NONLINEAR FINITE ELEMENT ANALYSIS OF REINFORCED CONCRETE STRUCTURES SUBJECTED TO TRANSIENT THERMAL LOADS

Cheng En Zhou
Master of Applied Science, 2004
Department of Civil Engineering
University of Toronto

Abstract

The need to incorporate fire loads into reinforced concrete structural design has long been recognized, and the traditional design method for structural fire resistance has been widely practiced by engineers mainly because of its simplicity. To simulate the structure's response to thermal loads, this research develops and implements a 2D nonlinear finite element transient analysis for reinforced concrete structures subjected to high temperatures.

The proposed computational scheme takes into account time-varying thermal loads, heat-of-hydration effects, and temperature-dependent material properties. Algorithms for calculating the closed-form element stiffness for a quadrilateral element with a fully-populated material stiffness are also developed. Then, the capability of a 2D nonlinear finite element transient thermal analysis is implemented into program VecTor2©, a nonlinear analysis program for 2D reinforced concrete membranes.

The results obtained from four numerical tests indicate that the proposed computational scheme and the implemented codes are accurate and reliable.

Acknowledgements

The author gratefully acknowledges his advisor, Prof. Frank J. Vecchio, for his continuous and patient guidance, encouragement and supports, without which this work would not have been completed in this form.

Great appreciation is also given to Nina for her constant encouragement and extended to all colleagues in VecTor Room.

I would also like to thank Dr. Kruppa in Paris, Dr. Bentz in Toronto, Dr. Griffiths in Colorado, and Dr. Lee in Singapore for their interest, discussion, suggestion, and help on this research.

Finally, thanks are also due to the Department of Civil Engineering for the conducive environment.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Background

Fire is one of the extreme loadings that can act on reinforced concrete structures. The need to incorporate this extreme loading into structural design has long been recognized, and the traditional design method for structural fire resistance has been widely practiced by engineers mainly because of its simplicity. However, the investigation of the World Trade Centre disaster by the BPAT (Building Performance Assessment Team) indicated that the fire issues were most crucial [1] in the collapse of the twin towers. Other than that, reinforced concrete structures are commonly exposed to thermal loads as the result of the design function of the structure, ambient conditions, heat of hydration, or exposure to fire [2]. Therefore, the research on the advanced analysis and design of reinforced concrete structures subject to thermal loads has attracted much attention recently.

To simulate the structure's response to various thermal loads, numerical techniques are normally required because the experimentation is usually too involved and expensive and the governing Partially Differential Equations (PDEs) are too complex for analytical solutions to be obtained. The numerical technique which has achieved the greatest degree of popularity and success is the finite element method.

1.2 Aims and Objectives

To understand the response of structures to thermal loads, one will have to consider some consequent analysis stages. First of all, the thermal actions originate from increases in temperature and can be time-varying. They have to be differentiated from mechanical loads. Next, the temperature distribution has to be calculated according to the subjected thermal actions. Again, one will have to consider the time effect in the transient heat analysis. Once the temperature distribution along the structure is known, one can evaluate the mechanical behaviour of the heated structure. Notably, some mechanical and thermal properties of material are highly temperature dependent, and some effects, like spalling and creep, could be of much importance in the structural analysis. Finally, one can compare the fire resistance factors or fire safety indices to predict the response.

The main purpose of this research is to develop and implement a 2D nonlinear transient conduction FE analysis for reinforced concrete structures subjected to high temperature. Basically, it is a thermal analysis in which the time-varying thermal loads and temperature-dependent thermal properties are taken into account. Some subroutines will be developed and then further embedded into the existing main-program VecTor2, a nonlinear analysis program for 2D reinforced concrete plates. Taking advantage of VecTor2's built-in realistic constitutive models, mainly based on the Modified Compression Field Theory [3] and Distributed Stress Field Model [4], one can obtain the complete response of all members within the structure, including external restraint forces, internal stresses, cracking development, deflections etc. Although currently not included, models to reflect the effects of spalling and thermal creep might be incorporated into the existing procedure without many difficulties.

1.3 Scope and Organization

The focus of this research is in the following areas:

- A systematic investigation will be carried out to explore the potential of thermal analysis of concrete structures, including the studies on fundamental theories of the thermal conduction analysis.

- A reliable finite element computational scheme will be developed, which involves temperature-dependent thermal properties and time-varying thermal loads.

- Program codes will be developed and modified, including V2HEAT (for transient heat flow analysis), V2TRED (for material property temperature-dependence), and V2STIF (for stiffness matrix calculation).

- Some corroboration problems will be tested.

This thesis is presented in six chapters as illustrated in Fig. 1.1 below.

