1.

7 Formulation of Parabolic, Hyperbolic, and Elliptic-Eigenvalue Problems

| § 7.1-7.2
| § 8.1 and some comments from § 8.2
| § Some comments on § 9.1

In order to understand this chapter, a detailed understanding of the material and notational conventions of Chapter 2 is required.

7.1. PARABOLIC CASE: HEAT EQUATION

A particular parabolic case

The heat equation is the prototypical "parabolic" partial differential equation of mathematical physics (e.g., see Stakgold [1]). The formulation given here generalizes the formulation of Chapter 2, which was restricted to steady heat conduction, to time-dependent heat conduction processes. We view all the data to be functions of time, denoted by t, as well as space. For example, the volumetric heat source ℓ is written as

$$\ell : \Omega \times]0, T[\rightarrow \mathbb{R} \tag{7.1.1}$$

which means ℓ is a function of $x \in \Omega$ and $t \in]0, T[$, the open time interval of length $T > 0$. Likewise, we write $\ell = \ell(x, t)$. Similarly, the boundary data is also time-dependent:

$$g : \Gamma_g \times]0, T[\rightarrow \mathbb{R} \tag{7.1.2}$$

$$h : \Gamma_h \times]0, T[\rightarrow \mathbb{R} \tag{7.1.3}$$

Note that Γ_g and Γ_h are assumed *not* to vary with time. To render the initial/boundary-value problem well posed, an ***initial condition*** on temperature must also be specified. We denote the initial temperature by

Dealing with time varying Γ_g & Γ_h possible - just adds some additional tracking of things
This is not saying that g and h can not very with time - just says what they act on does not vary.

2.

Have to have initial conditions

$$u_0 : \Omega \to \mathbb{R} \tag{7.1.4}$$

Two other material properties come into play: The ***density***, ρ, and ***capacity***, c; both are assumed positive functions of $\boldsymbol{x} \in \Omega$.

The ***strong form*** of the initial/boundary-value problem may now be stated:

$$(S)\left\{\begin{array}{l}\text{Given } \ell, g, h, \text{ and } u_0, \text{ as in (7.1.1) through (7.1.4), find } u : \overline{\Omega} \times [0, T] \to \mathbb{R} \text{ such that}\\[4pt] \rho c u_{,t} + q_{i,i} = \ell \quad \text{on} \quad \Omega \times]0, T[\qquad (\textbf{\textit{heat equation}}) \qquad (7.1.5)\\ u = g \quad \text{on} \quad \Gamma_g \times]0, T[\qquad (7.1.6)\\ -q_i n_i = h \quad \text{on} \quad \Gamma_h \times]0, T[\qquad (7.1.7)\\ u(\boldsymbol{x}, 0) = u_0(\boldsymbol{x}) \quad \boldsymbol{x} \in \Omega \qquad (7.1.8)\end{array}\right.$$

Recall from Chapter 2 that

$$q_i = -\kappa_{ij} u_{,j} \tag{7.1.9}$$

where $\kappa_{ij} = \kappa_{ij}(\boldsymbol{x})$ denotes the conductivity tensor. The notation $u_{,t}$ denotes the time derivative of u (i.e., $u_{,t} = \partial u / \partial t$).

Let $\mathcal{V}$ denote the usual space of weighting functions satisfying zero-temperature boundary conditions. Note that functions in $\mathcal{V}$ do *not* depend on time in any way. Let $\mathcal{S} = \mathcal{S}_t$ denote the space of trial solutions. Note that $\mathcal{S}$ varies as a function of t due to the temperature boundary condition, (7.1.2),

$$\mathcal{S} = \mathcal{S}_t = \left\{ u(\cdot\,, t) \,\middle|\, \underbrace{u(\boldsymbol{x}, t) = g(\boldsymbol{x}, t),\ \boldsymbol{x} \in \Gamma_g}_{\text{essential BC}},\ u(\cdot\,, t) \in H^1(\Omega) \right\} \tag{7.1.10}$$

The ***weak formulation*** consists of the following:

$$(W)\left\{\begin{array}{l}\text{Given } \ell, g, h, \text{ and } u_0, \text{ find } u(t) \in \mathcal{S}_t;\ t \in [0,T] \text{ such that for all } w \in \mathcal{V},\\[6pt] \boxed{(w, \rho c \dot{u}) + a(w, u) = (w, \ell) + (w, h)_\Gamma} \qquad (7.1.11)\\[6pt] (w, \rho c u(0)) = (w, \rho c u_0) \qquad (7.1.12)\end{array}\right.$$

Initial Condition

Remark

In the weak formulation, $\boldsymbol{x}$ is suppressed as an argument of u. Thus $u(0)$ represents the function u of $\boldsymbol{x}$ at time 0 [i.e., $u(0) = u(\cdot\,, 0)$]. The superposed dot is used to denote time differentiation. These notations, among other things, enable us to remove commas from the weak formulation which might be confused with those normally appearing in the bilinear forms. Equation (7.1.11) is the weak formulation of the heat equation (7.1.5) and heat-conduction boundary condition (7.1.7), and (7.1.12) is the weak form of the initial condition (7.1.8).

