omega} g(\mathbf{x}) \Psi_i(\mathbf{x}) d\Omega = 0, \quad (2.16)$$

with Ψ_i being test functions of discretization nodes. Equation (2.16) is a foundation for further procedures of different weak form methods. The procedure for boundary nodes is analogous but often much simpler because of simpler differential operator.

The selection of the test functions Ψ_i provide different methods. For example, if the test functions are the residual r itself, we obtain $\int_{\Omega} r^2(\mathbf{x}) d\Omega = 0$, which is the least square method. If the test functions are the *Dirac delta* function δ , we get the collocation method, because $\delta(\mathbf{x} - \mathbf{x}_i) = 0$ in all $\mathbf{x} \neq \mathbf{x}_i$ from Ω . Here, $\int_{\Omega} r(\mathbf{x}) \delta d\Omega = 0$ is fulfilled through $r(\mathbf{x}_i) = 0$ in nodes, which is the foundation of the collocation methods. Finally, the well-known FEM is obtained if the interpolating shape functions ϕ_i are also taken for the test functions.

Forcing the residual to be orthogonal on the entire domain, by applying the rule for each node over its subdomain or element, we obtain a global equation system of the kind $\mathbf{Ku} = \mathbf{f}$ (2.4). The nonzero elements of \mathbf{K} and \mathbf{f} are determined by a numerical evaluation of the integrals in Eq. (2.14), which usually represents the main part of the computational complexity in weak form methods. For each discretization node, all the contributions from the neighboring nodes are assembled in the global system matrix, which is sparse again, but can be either symmetric or not, depending on the selected test functions. The unknown parameters \mathbf{u}_i of the approximate solution are obtained by a system solver that can be tailored to specific types of system matrices. Now, the approximate solution $\hat{\mathbf{u}}$ can be reconstructed at any point from Ω .

A mesh-based representative of the weak form approach is the well-known FEM, based on the element wise interpolation with monomial-based shape functions.

The meshless variant of the weak form approach is MLPG1, which uses weight functions of MLS approximation as the test functions. Typical representatives of both strong and weak methods will be analyzed in more detail in subsequent chapters.

2.4 Discretization of Time

In many practical cases, physical phenomena are evolving in time and therefore the modeling PDEs are time dependent, e.g. in a simulation of the cooling processes after surgery [67, 68], in the analysis of solidification [69], in vibration analysis [70, 71], etc. In all the enumerated cases, the evolution of the solution in time plays a crucial role and thus the time becomes an important independent variable. The common solution methodologies for time-dependent problems are based on time stepping and the integration of the PDE over time. There are several approaches to tackle temporal stepping, implemented either in explicit or implicit schemes. In the explicit methods, the solution in the next time step is calculated as the prediction from past time steps and, therefore, only a matrix vector multiplication is necessary. On the other hand, in the implicit methods, the solution in the next time step is defined in terms of the past steps as well as next time step, a yet unknown value; therefore, the solution of a linear system of equations is required. The explicit methods are simpler; however, they are unstable with larger time steps, while the implicit methods require a more sophisticated solution approach and are more stable.

For example, the Verlet [72] or very similar leapfrog methods and other symplectic [73] time integration methods are commonly used in particle methods like SPH and molecular dynamics. Most of these methods achieve the second order of accuracy in time and can be implemented in the explicit or implicit forms. The multistep methods can provide even higher accuracy on account of the need to store data from several previous time steps. This excessive data storage can be avoided by using the time instances between two consecutive time steps, like in the Runge Kutta methods [74]. There are also higher order multivalued methods that are based on polynomial interpolation [75]. They have more freedom in the selection of time steps and can easily achieve a higher accuracy and stability. Although the sophisticated time discretization gains benefits in terms of the accuracy and stability, it loses much with respect to computational time and in the implementation effort.

Suppose that the PDE from Eq. (2.1) depends, besides on space variables, also on time. If we implement just a spatial discretization, as described in the previous sections, and leave the time variable untouched, a global system of Ordinary Differential Equations (ODEs), with unknown nodal parameters that depend on time, can be constructed. Such a semidiscrete system can be solved by well-developed methodologies for the solution of ODEs. Alternatively, the times could be treated as the other independent variables, which leads to fully discrete methods. However, in most practical implementations, the time is discretized separately. One of the practical reasons for this is the fact that “space” is usually bounded, but “time” remains open into the future.

