METODA ELEMENTÓW SKOŃCZONYCH

THE FINITE ELEMENT METHOD

X.1. Introduction

Many CFD practitioners prefer finite volume methods because the derivation of the discrete equations is based directly on the underlying physical principles, thus resulting in “physically sound” schemes. From a mathematical point of view, finite volume, difference, and element methods are closely related, and it is difficult to decide that one approach is superior to the others; these spatial discretization methods have different advantages and disadvantages.

Today the Finite Element Method (FEM) has been widely employed in solving field problems arising in modern industrial practices. The text in this chapter (section) is short introduction to the application of the FEM to the analysis of fluid flow which is a very common phenomenon in many processes of manufacturing and engineering.

In this chapter / section we shall introduce the reader to a finite element treatment of the equations of motion for various problems of fluid mechanics. Much of the activity in fluid mechanics has however pursued a finite difference formulation and more recently a derivative of this known as the finite volume technique. Competition between the newcomer of finite elements and established techniques of finite differences have appeared on the surface and led to a much slower adoption of the finite element process in fluid mechanics than in structures. The reasons for this are perhaps simple. In solid mechanics or structural problems, the treatment of continua arises only on special occasions. The engineer often dealing with structures composed of bar-like elements does not need to solve continuum problems. Thus his interest has focused on such continua only in more recent times. In fluid mechanics, practically all situations of flow require a two or three dimensional treatment and here approximation was frequently required. This accounts for the early use of finite differences in the 1950s before the finite element process was made available. However, as it was pointed out in book [Zie2000a], there are many advantages of using the finite element process. This not only allows a fully unstructured and arbitrary domain subdivision to be used but also provides an approximation which in self-adjoint problems is always superior to or at least equal to that provided by finite differences.

One advantage of the finite element method over finite difference methods is the relative easy with which the boundary conditions of the problem are handled [Bur1985]. Many physical problems have boundary conditions involving derivatives and the boundary of the region is irregularly shaped. Boundary conditions of this type are very difficult to handle using finite difference techniques, since each boundary conditions involving a derivative must be approximated by a difference quotient at the grid points, and irregular shaping of the boundary makes placing the grid points difficult.

The construction procedure in the FEM is independent of the particular boundary conditions of the problem.

X.2. The method of weighted residuals (Galerkin's method)

Some physical problems can be stated directly in the frame of variational principle which consists of determining the function which makes a certain integral statement called functional stationary. However the form of the variational principle is not always obvious and such a principle does not exist for many continuum problems.

As an alternative to solve such differential equations we may use a variety of weighted residual methods. Weighted residual methods are numerical techniques which can be used to solve a single or set of partial differential equations. Consider such a set in domain Ω with boundary $\delta\Omega=\Gamma$, where u is the exact solution and may represent a single variable or a column vector of variables. where at least the first order gradient in the variable is prescribed.

Applying the method of weighted residuals involves basically two steps. The first step is to assume the general functional behavior of the dependent field variable in some way so as to approximately satisfy the given differential equation and boundary conditions. Substitution of this approximation into the original differential equation and boundary conditions then results in some error called a residual. This residual is required to vanish in some average sense over the entrie solution domain. The second step is to solve the equation (or equations) resulting from the first step and thereby specialize the general functional form to a particular function, which then becomes the approximate solution sought. According to Galerkin's method, the weighting functions are chosen to be the same as the approximating functions.

Let us consider differential equation

$$L(u) = f \tag{x.y}$$

defined within a domain Ω and with boundary conditions specified at the boundary of Γ.

An operator L is said to be linear if and only if it satisfies the relation

$$L(a \cdot u + b \cdot v) = aL(u) + bL(v) \tag{x.y}$$

for any scalars a and b and dependent variables u and v. When an operator does not satisfy the above condition it is said to be nonlinear. The function u (i.e. solution) is not only required to satisfy the operator equation, it is also required to satisfy the boundary conditions associated with the operator.