Semi discretization of time dependent heat conduction

MWR $$\int_\Omega w(\rho c u_{,t} + q_{i,i} - f)\, d\Omega$$

$$\int_\Omega w\rho c u_{,t}\, d\Omega + \int_\Omega w q_{i,i}\, d\Omega - \int_\Omega w f\, d\Omega$$

$$\int_\Omega w\rho c u_{,t}\, d\Omega - \int_\Omega w_{,i} q_i\, d\Omega + \int_\Gamma w q_i n_i\, d\Gamma - \int_\Omega w f\, d\Omega$$

using: $q_i = -K_{ij} u_{,j}$ in Ω, $-q_i n_i = h$ on Γ_h
and $w|_{\Gamma_g} = 0$

$$\int_\Omega w\rho c u_{,t} + \int_\Omega w_{,i} K_{ij} u_{,j}\, d\Omega = \int_\Omega w f\, d\Omega + \int_{\Gamma_h} w h\, d\Gamma$$

$$(w, \rho c \dot{u}) + \underbrace{a(w,u) = (w,f) + (w,h)_\Gamma}_{\text{identical to the static case}}$$

$$\dot{u} = u_{,t} = \frac{\partial u}{\partial t}$$

4.

Exercise 1. Verify the formal equivalence of (S) and (W) by proceeding along the lines of Sec. 2.3. The procedure is virtually identical to Sec. 2.3 if (7.1.11) is written as

$$a(w, u) = (w, \tilde{\ell}) + (w, h)_\Gamma$$

where

$$\tilde{\ell} = \ell - \rho c \dot{u}$$

To develop the analogous Galerkin formulation, we proceed as in Sec. 2.4. The approximate weighting function space, $\mathcal{V}^h$, is identical to before. Functions $u^h \in \mathcal{S}^h$ are built up in the usual way:

$$u^h = v^h + g^h \quad , \dot{u}^h = \dot{v}^h + \dot{g}^h \leftarrow \text{also needed} \tag{7.1.13}$$

where $v^h(t) \in \mathcal{V}^h$ (i.e, for each fixed t, $v^h(\cdot\,, t)$ is a function in $\mathcal{V}^h$) and g^h enables satisfaction of the temperature boundary condition. Note that $g^h(t) \in \mathcal{S}_t^h$. All the functions in (7.1.13) depend upon both space and time, i.e.,

$$u^h(x, t) = v^h(x, t) + g^h(x, t) \tag{7.1.14}$$

The ***Galerkin formulation*** is stated as follows:

(G) Given ℓ, g, h, and u_0, find $u^h = v^h + g^h$, $u^h(t) \in \mathcal{S}_t^h$, such that for all $w^h \in \mathcal{V}^h$,

$$\boxed{(w^h, \rho c \dot{v}^h) + a(w^h, v^h) = (w^h, \ell) + (w^h, h)_\Gamma - \overbrace{(w^h, \rho c \dot{g}^h) - a(w^h, g^h)}^{\text{Two terms related to essential BC}}} \tag{7.1.15}$$

$$(w^h, \rho c v^h(0)) = (w^h, \rho c u_0) - (w^h, \rho c g^h(0)) \tag{7.1.16}$$

Note that all given quantities appear on the right-hand sides of (7.1.15) and (7.1.16). The Galerkin equation given by (7.1.15) is called a ***semidiscrete*** equation because time is left continuous.

The matrix equations require representations of v^h and g^h in terms of basis functions. These are given by

$$v^h(x, t) = \sum_{A \in \eta - \eta_g} N_A(x) d_A(t) \tag{7.1.17}$$

$$g^h(x, t) = \sum_{A \in \eta_g} N_A(x) g_A(t) \tag{7.1.18}$$

In this form ↓

Note that the entire time-dependence is carried by the nodal values (i.e., the d_A's and g_A's). The shape functions are identical to those used previously (see Chapters 2 and 3). In particular, they are *not* time-dependent.

Remark

Upon encountering representations such as (7.1.17) and (7.1.18) for the first time, we may wonder if the approximation is valid only when the exact solution is "separable" in x and t (e.g., see [1] for a discussion of the separation-of-variables

5.

technique). This is not the case. Even nonseparable exact solutions may be approximated arbitrarily closely by representations like (7.1.17) and (7.1.18).

To develop the matrix equations, we substitute (7.1.17) and (7.1.18) into (7.1.15) and (7.1.16) and proceed along the same lines as in Secs. 2.4 and 2.5. The end result is the following ***matrix problem***:

Given $\boldsymbol{F}:]0, T[\rightarrow \mathbb{R}^{n_{eq}}$, find $\boldsymbol{d}: [0, T] \rightarrow \mathbb{R}^{n_{eq}}$ such that

$$\boxed{\begin{aligned} \boldsymbol{M}\boldsymbol{d} + \boldsymbol{K}\boldsymbol{d} &= \boldsymbol{F} \qquad t \in]0, T[\\ \boldsymbol{d}(0) &= \boldsymbol{d}_0 \end{aligned}} \qquad \begin{aligned} &(7.1.19) \\ &(7.1.20) \end{aligned}$$

where

$$\boldsymbol{M} = \underset{e=1}{\overset{n_{el}}{\mathbf{A}}} (\boldsymbol{m}^e) \tag{7.1.21}$$

$$\boxed{\begin{aligned} \boldsymbol{m} &= [m_{ab}^e] \\ m_{ab}^e &= \int_{\Omega^e} N_a \rho c N_b \, d\Omega \end{aligned}} \qquad \begin{aligned} &(7.1.22) \\ &(7.1.23) \end{aligned}$$

$$\boldsymbol{K} = \underset{e=1}{\overset{n_{el}}{\mathbf{A}}} (\boldsymbol{k}^e) \tag{7.1.24}$$