If ODEs comprise time derivatives up to the order of k , denoted with $u_{,t^k}$, they can be written in an explicit form as:

$$\mathbf{u}_{,t^k} = f(t, u, u_{,t}, u_{,t^2}, \dots, u_{,t^{k-1}}). \quad (2.17)$$

The system can be transformed into an equivalent system of ODEs of first order by introduction of k new unknowns $u_1 = u$, $u_2 = u_{,t}$, \dots , $u_k = u_{,t^{k-1}}$:

$$\mathbf{u}_{,t} = [u_{1,t}, u_{2,t}, \dots, u_{k-1,t}, u_{k,t}] = [u_2, u_3, \dots, u_k, f(t, u_1, u_2, \dots, u_k)] = \mathbf{g}(t, \mathbf{u}). \quad (2.18)$$

Now, the discretization of time is applied, which transforms the system of ODEs into a system of algebraic equations for each time step.

A general approach in the solution of a single time-dependent ODE $\mathbf{u}_{,t} = \mathbf{g}(t, \mathbf{u})$ is to start from the initial time t_0 with a given initial value \mathbf{u}_0 and to follow the solution trajectory determined by the ODE. The initial slope of the solution components is determined from the ODE itself, by inserting the initial time and the initial value: $\mathbf{u}_{,t}^0 = \mathbf{g}(t_0, \mathbf{u}^0)$. Using the initial slope, the solution value \mathbf{u}^1 at the next time step $t_1 = t_0 + \Delta t$ can be predicted and then the slope at t_1 is calculated from the ODE as: $\mathbf{u}_{,t}^1 = \mathbf{g}(t_1, \mathbf{u}^1)$, and so on for all further time steps.

The simplest method based on this strategy is the explicit Euler's method, which estimates the solution \mathbf{u}^{k+1} at time $t_{k+1} = t_k + \Delta t$ by:

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \Delta t \mathbf{g}(t_k, \mathbf{u}^k). \quad (2.19)$$

Euler's method can be derived from the Taylor series, by using just the first two terms. The predicted solution value depends only on a single previous solution value [75]. Unfortunately, explicit methods have a limited stability region that restricts the length of Δt , which consequently prolongs the computing time to obtain the solution.

The stability region can be increased by using a more sophisticated, implicit approach, that is used, for example, in the backward Euler's method:

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \Delta t \mathbf{g}(t_{k+1}, \mathbf{u}^{k+1}). \quad (2.20)$$

Now, the solution values in the next time step \mathbf{u}^{k+1} are obtained with the evaluation of \mathbf{g} in \mathbf{u}^{k+1} , which is still not known. If \mathbf{g} is nonlinear an iterative solution methods must be used. It can be shown [75] that the backward Euler's method is unconditionally stable. Both Euler methods are first-order accurate $O(\Delta t)$.

By averaging the explicit and implicit Euler methods:

$$\mathbf{u}^{k+1} = \mathbf{u}^k + \Delta t (\mathbf{g}(t_k, \mathbf{u}^k) + \mathbf{g}(t_{k+1}, \mathbf{u}^{k+1}))/2, \quad (2.21)$$

a trapezoid method is defined, which is implicit, unconditionally stable and second-order accurate $O(\Delta t^2)$. Alternatively, by defining the solution $\mathbf{u}^{k+\frac{1}{2}}$ and its first

derivative $\mathbf{u}_{,t}^{k+\frac{1}{2}}$ in the midpoint $t_{k+1} = t_k + \Delta t/2$, with the interpolation of the current and next solution values, the well-known Crank–Nicolson time discretization scheme is obtained [76]:

$$\mathbf{u}^{k+\frac{1}{2}} = \frac{\mathbf{u}^{k+1} + \mathbf{u}^k}{2}, \quad \mathbf{u}_{,t}^{k+\frac{1}{2}} = \frac{\mathbf{u}^{k+1} - \mathbf{u}^k}{\Delta t}. \quad (2.22)$$

Return now to our initial time-dependent PDE (2.1) and suppose that it has already been discretized in space. The derived expressions for \mathbf{u} and $\mathbf{u}_{,t}$ are used as discretized values of the unknown solution in the global ODE system:

$$\mathbf{C}\mathbf{u}_{,t}(t) + \mathbf{K}\mathbf{u}(t) - \mathbf{f} = 0. \quad (2.23)$$

Besides the stiffness matrix \mathbf{K} and the load vector \mathbf{f} , the matrix \mathbf{C} , traditionally termed the damping matrix, collects the terms related to the time derivatives.

For example, if the Crank–Nicolson method is used, the expressions from Eq. (2.22) are inserted into the ODE (2.23), which finally results in a global system of linear algebraic equations of the form:

$$\mathbf{A}\mathbf{u}^{k+1} = \mathbf{B}\mathbf{u}^k + \Delta t\mathbf{f}^{k+\frac{1}{2}}, \quad (2.24)$$

where, for internal nodes:

$$\mathbf{A} = 2\mathbf{C} + \Delta t\mathbf{K}, \quad \mathbf{B} = 2\mathbf{C} - \Delta t\mathbf{K}. \quad (2.25)$$

There are many numerical approaches for the solution of ODEs with unconditional stability and higher accuracy. In choosing the time step, we would like to minimize the calculation complexity with longer time steps and therefore fewer iteration steps towards the final solution.