In the weighted-residual method the solution u is approximated by expressions $\bar{u}$ of the form

$$\bar{u} = S_0 + \sum_{j=1}^{N} u_j S_j \qquad \text{(x.y)}$$

where S_j are trial functions, and S_0 must satisfy all the specified boundary conditions ($S_0 = 0$ if all the specified boundary conditions are homogeneous) of the problem, and S_i must satisfy the following conditions:

- S_j should be such that $L(S_j)$ is well defined and nonzero, i.e. sufficiently differentiable;
- S_j must satisfy at least the homogeneous form of the essential boundary conditions of the problem;
- for any N, the set $\{S_j, j = 1,2,\ldots N\}$ is linearly independent.

We begin by introducing the error, or residual, R_Ω in the approximation (by substitution of the approximation $\bar{u}$ into the operator equation) which is defined by

$$R_\Omega = L(\bar{u}) - f \qquad \text{(x.y)}$$

where $\bar{u}$ contains trial functions and satisfies the Dirichlet boundary conditions of $\bar{u} = u_0$ at $\Gamma_1 \subseteq \Gamma$. If the residual is smaller the approximation is better. It should be noted that R_Ω is a function of position in Ω.

Now we attempt to reduce this residual as close to zero as possible. If we have

$$\int_{\Omega} T_i R_{\Omega} d\Omega = 0 \qquad \text{(x.y)}$$

where $T_i, i = 1,2,\ldots,M$ is a set of arbitrary functions and $M \to \infty$, then it can be said that the residual R_{Ω} vanishes. Here T_i are called weighting functions which, in general, are not the same as the approximation (trial) functions S_i. Expanding above equation we have

$$\int_{\Omega} T_i \left(L(\bar{u}) - f \right) d\Omega = 0\,. \qquad \text{(x.y)}$$

A function $\bar{u}$ that satisfies above equation for every function T_i in Ω is a weak solution of the differential equation, whereas the strong solution $\bar{u}$ satisfies the differential equation at every point of Ω.

When the operator L is linear above equation can be simplified to the form

$$\sum_{j=1}^{N}\left(\int_{\Omega} T_i L(S_j)\,d\Omega\right)u_j = \int_{\Omega} T_i\left(f - L(S_0)\right)d\Omega \qquad \text{(x.y)}$$

or

$$\sum_{j=1}^{N} A_{ij}u_j = f_i \,. \qquad \text{(x.y)}$$

where

$$A_{ij} = \int_{\Omega} T_i L(S_j)\,d\Omega \qquad \text{(x.y)}$$

and

$$f_i = \int_{\Omega} T_i\left(f - L(S_0)\right)d\Omega \,. \qquad \text{(x.y)}$$

Note that the coefficients of matrix $\mathbf{A}$ is not symmetric $A_{ij} \neq A_{ji}$.

The weighted-residual method (when $T_i \neq S_i$) is also sometimes referred to as the Petrov-Galerkin method. For different choices of T_i the method is known by different names. We outline below the most frequently used methods.

The Galerkin method. For $T_i = S_i$ the weighted-residual method is known as the Galerkin method. When the operator is a linear differential operator of even order, the Galerkin method reduces to the Ritz method. In this case the resulting matrix will be symmetric because half of the differentiation can be transformed to the weight functions.

The least-squares method. The least-squares methods seeks a solution in the form (XXX) $\bar{u} = S_0 + \sum_{j=1}^{N} u_j S_j$ and determines the constants u_j by minimizing the integral of the square of the residual

$$\frac{\partial}{\partial u_i} \int_{\Omega} R_{\Omega}^2 d\Omega = 0 \qquad \text{(x.y)}$$

or

$$\int_{\Omega} \frac{\partial R_{\Omega}}{\partial u_i} R_{\Omega} d\Omega = 0 \qquad \text{(x.y)}$$

A comparison of Eq.(XXX) with Eq.(XXX) shows that $T_i = \dfrac{\partial R_\Omega}{\partial u_i}$. If L is a linear operator Eq.(XXX) becomes

$$\sum_{j=1}^{N}\left(\int_\Omega L(S_i)L(S_j)d\Omega\right)u_j = \int_\Omega L(S_i)\left(f - L(S_0)\right)d\Omega \qquad \text{(x.y)}$$

which yields a symmetric matrix but requires the same order of differentition as the operator equation.