(M)

$$\boxed{\begin{aligned} \boldsymbol{k}^e &= [k_{ab}^e] \\ k_{ab}^e &= \int_{\Omega^e} \boldsymbol{B}_a^T \boldsymbol{D} \boldsymbol{B}_b \, d\Omega \end{aligned}} \qquad \begin{aligned} &(7.1.25) \\ &(7.1.26) \end{aligned}$$

$$\boldsymbol{F}(t) = \boldsymbol{F}_{\text{nodal}}(t) + \underset{e=1}{\overset{n_{el}}{\mathbf{A}}} (\boldsymbol{f}^e(t)) \tag{7.1.27}$$

$$\boxed{\begin{aligned} \boldsymbol{f}^e &= \{f_a^e\} \\ f_a^e &= \int_{\Omega^e} N_a \ell \, d\Omega + \int_{\Gamma_h^e} N_a h \, d\Gamma - \sum_{b=1}^{n_{en}} (k_{ab}^e g_b^e + m_{ab}^e \dot{g}_b^e) \end{aligned}} \qquad \begin{aligned} &(7.1.28) \\ &(7.1.29) \end{aligned}$$

$$\boldsymbol{d}_0 = \boldsymbol{M}^{-1} \underset{e=1}{\overset{n_{el}}{\mathbf{A}}} (\hat{\boldsymbol{d}}^e) \tag{7.1.30}$$

$$\hat{\boldsymbol{d}}^e = \{\hat{d}_a^e\} \tag{7.1.31}$$

$$\hat{d}_a^e = \int_{\Omega^e} N_a \rho c u_0 \, d\Omega - \sum_{b=1}^{n_{en}} m_{ab}^e g_b^e(0) \tag{7.1.32}$$

Remarks

1. If we compare with the steady formulation of Chapter 2, we see that the only new matrix which appears is M, the ***capacity matrix***. That M is symmetric and positive-definite follows directly from its definition.

2. Equation (7.1.19) is a coupled system of ordinary differential equations. Thus to solve the initial-value problem, i.e., (7.1.19) and (7.1.20), we need to introduce algorithms for solving systems of ordinary differential equations. This subject is taken up in subsequent chapters.

3. Observe in (7.1.29) that the element capacity matrices come into play in accounting for the effect of specified boundary temperatures.

4. It is common in practice to simplify the specification of initial conditions. That is, rather than employ (7.1.30) through (7.1.32), which emanate from (7.1.16), we directly specify $\boldsymbol{d}_0$ such that the given nodal values are interpolated [i.e., $d_{0A} = u_0(x_A)$, $A \in \eta - \eta_g$].

5. The element capacity matrix $\boldsymbol{m}^e$ turns out to be virtually identical to the element "mass" matrix, which will be introduced in the next section. Consequently, we shall postpone more detailed consideration of the structure of $\boldsymbol{m}^e$ until later.

6. Recall that $g_b^e(t) = 0$ if degree of freedom b of element e is *not* specified. The same rule applies to $\dot{g}_b^e$.

Exercise 2. The details of arriving at (7.1.19) through (7.1.32) are straightforward given familiarity with the analogous developments in Chapter 2. If the results are not "obvious," the reader should fill in all details in step-by-step fashion.

Example

The one-dimensional heat equation is, of course, just a special case of the general formulation above. A strong form of the initial/boundary-value problem can be set which is a generalization of the one-dimensional model problem considered in Chapter 1. The equations are

$$(S)\quad \begin{cases} \rho c u_{,t} - (\kappa u_{,x})_{,x} = \ell & \text{on }]0, L[\times]0, T[\qquad (7.1.33) \\ u(L, t) = g(t) & t \in]0, T[\qquad (7.1.34) \\ -\kappa u_{,x}(0, t) = h(t) & t \in]0, T[\qquad (7.1.35) \\ u(x, 0) = u_0(x) & x \in]0, L[\qquad (7.1.36) \end{cases}$$

The element arrays are

$$\boxed{\begin{aligned} m_{ab}^e &= \int_{\Omega^e} N_a \rho c N_b \, d\Omega && (7.1.37) \\ k_{ab}^e &= \int_{\Omega^e} N_{a,x} \kappa N_{b,x} \, d\Omega && (7.1.38) \\ f_a^e &= \int_{\Omega^e} N_a \ell \, d\Omega + N_a(0) \delta_{e1} h - \sum_{b=1}^{n_{en}} (k_{ab}^e g_b^e + m_{ab}^e \dot{g}_b^e) && (7.1.39) \end{aligned}}$$

7.2 HYPERBOLIC CASE: ELASTODYNAMICS AND STRUCTURAL DYNAMICS ← A particular hyperbolic case

The developments of this section generalize those of Secs. 2.7 through 2.10. The nature of the generalization is similar in format to that of the previous section and the reader should first become familiar with Sec. 7.1 before considering this section even if he or she is uninterested in heat conduction.

The initial conditions this time involve specification of both displacements and velocities. Thus

$$u_{0i} : \Omega \rightarrow \mathbb{R} \tag{7.2.1}$$

and

$$\dot{u}_{0i} : \Omega \rightarrow \mathbb{R} \tag{7.2.2}$$

need both since we will have second order time derivatives coming

are given functions for each i, $1 \leq i \leq n_{sd}$. The superposed-dot notation in (7.2.2) is symbolic rather than operational.

The remaining prescribed data are

$$\ell_i : \Omega \times]0, T[\rightarrow \mathbb{R} \tag{7.2.3}$$

$$g_i : \Gamma_{g_i} \times]0, T[\rightarrow \mathbb{R} \tag{7.2.4}$$

$$h_i : \Gamma_{h_i} \times]0, T[\rightarrow \mathbb{R} \tag{7.2.5}$$

The ***density***, $\rho : \Omega \rightarrow \mathbb{R}$, assumed to be positive, needs also to be specified in the present case.