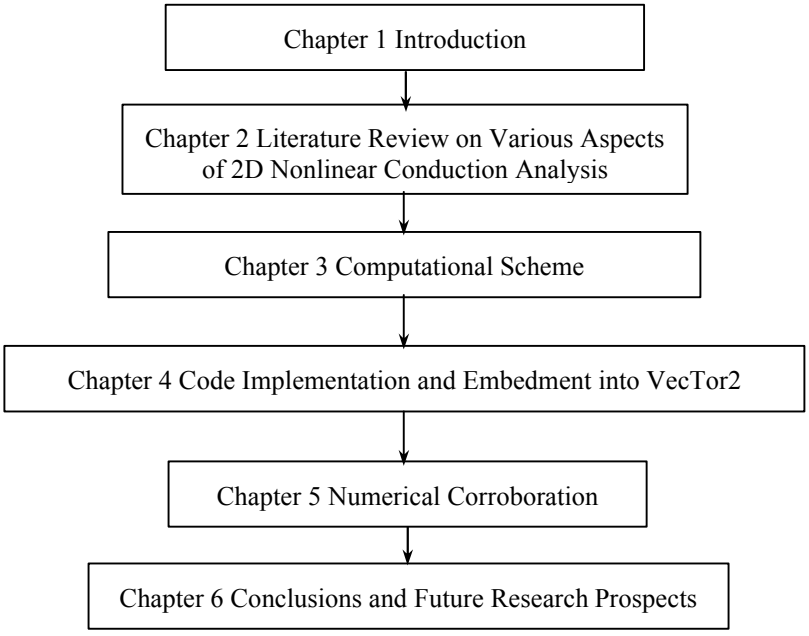


Figure 1.1 Structure of the Thesis

Chapter 2 Literature Review

Heat transfer is concerned with the physical processes underlying the transport of thermal energy due to a temperature difference or gradient. Conservation principles of mass, momentum, and energy always lead to differential equations (in a more general sense, integral equations). Because of those complex mathematical equations, numerical methods are usually preferred, instead of solving problems analytically.

Of the several means by which heat is transferred, conduction is probably the most widely understood and the most familiar. In the following sections, some characteristics of the numerical heat conduction analysis will be described and reviewed, namely, (i) general background on heat conduction analysis; (ii) underlying FE spatial approximation; (iii) underlying FD temporal descretization; (iv) temperature-dependent material properties; and (v) heat of hydration.

2.1 General Background on Heat Conduction Analysis

In the field of continuum mechanics, motions of a continuum can be described by either the Lagrangian approach (also called material description) or Eulerian technique (also known as spatial description). In the former description, individual particles (identified by material coordinates) are tracked with the passage of time. Alternatively, one can observe the changes in time at fixed positions in the space (identified by spatial coordinates), which is the Eulerian description. While the Lagrangian description is often used in solid mechanics, the Eulerian description dominates [5] in the field of fluid mechanics including

heat transfer. Consequently, the governing equations presented here will be based on the Eulerian description in which the current (heat) flow field is fixed at the reference coordinates rather than tracing the particles downstream.

2.1.1 Classification of Partial Differential Equations (PDEs)

The general form of the two-dimensional second-order PDEs can be expressed as:

$$A\frac{\partial^2 T}{\partial x^2} + B\frac{\partial^2 T}{\partial x \partial y} + C\frac{\partial^2 T}{\partial y^2} + D\frac{\partial T}{\partial x} + E\frac{\partial T}{\partial y} + FT + G = 0 \tag{2.1}$$

where the coefficients A, B, C, D, E, and F can be functions of both independent variables (coordinates x and y or time t) and dependent variables (temperature T). According to the physical meaning in the propagation of flow's disturbance and characteristics, Sneddon [6] classified Eq. (2.1) into three categories:

$$B^2 - 4AC \begin{cases} > 0 & \textit{Hyperbolic} \\ = 0 & \textit{Parabolic} \\ < 0 & \textit{Elliptic} \end{cases} \tag{2.2}$$

2.1.2 Governing Equations of Heat Conduction and Their Nature

Conduction problems are generally divided into two main categories: steady-state and transient (or unsteady-state) conduction. The former relates to the condition where the temperatures at all nodes are independent of time while the latter indicates the situation where energy storage occurs and the temperature distribution varies with time. Those two categories of problems are usually treated separately.

- Steady-State Conduction:

The governing equation in terms of Cartesian coordinates for an isotropic material in steady-state problems is of the form:

$$\nabla \cdot (k\nabla T) + Q = 0 \tag{2.3}$$

where, Q is volumetric heat resource and k is the material conductivity.

Two well-known equations of *elliptic* type (c.f. Eq. (2.2)) are:

Laplace equation: $$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \tag{2.4}$$

Poisson equation: $$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + G = 0 \tag{2.5}$$

Both equations above apply to the temperature distribution for constant-property conduction while Eq. (2.5) reflects distributed heat source present in the problem domains.

- Transient Conduction:

The governing equation for transient problems is given by:

$$\nabla \cdot (k\nabla T) + Q - rc\frac{\partial T}{\partial t} = 0 \tag{2.6}$$

where the time t is involved as an independent variable. r and c are the material's density and specific heat respectively.

Under the assumption of *constant* thermal properties, Eq. (2.6) leads to:

$$\frac{\partial T}{\partial t} = a\left(\frac{\partial^2 q}{\partial x^2} + \frac{\partial^2 q}{\partial y^2}\right) + G \tag{2.7}$$

where a is so-called material diffusivity.