The collocation method. The collocation method seeks approximate solution $\bar{u}$ by requiring the residual $R_\Omega = R_\Omega(\mathbf{x},\mathbf{u})$ in the equation to be identically to zero at N selected points $\mathbf{x}_i$, $i = 1,2,\ldots,N$ in the domain Ω

$$R_\Omega(\mathbf{x}_i, u_j) = 0\,. \qquad \text{(x.y)}$$

The selection of the points $\mathbf{x}_i$ is crucial in obtaining a well conditioned system of equations and ultimately in obtaining an accurate solution. The collocation points can be shown to be a special case of Eq. (XXX) $\int\limits_\Omega T_i R_\Omega d\Omega = 0$ for $T_i = \delta(\mathbf{x} - \mathbf{x}_i)$, where $\delta(\mathbf{x})$ is the Dirac delta function

$$\int\limits_\Omega f(\mathbf{x})\delta(\mathbf{x} - \xi)d\Omega = f(\xi)\,. \qquad \text{(x.y)}$$

The Courant method. To so-called Courant method combines the basic concepts of the Ritz method and the least-squares method (for linear operator). The method seeks approximate solution $\bar{u}$ by minimizing the modified quadratic functional

$$I_p(\bar{u}) = I(\bar{u}) + \frac{\alpha}{2}\left\| L(\bar{u}) - f \right\|^2 \qquad \text{(x.y)}$$

Where $I(u)$ is the quadratic functional associated with $L(u) = f$, when L is linear, and α is the penalty parameter (preassigned). Obviously the statement make sense only for operator equation that admit functional formulation.

X.3. Time discretization schemes

In time-dependent (unsteady) problems, the undetermined unknown parameters u_j are assumed to be functions of time, while the trial functions S_j are assumed to depend on spatial coordinates. This leads to two stages of solution, both of which employ approximation methods. In the solution of unsteady problems we can first consider the spatial approximation and the time (or timelike) approximation next. Such a strategy is commonly known as semidiscrete approximation in space. The spatial discretisation leave us with the first order ordinary differential equations with respect to time. There are numerous ways of accomplishing the discretisation of the time domain.

The first order system of equations

$$\mathbf{M}\dot{\mathbf{w}} + \mathbf{K}\mathbf{w} = \mathbf{r} \tag{x.1}$$

where $\overset{\bullet}{\mathbf{w}} = \frac{\partial \mathbf{w}}{\partial t}$ needs to be discretised in time. If we consider only single PDE we have $\mathbf{w} = \mathbf{u}$ else $\mathbf{w}$ is set of unknown parameters of few variables (e.g. velocity, temperature, pressure). The most commonly used method for such a system is the generalized mid-point or trapezoidal family of methods. The trapezoidal method applied to can be written as follows

$$\mathbf{M}(\mathbf{w}_{n+1}, t_{n+1})\dot{\mathbf{w}}_{n+1} + \mathbf{K}(\mathbf{w}_{n+1}, t_{n+1})\mathbf{w}_{n+1} = \mathbf{r}_{n+1} \tag{x.2}$$

and

$$\frac{\mathbf{w}_{n+\mathbf{1}} + \mathbf{w}_n}{2} = \frac{\mathbf{w}_{n+1} - \mathbf{w}_n}{\delta t_n} \tag{x.3}$$

Substituting (x.3) in (x.2) we obtain

$$\left(\frac{2\mathbf{M}_{n+1}}{\delta t}+\mathbf{K}_{n+1}\right)\mathbf{w}_{n+1}=\left(\frac{2}{\delta t}\mathbf{w}_{n}+\dot{\mathbf{w}}_{n}\right)\mathbf{M}_{n+1}+\mathbf{r}_{n+1} \qquad \text{(x.4)}$$