The ***strong form*** of the initial/boundary-value problem is

$$(S)\left\{\begin{array}{lll}
\multicolumn{3}{l}{\text{Given } \ell_i,\ g_i,\ h_i,\ u_{0i} \text{ and } \dot{u}_{0i}, \text{ as in (7.2.1) through (7.2.5), find } u_i : \overline{\Omega} \times [0, T] \rightarrow \mathbb{R} \text{ such that}} \\
\rho u_{i,tt} = \sigma_{ij,j} + \ell_i & \text{on } \Omega \times]0, T[\quad (\textbf{\textit{equation of motion}}) & (7.2.6) \\
u_i = g_i & \text{on } \Gamma_{g_i} \times]0, T[& (7.2.7) \\
\sigma_{ij} n_j = h_i & \text{on } \Gamma_{h_i} \times]0, T[& (7.2.8) \\
u_i(x, 0) = u_{0i}(x) & x \in \Omega & (7.2.9) \\
u_{i,t}(x, 0) = \dot{u}_{0i}(x) & x \in \Omega & (7.2.10)
\end{array}\right.$$

Recall that $\sigma_{ij} = c_{ijkl} u_{(k,l)}$ and $c_{ijkl} = c_{ijkl}(x)$. Note that the second time derivative (i.e., acceleration) appears in (7.2.6). This is the reason that two initial conditions are required.

The corresponding ***weak formulation*** is:[1]

[1] We now use "direct notation"; see Chapter 2 for elaboration.

Semidiscretization -

$$\int_\Omega w_i(\rho a_{i,tt} - \sigma_{ij,j} + f_i)\, d\Omega = 0$$

$$\int_\Omega w_i u_{i,tt} - \int_\Omega w_i \sigma_{ij,j}\, d\Omega + \int_\Omega w_i f_i\, d\Omega = 0$$

$$\int_\Omega w_i u_{i,tt}\, d\Omega + \int_\Omega w_{(i,j)} \sigma_{ij}\, d\Omega - \int_\Gamma w_i \sigma_{ij} n_j\, d\Gamma + \int_\Omega w_i f_i\, d\Omega$$

Using $\sigma_{ij} = C_{ijkl}\, u_{(k,l)}$ in Ω, $\sigma_{ij} n_j = h_i$ on Γ_{h_i} and $w|_{\Gamma_{h_i}} = 0$

$$\int_\Omega w_i u_{i,tt} + \int_\Omega w_{(i,j)} C_{ijkl}\, u_{(k,l)}\, d\Omega = \int_\Omega w_i f_i\, d\Omega + \sum_{i=1}^{n_{sd}} \left(\int_{\Gamma_{h_i}} \underbrace{w_i h_i}_{\text{no sum its}}\, d\Gamma \right)$$

here

$$(\underset{\sim}{w}, \rho \ddot{\underset{\sim}{u}}) + a(\underset{\sim}{w}, \underset{\sim}{u}) = (\underset{\sim}{w}, \underset{\sim}{f}) + (\underset{\sim}{w}, \underset{\sim}{h})_\Gamma$$

$$(W)\begin{cases}\text{Given } \ell, g, h, u_0, \text{ and } \dot{u}_0, \text{ find } u(t) \in \mathcal{S}_t,\ t \in [0, T], \text{ such that for all } w \in \mathcal{V}, \\ \boxed{(w, \rho\ddot{u}) + a(w, u) = (w, \ell) + (w, h)_\Gamma} \qquad (7.2.11) \\ (w, \rho u(0)) = (w, \rho u_0) \qquad (7.2.12) \\ (w, \rho\dot{u}(0)) = (w, \rho\dot{u}_0) \qquad (7.2.13)\end{cases}$$

9

Exercise 1. Verify the formal equivalence of (S) and (W). The arguments of Chapter 2 may be used virtually unaltered if (7.2.11) is written as

$$a(w, u) = (w, \tilde{\ell}) + (w, h)_\Gamma$$

where

$$\tilde{\ell} = \ell - \rho\ddot{u}$$

The ***semidiscrete Galerkin formulation of elastodynamics*** is:

$$(G)\begin{cases}\text{Given } \ell, g, h, u_0, \text{ and } \dot{u}_0, \text{ find } u^h = v^h + g^h,\ u^h(t) \in \mathcal{S}_t^h, \text{ such that for all } w^h \in \mathcal{V}^h, \\ \boxed{(w^h, \rho\ddot{v}^h) + a(w^h, v^h) = (w^h, \ell) + (w^h, h)_\Gamma - (w^h, \rho\ddot{g}^h) - a(w^h, g^h)} \qquad (7.2.14) \\ (w^h, \rho v^h(0)) = (w^h, \rho u_0) - (w^h, \rho g^h(0)) \qquad (7.2.15) \\ (w^h, \rho\dot{v}^h(0)) = (w^h, \rho\dot{u}_0) - (w^h, \rho\dot{g}^h(0)) \qquad (7.2.16)\end{cases}$$