From the user's point of view, a tolerable error estimation $maxerr$ is needed for the proper selection of the time step. In the case of Euler's methods, the local error in the time step k is approximately $(\Delta t^2/2)\mathbf{u}_{,t^2}^k$, which limits the time step size by:

$$\Delta t \leq \sqrt{2maxerr/\|\mathbf{u}_{,t^2}^k\|}, \quad (2.26)$$

where $\mathbf{u}_{,t^2}^k$ can be estimated from previous known solution values by:

$$\mathbf{u}_{,t^2}^k \approx \frac{\mathbf{u}_{,t}^k - \mathbf{u}_{,t}^{k-1}}{\Delta t}. \quad (2.27)$$

The final selection of the solution approach, i.e., explicit or implicit, is conditioned by the stability, accuracy, execution time of the solution program, and above all, by the ease of implementation. The *explicit methods* are simple to implement and

computationally effective; their asymptotic complexity of the calculation, in each time step, is $O(bN)$, for N equations with b nonzero elements in each equation. However, explicit stepping suffers from stability issues. It is clear that the calculation of the solution cannot progress with the time steps faster than the physical phenomena that are modeled by the PDE. For example, if we are modeling a convective wave, we have to take care that the local information does not reach out of the support domain within a time step. The condition is also known as the Courant Friedrichs Lewy condition (CFL). For example, the CFL for the 1D wave equation solved by the FDM can be written as

$$v_x \frac{\Delta t}{\Delta x} < c, \quad (2.28)$$

where v_x is the local velocity and c is a criterion, which for explicit solvers is typically equal to 1. Similarly, the stability of the 1D diffusion equation solved by the FDM is:

$$D \frac{\Delta t}{\Delta x^2} < c, \quad (2.29)$$

where D is the diffusion coefficient and c is a criterion, which for explicit solvers is equal to 0.5. Equations (2.28) and (2.29) can also be derived formally [77].

On the other hand, the *implicit methods* provide a stable time stepping, but require additional computation. In general, we have to solve a global system of equations in order to advance to the next time step. While the system of equations is usually sparse, banded, and even symmetric, such an approach could still be beneficial, particularly since advanced iterative solvers, e.g., the Biconjugate Gradient STABILized (BiCGSTAB) method, are quite efficient, especially with a good preconditioner. Since the solutions from two consecutive steps are in general similar, the preconditioner can essentially be the solution from the previous time step. Consequently, the system can often be solved in a single iteration step with a calculation complexity of $O(b^2N)$ for N equations. The complexity is still considerably higher than in the case of the explicit methods; however, the desired solution can be reached with a significantly smaller number of time steps. Still, the longer time steps have to be paid for by a more complex solution of the global system, i.e., more iterations in the sparse solver.

2.5 Summary of Solution Methodology

Let us summarize the constitutive algorithms that are needed for the implementation of the PDE solution methodology. After the discretization of the independent variables, a global system of algebraic equations or ODEs is obtained. If the PDE is time dependent, the solution is built step by step as it develops over time. The list of necessary constitutive steps of the solution methodology is given below, separately for strong and weak formulations, either with the mesh-based or the meshless approaches.

Strong mesh-based forms:

- mesh construction,
- approximation of derivatives in nodes,
- construction of the global system,
- solution of the global system in the implicit methods,
- matrix vector multiplication in the explicit methods.

Strong meshless forms:

- placing of discretization nodes,
- determination of the support domain,
- approximation of derivatives in nodes,
- construction of the global system,
- solution of the global system in the implicit methods,
- matrix vector multiplication in the explicit methods.

Weak mesh-based forms:

- mesh construction and determination of elements,
- interpolation or approximation of the trial function by nodal shape functions over each element,
- for each element integrate the weak form,
- construction of the global system,
- solution of the global system in the implicit methods,
- matrix vector multiplication in the explicit methods.

Weak meshless forms:

- placing of discretization nodes,
- determination of the nodal support domain for the construction of shape functions,
- interpolation or the approximation of the trial function,
- for each node, the integration of integrals in the weak form,
- construction of the global system,
- solution of the global system in the implicit methods,
- matrix vector multiplication in the explicit methods,
- reconstruction of the approximate solution from the fictitious parameters if the Kronecker delta property is not satisfied.

The listed supporting algorithms, i.e., domain discretization, determination of the support domain, interpolation and approximation, numerical integration, and the solution of a linear system, are described in Chap. 3.

Parallel Scientific Computing

Theory, Algorithms, and Applications of Mesh Based
and Meshless Methods

Trobec, R.; Kosec, G.

2015, XI, 107 p. 37 illus., 3 illus. in color., Softcover

ISBN: 978-3-319-17072-5