The method involves the calculation of the derivatives on the right hand side. Here $\mathbf{M}_{n+1}=\mathbf{M}(\mathbf{w}_{n+1},t_{n+1})$. The generalized mid-point family of methods is written as

$$\mathbf{M}(\mathbf{w}_{n+\alpha},t_{n+\alpha})\dot{\mathbf{w}}_{n+\alpha}+\mathbf{K}(\mathbf{w}_{n+\alpha},t_{n+\alpha})\mathbf{w}_{n+\alpha}=\mathbf{r}(\mathbf{w}_{n+\alpha},t_{n+\alpha}) \qquad \text{(x.5)}$$

where

$$\mathbf{w}_{n+1}=(1-\alpha)\mathbf{w}_{n}+\alpha\mathbf{w}_{n+1}, \qquad \text{(x.6)}$$

$$\dot{\mathbf{w}}_{n+\alpha} = \frac{\mathbf{w}_{n+1} - \mathbf{w}_n}{\delta t_n} \, , \tag{x.7}$$

$$t_{n+1} = t_n + \alpha \delta t_n = t_n + \alpha \delta t \, . \tag{x.8}$$

Substituting (x.6)-(x.8) into (x.5) we obtain

$$\left(\frac{\mathbf{M}_{n+\alpha}}{\delta t} + \alpha \mathbf{K}_{n+\alpha} \right) \mathbf{w}_{n+1} = \left(\frac{\mathbf{M}_{n+\alpha}}{\delta t} - (1 - \alpha) \mathbf{K}_{n+\alpha} \right) \mathbf{w}_n + \mathbf{r}_{n+\alpha} \, . \tag{x.9}$$

No calculation of derivatives is necessary for this method. By changing the values of α from 0 to 1, different members of this family of methods are identified.

We can obtain a number of well-known difference schemes by choosing the value of α:

α =0	forward difference (Euler) scheme (conditionally stable);
α =1/2	midpoint rule (Crank-Nicolson) scheme (unconditionally stable);
α =2/3	Galerkin method (unconditionally stable);
α =1	backward difference (backward Euler) scheme (conditionally stable).

All, except the forward Euler of the above schemes are implicit (they require matrix inversion for solution). As far as accuracy is concerned the midpoint rule is to be preferred. The generalized midpoint rule conserves linear and quadratic quantities, while the trapezoidal rule conserves only the linear ones.

It must be pointed out that one can except better results if smaller steps are used. In practice one wishes to take as large a time step as possible to cut down the computational expense. Larger time steps, in addition to decreasing the accuracy of the solution, can introduce some unwanted, numerically induced oscillations into the solution. Thus an estimate of an upper bound on time step proves to be very useful.

The system of equations for incompressible flow is non-linear and therefore an iterative solution is necessary within one time step. The fully discretised using the mentioned in this chapter generalized mid-point family of methods given in Eq.(xxxxxxx) may be written as

$$\left(\frac{\mathbf{M}_{n+\alpha}^{p}}{\delta t} + \alpha \mathbf{K}_{n+\alpha}^{p} \right) \mathbf{w}_{n+1}^{p+1} = \left(\frac{\mathbf{M}_{n+\alpha}^{p}}{\delta t} - (1-\alpha) \mathbf{K}_{n+\alpha}^{p} \right) \mathbf{w}_{n+\alpha}^{p} + \mathbf{r}_{n+\alpha}^{p} . \tag{x.9}$$

where p represents the iteration number. Above equation may be solved until the norm $\left| \mathbf{w}_{n+1}^{p+1} - \mathbf{w}_{n+1}^{p} \right|$ falls below an acceptable tolerance. To speed up convergence within one time step the Newton-Raphson method may be used [Hua1999].

X.4. Overview of the finite element method

There is an extensive literature on finite elements, both for theory and applications. Popular books include those by Huebner [Hue1975] (a definitive work from an engineering perspective), Hinton and Owen [Hin1979], Zienkiewicz and Taylor [Zie2000a, Zie2000b, Zie2000c].

In this chapter, we give a sketch of the finite element procedures. This sketch introduces important concepts of local approximation functions (linear and quadratic), the Galerkin method, treatment of boundary conditions, and assembly and solution of global matrices.

The governing equations of given problem must first be discrtetised spatially to obtain the finite element equations. The conventional Galerkin weighted residual technique discussed in previous section/chapter is the most powerful and general method available to achieve finite element spatial discretisation for any set of differential equations.

X.5. Local approximations

In the finite element method, the solution u of a PDE is approximated by low-order polynomials on local elements. The local elements constitute the mesh; typical elements used are triangles and quadrilaterals in 2D, and tetrahedra and hexahedra in 3D.