The representations of v^h and g^h are given by

$$v_i^h(x, t) = \sum_{A\in\eta-\eta_{g_i}} N_A(x) d_{iA}(t) \qquad (7.2.17)$$

$$g_i^h(x, t) = \sum_{A\in\eta_{g_i}} N_A(x) g_{iA}(t) \qquad (7.2.18)$$

These and the usual arguments lead to the ***matrix problem:***

Given F: $]0, T[\to \mathbb{R}^{n_{eq}}$, find d: $]0, T[\to \mathbb{R}^{n_{eq}}$ such that

$$M\ddot{d} + Kd = F \qquad t \in]0, T[\qquad (7.2.19)$$

$$d(0) = d_0 \qquad (7.2.20)$$

$$\dot{d}(0) = \dot{d}_0 \tag{7.2.21}$$

(M)

where

$$M = \mathop{\mathbf{A}}_{e=1}^{n_{el}} (m^e)$$

$$m^e = [m^e_{pq}] \tag{7.2.23}$$

$$m^e_{pq} = \delta_{ij} \int_{\Omega^e} N_a \rho N_b \, d\Omega \tag{7.2.24}$$

(Recall $p = n_{ed}(a-1) + i$ and $q = n_{ed}(b-1) + j$)

$$K = \mathop{\mathbf{A}}_{e=1}^{n_{el}} (k^e) \tag{7.2.25}$$

$$k^e = [k^e_{pq}] \tag{7.2.26}$$

$$k^e_{pq} = e_i^T \int_{\Omega^e} B_a^T D B_b \, d\Omega \, e_j \tag{7.2.27}$$

$$F(t) = F_{\text{nodal}}(t) + \mathop{\mathbf{A}}_{e=1}^{n_{el}} (f^e(t)) \tag{7.2.28}$$

$$f^e = \{f^e_p\} \tag{7.2.29}$$

$$f^e_p = \int_{\Omega} N_a f_i \, d\Omega + \int_{\Gamma^e_{h_i}} N_a h_i \, d\Gamma - \sum_{q=1}^{n_{ee}} (k^e_{pq} g^e_q + m^e_{pq} \ddot{g}^e_q) \quad \text{(no sum on } i\text{)} \tag{7.2.30}$$

$$d_0 = M^{-1} \mathop{\mathbf{A}}_{q=1}^{n_{el}} (\hat{d}^e) \tag{7.2.31}$$

$$\hat{d}^e = \{\hat{d}^e_p\} \tag{7.2.32}$$

$$\hat{d}^e_p = \int_{\Omega^e} N_a \rho u_{0i} \, d\Omega - \sum_{q=1}^{n_{ee}} m^e_{pq} g^e_q(0) \tag{7.2.33}$$

$$\dot{d}_0 = M^{-1} \mathop{\mathbf{A}}_{e=1}^{n_{el}} (\hat{\dot{d}}^e) \tag{7.2.34}$$

$$\hat{\dot{d}}^e = \{\hat{\dot{d}}^e_p\} \tag{7.2.35}$$

$$\hat{\dot{d}}^e_p = \int_{\Omega^e} N_a \rho \dot{u}_{0i} \, d\Omega - \sum_{q=1}^{n_{ee}} m^e_{pq} \dot{g}^e_q(0) \tag{7.2.36}$$

Remarks

1. The main addition to what we have encountered previously in the elastostatics formulation of Chapter 2 is the ***mass matrix***, M. The reader should verify that M is symmetric and positive-definite. Except for the Kronecker delta and different material parameter inside the integrand, the mass and capacity matrix [(7.1.21) through (7.1.23)] are identical. To appreciate the origin of the Kronecker delta, we will sketch part of the calculation that leads to the matrix formulation. If we restrict attention to the eth element, then

$$\begin{aligned}(w^h, \rho\ddot{u}^h)^e &= \int_{\Omega^e} w_i^h\, \rho\, \ddot{u}_i^h\, d\Omega \\ &= \delta_{ij} \int_{\Omega^e} w_i^h\, \rho\, \ddot{u}_j^h\, d\Omega \qquad (7.2.37) \\ &= \sum_{A,B} c_{iA}\, \delta_{ij} \int_{\Omega^e} N_A\, \rho\, N_B\, d\Omega\, \ddot{d}_{jB} \quad \text{(summation of } i \text{ and } j \text{ implied)}\end{aligned}$$

2. Equation (7.2.19) is a coupled system of second-order ordinary differential equations. Algorithms for solving equations of this type are described in Chapter 9.
3. Note that the element mass is involved in the adjustment of forces due to nonzero boundary accelerations (see (7.2.30)).
4. As mentioned in the previous section, we usually simplify the specification of initial conditions in practice. Nodal interpolates in this case take the form

$$d_{0P} = u_{0i}(x_A) \qquad (7.2.38)$$

$$\dot{d}_{0P} = \dot{u}_{0i}(x_A) \qquad (7.2.39)$$

where $P = \text{LM}(i, A)$. Recall LM is the array described in Chapter 2.

Exercise 2. Fill in the omitted details leading to the matrix formulation of elastodynamics.

Viscous Damping

In structural dynamics we often work with systems of the form

$$\boxed{M\ddot{d} + C\dot{d} + Kd = F} \qquad (7.2.40)$$

where C is the ***viscous damping matrix***. A particularly convenient form of C is the ***Rayleigh damping matrix***

$$\boxed{C = aM + bK} \qquad (7.2.41)$$

where a and b are parameters. The two constituents of Rayleigh damping are seen to be mass and stiffness proportional. We would like to enlarge our theoretical framework to include Rayleigh damping. The necessary modifications are as follows: Replace the equation of motion, (7.2.6), by

$$\rho u_{i,tt} + a\rho u_{i,t} = \sigma_{ij,j} + \ell_i \tag{7.2.42}$$

where the generalized Hooke's law is modified to account for the stiffness proportional effect, namely,

$$\sigma_{ij} = c_{ijkl}(u_{(k,l)} + b\,\dot{u}_{(k,l)}) \tag{7.2.43}$$

In addition to the appearance of the $\mathbf{C}\dot{\mathbf{d}}$-term in (7.2.40), the effect of the viscous damping matrix is also felt in modifying the forces due to prescribed displacement boundary conditions. Specifically,

$$f_p^e = \text{right-hand side of (7.2.30)} - \sum_{q=1}^{n_{ee}} c_{pq}^e\, \dot{g}_q^e \tag{7.2.44}$$

where

$$\mathbf{c}^e = a\mathbf{m}^e + b\mathbf{k}^e \tag{7.2.45}$$

Everything else remains the same. The parameters a and b may be selected to produce desired damping characteristics.

Example

In one dimension, the above formulation leads to a ***wave equation***. This also may be viewed as a generalization of the one-dimensional model problem of Chapter 1. Various interpretations are possible. For example, the axial motion of an elastic rod of length L is governed by the equation

$$\rho u_{,tt} = \sigma_{,x} + \ell \qquad \text{on }]0, L[\times]0, T[\tag{7.2.46}$$

where

$$\sigma = E\, u_{,x} \tag{7.2.47}$$

and $E = E(x)$ is Young's modulus. Boundary and initial conditions may be specified in analogous fashion to Chapter 1, namely,

$$u(L, t) = g(t) \qquad t \in]0, T[\tag{7.2.48}$$

$$-Eu_{,x}(0, t) = h(t) \qquad t \in]0, T[\tag{7.2.49}$$

$$u(x, 0) = u_0(x) \qquad x \in]0, L[\tag{7.2.50}$$

$$\dot{u}(x, 0) = \dot{u}_0(x) \qquad x \in]0, L[\tag{7.2.51}$$

The resulting element arrays are virtually identical to the ones encountered in the one-dimensional heat equation example at the end of Sec. 7.1, viz.,

$$m_{ab}^e = \int_{\Omega^e} N_a \rho N_b \, d\Omega \tag{7.2.52}$$

13

8 Algorithms for Parabolic Problems

8.1 ONE-STEP ALGORITHMS FOR THE SEMIDISCRETE HEAT EQUATION: GENERALIZED TRAPEZOIDAL METHOD

Recall that the semidiscrete heat equation can be written as

$$\boldsymbol{M}\dot{\boldsymbol{d}} + \boldsymbol{K}\boldsymbol{d} = \boldsymbol{F} \tag{8.1.1}$$

where $\boldsymbol{M}$ is the capacity matrix, $\boldsymbol{K}$ is the conductivity matrix, $\boldsymbol{F}$ is the heat supply vector, $\boldsymbol{d}$ is the temperature vector, and $\dot{\boldsymbol{d}}$ is the time (t) derivative of $\boldsymbol{d}$. The matrices $\boldsymbol{M}$ and $\boldsymbol{K}$ are assumed symmetric, $\boldsymbol{M}$ is positive-definite, and $\boldsymbol{K}$ is positive-semidefinite (usually $\boldsymbol{K}$ is positive-definite too). The heat supply is a prescribed function of t. We write $\boldsymbol{F} = \boldsymbol{F}(t)$ for $t \in [0, T]$. Equation (8.1.1) is to be thought of as a coupled system of n_{eq} ordinary differential equations.

The initial-value problem consists of finding a function $\boldsymbol{d} = \boldsymbol{d}(t)$ satisfying (8.1.1), and the initial condition

$$\boldsymbol{d}(0) = \boldsymbol{d}_0 \tag{8.1.2}$$

where $\boldsymbol{d}_0$ is given.

Perhaps the most well known and commonly used algorithms for solving (8.1.1) are members of the ***generalized trapezoidal family of methods,*** which consists of the following equations:

14

$$\boxed{\begin{aligned} Mv_{n+1} + Kd_{n+1} &= F_{n+1} \\ d_{n+1} &= d_n + \Delta t \, v_{n+\alpha} \\ v_{n+\alpha} &= (1-\alpha)v_n + \alpha v_{n+1} \end{aligned}}$$

(8.1.3)
(8.1.4)
(8.1.5)

where d_n and v_n are the approximations to $d(t_n)$ and $\dot{d}(t_n)$, respectively; $F_{n+1} = F(t_{n+1})$; Δt is the time step, assumed constant for the time being; and α is a parameter, taken to be in the interval $[0, 1]$.

Some well-known members of the generalized trapezoidal family are identified below.

α	Method
0	Forward differences; forward Euler
$\frac{1}{2}$	Trapezoidal rule; midpoint rule; Crank-Nicolson
1	Backward differences; backward Euler

So that the reader has a basic appreciation of the computations entailed by the algorithm, we will present next a brief overview of implementational considerations.

Implementation 1: v-form

The computational problem is to determine d_{n+1} and v_{n+1} given d_n and v_n. The procedure begins at $n = 0$ with d_0 known. The initial value, v_0, may be determined from the time-discrete heat equation evaluated at $t = 0$:

$$Mv_0 = F_0 - Kd_0 \tag{8.1.6}$$

There are several ways of implementing the recursion relationship that takes us from step n to step $n + 1$. Let the ***predictor value*** of d_{n+1} be defined by

$$\boxed{\tilde{d}_{n+1} = d_n + (1-\alpha)\Delta t \, v_n} \tag{8.1.7}$$

Note that (8.1.4) and (8.1.5) can be combined and expressed in terms of (8.1.7):

$$d_{n+1} = \tilde{d}_{n+1} + \alpha \Delta t \, v_{n+1} \tag{8.1.8}$$

Substituting this expression into (8.1.3) results in an equation that may be solved for v_{n+1}:

$$\boxed{(M + \alpha \Delta t \, K)v_{n+1} = F_{n+1} - K\tilde{d}_{n+1}} \tag{8.1.9}$$

function of terms at $\underline{\underline{t_n}}$ which we know getting 8.1.8

put 8.1.5 in 8.1.4

$$d_{n+1} = d_n + \Delta t (1-\alpha) v_n + \Delta t \alpha v_{n+1}$$

rearrange

$$\Delta t (1-\alpha) v_n = d_{n+1} - d_n - \Delta t \alpha v_{n+1}$$

supstitute this into 8.1.7

$$\tilde{d}_{n+1} = d_n + d_{n+1} - d_n - \Delta t \alpha v_{n+1}$$

$$d_{n+1} = \tilde{d}_{n+1} + \Delta t \alpha v_{n+1} \Leftarrow 8.1.8$$

Observe that the terms on the right-hand side of (8.1.9) are known. Once v_{n+1} is determined, (8.1.8) serves to define d_{n+1}, and so on.