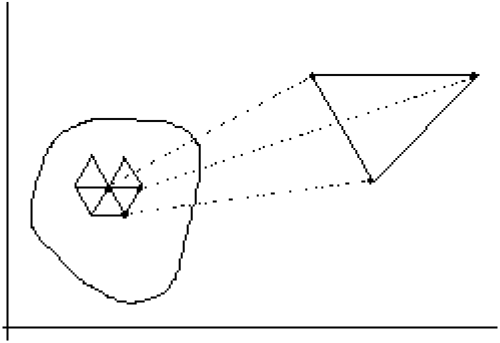


Figure 3.1: 2D triangular mesh.

To give a simple example, consider a triangular mesh in 2D (Figure 3.1).

We concentrate on the single triangle with corner nodes $\{i, j, k\}$, and let the values of u at the nodes be $\{u_i, u_j, u_k\}$. We approximate u within the local element by

$$\bar{u} = \left[N_i(x,y), N_j(x,y), N_k(x,y)\right] \cdot \left[u_i, u_j, u_k\right]^T \tag{x.y}$$

where $\{N_i(x,y), N_j(x,y), N_k(x,y)\}$ are interpolation functions. In the simplest case, these are linear polynomials such that

$$N_l(x_p, y_p) = \delta_{lp} \tag{x.y}$$

where δ_{lp} is the Kronecker symbol.

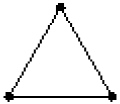
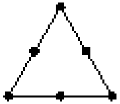
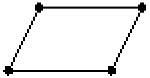
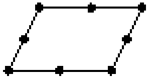
For example, if the local element is the triangle with nodes at $(0,0),(1,0),(1,1)$, the three linear interpolation functions are

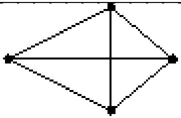
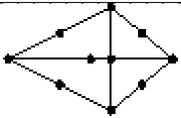
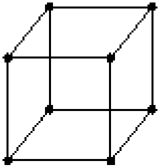
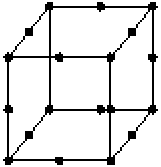
$$N_1 = 1 - x\,,\ N_2 = x - y\,,\ N_3 = y \qquad \text{(x.y)}$$

and, given nodal values (u_i, u_j, u_k), the linear approximation to u in the element is

$$\bar{u} = (1 - x)u_i + (x - y)u_j + yu_k\,. \qquad \text{(x.y)}$$

We can use (for example) the following element types:

 3-node triangle, linear approximation	 6-node triangle, quadratic approximation
 4-node quadrilateral, bi-linear approximation	 8-node quadrilateral, bi-quadratic approximation (serendipity element)

 4-node tetrahedron, linear approximation	 10-node tetrahedron, quadratic approximation
 8-node hexahedron, tri-linear approximation	 20-node hexahedron, tri-quadratic approximation (serendipity element)

Figures X. Examples of finite element.

X.6. Calculation of the nodal values

X.5.1. Solution of steady problems

The nodal values are pointwise approximations to the solution of a system of PDEs

$$L(u) = f \tag{x.y}$$

defined within a domain Ω and with boundary conditions specified at the boundary of Γ. The boundary conditions typically specify and/or its derivatives. If u is specified on the boundary, it is known as a Dirichlet boundary condition. A natural boundary condition specifies the value of terms arising from integration by parts, such as the flux. The PDE system might be a well-specified problem in its own right, or it might result from an algorithm applied to a more complex problem. Domain Ω can be in 2 or 3 dimensions, whilst both L and u can have multiple components. In the above, f represents a forcing term for the PDEs, and $L(u)$ typically includes derivatives of u up to second order.

We represent the solution $u(x)$ of the PDE system as follows:

$$u(x) = \sum u_i S_i(x) \qquad \text{(x.y)}$$

The PDE will be satisfied in the weak sense provided

$$\int_{\Omega} T_j(x)\{L(u) - f\}dV = 0\,,\;\; j = 1,\ldots,N_e \qquad \text{(x.y)}$$

for a given set of test functions T_j. If L has multiple components, then T has a corresponding number. In the Galerkin method as implemented in many codes, the shape functions S_i and the test functions T_j are identical. However, since the shape functions do not have second derivatives everywhere, we usually integrate some terms by parts prior to the substitution of the shape function representation for u. In the finite element method the shape function S_i for each node is continuous and identically zero outside the elements of which the node is a part. Within each of those elements, S_i is a low-order polynomial which takes the value one at node j and zero at all other nodes.

To accomplish the integration by parts, we symbolically decompose the operator L into first- and second-order operators

$$L = L_1 + \nabla L_2 \tag{x.y}$$

Here both L_1 and L_2 are first-order operators. L_2 may be vector or tensor valued, with possibly a reduction operation when the grad is applied. The weak form of the PDE thus gives

$$\begin{aligned}\int_{\Omega} T_j L(u)\,dV &= \int_{\Omega} T_j L_1(u)\,dV + \int_{\Omega} \nabla T_j L_2(u)\,dV - \int_{\Omega} L_2(u)\nabla T_j\,dV = \\ &= \int_{\Omega} T_j L_1(u)\,dV + \int_{\partial\Omega} \mathbf{n} T_j L_2(u)\,dS - \int_{\Omega} L_2(u)\nabla T_j\,dV = \\ &= \int_{\Omega} T_j f\,dV\end{aligned} \qquad \text{(x.y)}$$

where $\partial\Omega$ is the boundary and $\mathbf{n}$ is the unit outward normal. Using this integrated form of the PDE, it is now possible to approximate u using the shape functions. This process is known as assembly, and the end result is a finite dimensional system over the N_e nodes:

$$\sum_{j=1}^{N_e} K_{ij} u_j = r_i\,,\ i = 1,\ldots,N_e \qquad \text{(x.y)}$$

where

$$K_{ij} = \int_{\Omega} S_i L_1\left(S_j\right) dV - \int_{\Omega} L_2\left(S_j\right) \nabla S_i dV \tag{x.y}$$

and

$$r_i = -\int_{\partial\Omega} \mathbf{n} S_i L_2\left(u\right) dS + \int_{\Omega} S_i f dV \,. \tag{x.y}$$

The matrix $\mathbf{K}$ is called the stiffness or global matrix, and vector $\mathbf{r}$ is called the load vector.

X.6.2. Solution of time dependent problems

X.6.2.1. First strategy

Let us consider time-dependent PDE equation

$$\frac{\partial u}{\partial t} + L(u) = f \qquad \text{(x.y)}$$

defined within a domain Ω and with boundary conditions specified at the boundary of Γ. $L(u)$ typically includes derivatives of u up to second order.

The numerical strategies are based on discretizing governing equations first in time, to get a set of simpler partial differential equations, and then discretizing the time-discrete equations in space. There are two main discretizing in time schemes: backward Euler and Crank–Nicolson.

The backward Euler method uses the algorithm

$$\frac{u_{n+1} - u_n}{\delta t} + L(u_{n+1}) = f \qquad \text{(x.y)}$$

which is equivalent to

$$u_{n+1} + \delta t L(u_{n+1}) = \delta t (u_n + f)\,. \qquad \text{(x.y)}$$

In the heart of the algorithm, the equation (x.y) is assembled and solved at each timestep.

The Crank–Nicolson approximation uses the algorithm:

$$\frac{u_{n+1}-u_n}{\delta t}+\frac{1}{2}L\left(u_{n+1}+u_n\right)=f\ . \qquad \text{(x.y)}$$

Next define $\delta u = u_{n+1}-u_n$ and verify that δu satisfies

$$\delta u+\frac{\delta t}{2}L\left(\delta u\right)=-\delta t L\left(u_n\right)+\delta t\ f\ . \qquad \text{(x.y)}$$

The above equation is assembled and solved for δu at each timestep.

Let us consider the backward Euler approximation with the finite element method

$$u_{n+1} + \delta t L(u_{n+1}) = \delta t (u_n + f)\,. \qquad \text{(x.y)}$$

The PDE will be satisfied in the weak sense provided

$$\int_{\Omega} T_j(x)\{u_{n+1} + \delta t L(u_{n+1}) - \delta t (u_n + f)\} dV = 0\,,\ \ j = 1,\ldots,N_e \qquad \text{(x.y)}$$

for a given set of test functions T_j. To accomplish the integration by parts, we symbolically decompose the operator L into first- and second-order operators