Remarks

1. In the case of $\alpha = 0$, the method is said to be ***explicit***. The attribute of explicit methods may be seen from (8.1.9) if M is assumed "lumped" (i.e., diagonal). In this case the solution may be advanced without the necessity of equation solving.

2. If $\alpha \neq 0$, the method is said to be ***implicit***. In these cases a system of equations, with coefficient matrix $(M + \alpha \Delta t\, K)$, needs to be solved at each step to advance the solution. As long as Δt is constant, only one factorization is required.

3. The right-hand-side vector is formed one element at a time. Recall that the definition of F is given in this way (see (7.1.27) through (7.1.29)). The product

$$K\tilde{d}_{n+1} = \mathop{\mathbf{A}}_{e=1}^{n_{el}} (k^e \tilde{d}^e_{n+1}) \tag{8.1.10}$$

where $\tilde{d}^e_{n+1}$ contains zeros in degrees of freedom that correspond to prescribed boundary conditions. The effect of the boundary-condition terms is already accounted for in the definition of F [see (7.1.29)]. This aspect of the implementation warrants further discussion. We will postpone the details until we consider a general implementation of a class of hyperbolic and hyperbolic-parabolic algorithms in Chapter 9. The general case can be specialized to the present circumstances.

4. It is useful to note that the implementation manifested by (8.1.9) can easily be generalized to so-called implicit-explicit methods, where part of K is treated implicitly and part explicitly. The *only* required change is to replace K on the *left-hand side* of (8.1.9) with the part to be treated implicitly, say K^I. This could be the assembly of a subset of elements, which leads to ***implicit-explicit element mesh partitions*** (see Hughes et al. [1–3]). It may be observed that the implementation is trivial. We shall put off a more detailed discussion until Chapter 9.

Implementation 2: d-form

Another possibility is to eliminate v_{n+1} from (8.1.3) via (8.1.8). Thus in place of (8.1.9) we have

$$\frac{1}{\alpha\Delta t}(M + \alpha \Delta t\, K)d_{n+1} = F_{n+1} + \frac{1}{\alpha \Delta t} M \tilde{d}_{n+1} \tag{8.1.11}$$

In this implementation, (8.1.11) is used to determine d_{n+1} and then v_{n+1} may be determined from (8.1.8), i.e.,

$$v_{n+1} = \frac{d_{n+1} - \tilde{d}_{n+1}}{\alpha \Delta t} \tag{8.1.12}$$

The advantage of this implementation occurs when M is diagonal. In this case the calculation of the right-hand side of (8.1.11) may be performed much more economically than the right-hand side of (8.1.9). The equation-solving burden is of course the same for (8.1.9) and (8.1.11).

Remark

Note that if $\alpha\Delta t \to \infty$ in (8.1.11), the equilibrium, or steady-state solution (i.e., one for which $\dot{d} = \mathbf{0}$) is approached, viz.,

$$Kd_{n+1} \to F_{n+1} \tag{8.1.13}$$

This fact can be exploited in a transient analysis computer program if the equilibrium solution is desired. Just select a value of $\alpha\Delta t$ large enough so that (8.1.13) holds to desired precision.

Exercise 1. Derive an implementation in which the v_n's are unnecessary. This results in a saving of computer storage. [*Answer*: $(M + \alpha\Delta t\, K)d_{n+1} = (M - (1 - \alpha)\Delta t\, K)d_n + \Delta t(\alpha F_{n+1} + (1 - \alpha)F_n)$.]

8.2 ANALYSIS OF THE GENERALIZED TRAPEZOIDAL METHOD

Must pay attention to what these analyses tell you

The primary requirement of the algorithms given in the last section is that they converge. We shall call an algorithm ***convergent*** if for t_n fixed and $\Delta t = t_n/n$, $d_n \to d(t_n)$ as $\Delta t \to 0$. To establish the convergence of an algorithm, two additional notions must be considered: ***stability*** and ***consistency***. We shall show later on that once stability and consistency are verified, convergence is automatic. In addition, we shall be concerned with the ***accuracy*** of an algorithm, i.e., the rate of convergence as $\Delta t \to 0$, and allied topics such as the behavior of the (spurious) higher modes of the semidiscrete system. There are several techniques that can be employed to study the characteristics of an algorithm. In the present context the most revealing approach appears to be the "modal approach" (sometimes called spectral, or Fourier, analysis) in which the problem is decomposed into n_{eq} uncoupled scalar equations. It can be rigorously established that the behavior of the entire coupled system reduces to consideration of the individual modal equations that comprise it. Our first step in analyzing the family of algorithms introduced in Sec. 8.1 will be to perform the reduction to single-degree-of-freedom (SDOF) form.

8.2.1 Modal Reduction to SDOF Form

The essential property used in reducing to SDOF form is the orthogonality of the eigenvectors of the associated eigenvalue problem. Recall that

$$(K - \lambda_l^h M)\psi_l = \mathbf{0}, \qquad l \in \{1, 2, \ldots, n_{eq}\} \tag{8.2.1}$$

where

$$0 \le \lambda_1^h \le \lambda_2^h \le \cdots \le \lambda_{n_{eq}}^h \tag{8.2.2}$$

and

$$\psi_l^T M\, \psi_m = \delta_{lm} \qquad \text{(orthonormality)} \tag{8.2.3}$$

18.

9 Algorithms for Hyperbolic and Parabolic-Hyperbolic Problems

ONE-STEP ALGORITHMS FOR THE SEMIDISCRETE EQUATION OF MOTION

9.1.1 The Newmark Method

Recall from Chapter 7 that the semidiscrete equation of motion is written as

$$M\ddot{d} + C\dot{d} + Kd = F \tag{9.1.1}$$

where M is the mass matrix, C is the viscous damping matrix, K is the stiffness matrix, F is the vector of applied forces, and d, $\dot{d}$, and $\ddot{d}$ are the displacement, velocity and acceleration vectors, respectively. We take M, C, and K to be symmetric; M is positive-definite, and C and K are positive-semidefinite.

The initial-value problem for (9.1.1) consists of finding a displacement, $d = d(t)$, satisfying (9.1.1) and the given initial data:

$$d(0) = d_0 \tag{9.1.2}$$

$$\dot{d}(0) = v_0 \tag{9.1.3}$$

Perhaps the most widely used family of direct methods for solving (9.1.1) to (9.1.3) is the *Newmark family* [1], which consists of the following equations:

$$Ma_{n+1} + Cv_{n+1} + Kd_{n+1} = F_{n+1} \tag{9.1.4}$$

$$d_{n+1} = d_n + \Delta t v_n + \frac{\Delta t^2}{2}[(1 - 2\beta)a_n + 2\beta a_{n+1}] \tag{9.1.5}$$

$$\boldsymbol{v}_{n+1} = \boldsymbol{v}_n + \Delta t[(1-\gamma)\boldsymbol{a}_n + \gamma\boldsymbol{a}_{n+1}] \tag{9.1.6}$$

where $\boldsymbol{d}_n$, $\boldsymbol{v}_n$, and $\boldsymbol{a}_n$ are the approximations of $\boldsymbol{d}(t_n)$, $\dot{\boldsymbol{d}}(t_n)$, and $\ddot{\boldsymbol{d}}(t_n)$, respectively. Equation (9.1.4) is simply the equation of motion in terms of the approximate solution, and (9.1.5) and (9.1.6) are finite difference formulas describing the evolution of the approximate solution. The parameters β and γ determine the stability and accuracy characteristics of the algorithm under consideration. Equations (9.1.4) to (9.1.6) may be thought of as three equations for determining the three unknowns $\boldsymbol{d}_{n+1}$, $\boldsymbol{v}_{n+1}$, and $\boldsymbol{a}_{n+1}$, it being assumed that $\boldsymbol{d}_n$, $\boldsymbol{v}_n$, and $\boldsymbol{a}_n$ are known from the previous step's calculations. The Newmark family contains as special cases many well-known and widely used methods.

Implementation: a-form

There are several possible implementations. We will sketch one, but we leave further details until Sec. 9.4, which deals with operator and mesh partitions. The results in Sec. 9.4 include the Newmark method as a special case. Define ***predictors***:

$$\tilde{\boldsymbol{d}}_{n+1} = \boldsymbol{d}_n + \Delta t\boldsymbol{v}_n + \frac{\Delta t^2}{2}(1-2\beta)\boldsymbol{a}_n \tag{9.1.7}$$

$$\tilde{\boldsymbol{v}}_{n+1} = \boldsymbol{v}_n + (1-\gamma)\Delta t\boldsymbol{a}_n \tag{9.1.8}$$

Equations (9.1.5) and (9.1.6) may then be written as

$$\boldsymbol{d}_{n+1} = \tilde{\boldsymbol{d}}_{n+1} + \beta\Delta t^2\boldsymbol{a}_{n+1} \tag{9.1.9}$$

$$\boldsymbol{v}_{n+1} = \tilde{\boldsymbol{v}}_{n+1} + \gamma\Delta t\boldsymbol{a}_{n+1} \tag{9.1.10}$$

To start the process, $\boldsymbol{a}_0$ may be calculated from

$$\boldsymbol{M}\boldsymbol{a}_0 = \boldsymbol{F} - \boldsymbol{C}\boldsymbol{v}_0 - \boldsymbol{K}\boldsymbol{d}_0 \tag{9.1.11}$$

or specified directly. The recursion relation determines $\boldsymbol{a}_{n+1}$:

$$\boxed{(\boldsymbol{M} + \gamma\Delta t\boldsymbol{C} + \beta\Delta t^2\boldsymbol{K})\boldsymbol{a}_{n+1} = \boldsymbol{F}_{n+1} - \boldsymbol{C}\tilde{\boldsymbol{v}}_{n+1} - \boldsymbol{K}\tilde{\boldsymbol{d}}_{n+1}} \tag{9.1.12}$$

Equations (9.1.9) and (9.1.10) may then be used to calculate $\boldsymbol{d}_{n+1}$ and $\boldsymbol{v}_{n+1}$, respectively.

This form of implementation is convenient for generalization to algorithms that employ "mesh partitions" (see Sec. 9.4) but is not the most efficient implementation.

Past Newmark methods there are methods like Generalized α that introduces an additional parameter

Ref: J. Chung and G.M. Hulbert, "A Time Integration Algorithm for Structural Dynamics with Improved Numerical Dissipation: The Generalized - α Method", J. Applied Mechanics, Vol. 60, pp. 371-375, 1993

There are lots of other families of methods if you want to go to higher order convergence rates - Runge-Kutta Methods - have not be heavily used with finite elements - which have historically been lower order. May be more important with use of high order F.E. methods.