$$L = L_1 + \nabla L_2 \qquad \text{(x.y)}$$

The weak form of the PDE thus gives

$$\begin{aligned}
&\int_\Omega T_j\left(u_{n+1} + \delta t L\left(u_{n+1}\right)\right) dV = \\
&= \int_\Omega T_j u_{n+1} dV + \int_\Omega T_j L_1\left(u_{n+1}\right) dV + \int_\Omega \nabla T_j L_2\left(u_{n+1}\right) dV - \int_\Omega L_2\left(u_{n+1}\right) \nabla T_j dV = \\
&= \int_\Omega T_j u_{n+1} dV + \int_\Omega T_j L_1\left(u_{n+1}\right) dV + \int_{\partial\Omega} \mathbf{n} T_j L_2\left(u_{n+1}\right) dS - \int_\Omega L_2\left(u_{n+1}\right) \nabla T_j dV = \\
&= \int_\Omega T_j \delta t\left(u_n + f\right) dV
\end{aligned} \qquad \text{(x.y)}$$

The end result is a finite dimensional system over the N_e nodes:

$$\sum_{j=1}^{N_e} K_{ij} u_j = r_i \,,\ i = 1,\ldots,N_e \qquad \text{(x.y)}$$

where

$$K_{ij} = \int_\Omega S_i S_j \, dV + \int_\Omega S_i L_1\left(S_j\right) dV - \int_\Omega L_2\left(S_j\right) \nabla S_i \, dV \qquad \text{(x.y)}$$

and

$$r_i = -\int_{\partial\Omega} \mathbf{n} S_i L_2\left(u_{n+1}\right) dS + \int_\Omega S_i \left(u_n + f\right) dV \,. \qquad \text{(x.y)}$$

X.6.2.2. Second strategy

Strategies for time-dependent problems presented in previous sections were based on discretizing governing equations first in time and then discretizing the time-discrete equations in space.

One can write finite element shape functions to include the time variable and thus incorporate it into the general finite element method procedure [Hua1999]. However, due to the conceptual simplicity of the time dimension simpler finite difference approximations presented in the previous section are generally favoured. Most schemes currently used are constructed in this way. We can also discretizing governing equations first in space.

Let us consider time-dependent PDE equation

$$\frac{\partial u}{\partial t} + L(u) = f \qquad \text{(x.y)}$$

defined within a domain Ω and with boundary conditions specified at the boundary of Γ. $L(u)$ typically includes derivatives of u up to second order.

The main numerical strategies are based on discretizing governing equations first in time, to get a set of simpler partial differential equations, and then discretizing the time-discrete equations in space.

The PDE will be satisfied in the weak sense provided

$$\int_{\Omega} T_j(x) \left\{ \frac{\partial u}{\partial t} + L(u) - f \right\} dV = 0\,, \ \ j = 1,\ldots,N_e \qquad \text{(x.y)}$$

for a given set of test functions T_j.

To accomplish the integration by parts, we symbolically decompose the operator L into first- and second-order operators $L = L_1 + \nabla L_2$. Here both L_1 and L_2 are first-order operators. The weak form of the PDE thus gives

$$\begin{aligned}
&\int_\Omega T_j \frac{\partial u}{\partial t} dV + \int_\Omega T_j L(u) dV = \\
&= \int_\Omega T_j \frac{\partial u}{\partial t} dV + \int_\Omega T_j L_1(u) dV + \int_\Omega \nabla T_j L_2(u) dV - \int_\Omega L_2(u) \nabla T_j dV = \\
&= \int_\Omega T_j \frac{\partial u}{\partial t} dV + \int_\Omega T_j L_1(u) dV + \int_{\partial\Omega} \mathbf{n} T_j L_2(u) dS - \int_\Omega L_2(u) \nabla T_j dV = \int_\Omega T_j f dV
\end{aligned} \quad . \qquad \text{(x.y)}$$

The end result is a finite dimensional system over the N_e nodes:

$$\sum_{j=1}^{N_e} M_{ij}\dot{u}_j + \sum_{j=1}^{N_e} K_{ij}u_j = r_i\,,\ i = 1,\ldots,N_e \tag{x.y}$$

where

$$M_{ij} = \int_{\Omega} S_i S_j dV \tag{x.y}$$

$$K_{ij} = \int_{\Omega} S_i L_1\left(S_j\right) dV - \int_{\Omega} L_2\left(S_j\right)\nabla S_i dV \tag{x.y}$$

and

$$r_i = -\int_{\partial\Omega} \mathbf{n} S_i L_2\left(u\right) dS + \int_{\Omega} S_i\, f dV\,. \tag{x.y}$$

The matrix $\mathbf{M}$ is called the mass matrix.

X.7. Assembly and sub-assembly

Although the components of $\mathbf{K}$ are written as integrals over the whole mesh, they are in fact zero everywhere except on elements containing both node i and node j. Nodes that have no element in common have a zero entry; hence $\mathbf{K}$ is a sparse matrix. Assembly in FEM is carried out element by element. Each pair of nodes of the element generates a component to be added to $\mathbf{K}$. These components are added into an element matrix, prior to being added into the global matrix. In this process, subsets of the global vectors required as data for assembly, including the coordinates, are selected and sorted into a standard nodal order for the element. This is referred to as the local level; the vectors are called local vectors. The integrals making up $\mathbf{K}$ have to be evaluated, and this is done by Gauss quadrature. Standard interpolation formulae are used to calculate the quantities concerned at quadrature points, and weighted sums of these values are used to approximate the integral.

X.8. Boundary conditions

The finite element method distinguishes between essential and natural boundary conditions:

a) essential (Dirichlet)

$$G(u) = g \qquad \text{(x.y)}$$

where the value of variable is prescribed;

b) natural (Neumann)

$$S(u) = s \qquad \text{(x.y)}$$

As an introduction to these concepts, consider the weak form of the left-hand side of Laplace's equation:

$$\int_{\Omega} S_i \nabla^2 u \, dV = -\int_{\Omega} \nabla S_i \cdot \nabla u \, dV + \int_{\partial\Omega} S_i \mathbf{n} \cdot \nabla u \, dS \qquad \text{(x.y)}$$

The last term is an integral over the boundary of the normal derivative (or flux). This is called a natural boundary condition; the boundary integrands represent a physical quantity (for example, flux in a diffusion problem, or stress in a linear elasticity problem). The condition is implemented by substituting directly if the integrand is known, or by substituting an expression involving unknowns. Natural boundary conditions are specified at the time that the PDE problem is specified. On the other hand, an absolute specification $u_i = c_i$ at some set of boundary nodes is called an essential boundary condition. This is enforced by including it in the set of equations, replacing the equation which had been formed by using S_i as a test function.

To illustrate this point, suppose that node 3 is a boundary node with value $u_3 = U$. The third row of the matrix is independent of other nodal values and is given by

$$[0,0,1,0,\dots,0]\cdot[u_1,u_2,u_3,\dots,u_N] = U \qquad \text{(x.y)}$$

Incorporation of this boundary condition into the matrix system gives the new system

$$\begin{bmatrix} K_{11} & K_{12} & K_{13} & K_{14} & \dots \\ K_{21} & K_{22} & K_{23} & K_{24} & \dots \\ 0 & 0 & 1 & 0 & \dots \\ K_{41} & K_{42} & K_{43} & K_{44} & \dots \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ \dots \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ U \\ r_4 \\ \dots \end{bmatrix} \qquad \text{(x.y)}$$

If the matrix $\mathbf{K}$ is symmetric it is necessary to do a further elimination to regain symmetry:

$$\begin{bmatrix} K_{11} & K_{12} & 0 & K_{14} & \dots \\ K_{21} & K_{22} & 0 & K_{24} & \dots \\ 0 & 0 & 1 & 0 & \dots \\ K_{41} & K_{42} & 0 & K_{44} & \dots \\ \dots & \dots & \dots & \dots & \dots \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ \dots \end{bmatrix} = \begin{bmatrix} r_1 - UK_{13} \\ r_2 - UK_{23} \\ U \\ r_4 - UK_{43} \\ \dots \end{bmatrix} \qquad \text{(x.y)}$$

To summarise the discussion so far, essential boundary conditions are implemented by modification of the global matrix and right-hand side (RHS) vector, whilst natural boundary conditions are often accounted for in the RHS vector alone. These concepts are so important, however, that we now provide a more detailed commentary.