By comparing Eq. (2.7) and the generalized form in Eq. (2.2), one can see that the transient two-dimensional conduction problem possesses a *parabolic* nature with respect to its time dependence and an *elliptic* behaviour with respect to the spatial coordinates.

2.1.3 Initial and Boundary Conditions of Heat Conduction

Correct application of boundary conditions is essential to the convergence and accuracy of numerical solutions. Generally, the number of boundary conditions required is determined by the order of the highest derivatives appearing in each independent variable in the governing PDEs. For example, a transient process governed by a first derivative in time (e.g. Eq. (2.6)) will require one initial condition in order to carry out the time integration. Also, two spatial boundary conditions are needed for each coordinate in which a second derivative appears.

The initial temperature field can be specified as:

$$T(x, y, t = 0) = T_0(x, y) \quad \text{in } \Omega \tag{2.8}$$

Spatial boundary conditions are normally of the following types:

- *Dirichlet* (or essential) boundary conditions: $T = \overline{T}(x, y)$ on Γ_T (2.9)
- *Neumann* (or natural) boundary conditions: $-k\frac{\partial T}{\partial n} = \overline{q}_n$ on Γ_q (2.10)

- *Cauchy* (or mixed) boundary conditions: $a(x,y)T + b(x,y)\dfrac{\partial T}{\partial n} = f(x,y)$ (2.11)

where, $\dfrac{\partial T}{\partial n} = \vec{n}\cdot(\nabla T) = n_x \dfrac{\partial T}{\partial x} + n_y \dfrac{\partial T}{\partial y}$ with unit vector $\vec{n}$ normal to the boundary Γ and its direction-cosine components expressed by n_x and n_y. $\overline{T}$ and $\overline{q}$ are the prescribed values of temperature and heat flux on the corresponding boundaries Γ_T and Γ_q.

2.2 Underlying Finite Element Spatial Approximation

A finite element (FE) method is a mathematical procedure for satisfying a partial differential equation in an average sense over a finite element. Various methods exist but all of them require that an integral representation of the PDE be constructed. The FE method is attractive because of its integral formulation and the use of *unstructured* grids, which are preferred [7] for flows. In the following sections, the construction of such integral formulations and commonly used 2D finite elements will be briefly described.

2.2.1 Integral Representations

The differential and integral forms of the governing equations provide alternate starting points for a numerical solution. The differential equations apply locally in an appropriate time-space continuum while global forms may be obtained by integrating the differential equations over a suitable region of time and space. Within the FE methods, it is necessary to transform the governing equation from its differential form into an equivalent integral one. This can be accomplished in two different ways: by classical variational principle or by the more general weighted-Galerkin residual method.

Consider the governing equations for an isotropic material in transient problems:

$$A(T) = \nabla \cdot (k\nabla T) + Q - rc\frac{\partial T}{\partial t} = 0 \quad \text{in}\,\Omega \tag{2.12.a}$$

with the boundary conditions defined as:

$$B(T) = T - \overline{T} = 0 \quad \text{on}\,\Gamma_T \tag{2.12.b}$$

$$\text{or } = k\frac{\partial T}{\partial n} + \overline{q} = 0 \quad \text{on}\,\Gamma_q \tag{2.12.c}$$

After dividing the continuum into a finite number of elements, the behavior of which is specified by a finite number of nodal parameters **a**, the FE method approximates the solution in the form as:

$$T \approx T^h = \sum_i N_i a_i = \mathbf{Na} \tag{2.13}$$

where **N** are *shape functions* prescribed in terms of independent variables (such as the coordinates) and usually defined locally for elements.

Because of the virtual work principle and the property of definite integral requiring that the total be the sum of the parts, that is,

$$\int_\Omega ( )d\Omega = \sum \int_{\Omega^e} ( )d\Omega \tag{2.14.a}$$

and

$$\int_\mathrm{G} ( )d\mathrm{G} = \sum \int_{\mathrm{G}^e} ( )d\mathrm{G}, \tag{2.14.b}$$

It can be shown that the properties of such discrete systems would be recovered if the local approximation in Eq. (2.13) can be cast in an overall integral form:

$$\int_{\Omega} A(T^h)d\Omega + \int_{\Gamma} B(T^h)d\mathrm{G} = \sum_e \int_{\Omega^e} A(T^h)d\Omega^e + \sum_e \int_{\Gamma^e} B(T^h)d\Gamma^e \tag{2.15}$$

When developing the weighted-Galerkin residual formulation given above, one important theorem known as *Gauss's theorem* is often used:

$$\int_v \nabla \cdot \vec{r}dv = \int_s \vec{n} \cdot \vec{r}ds \tag{2.16}$$

For example, in integrating Eq. (2.7) over a non-deforming two-dimensional region, by using the above *Gauss*'s theorem one can have (note that exchangeable consequence between differential and integral operations):

$$\frac{\partial}{\partial t}\int_{\Omega} Td\Omega = a\int_{\Gamma} \frac{\partial T}{\partial n}d\Gamma + \int_{\Omega} Gd\Omega \tag{2.17}$$

where the a is taken as *constant*. Physically, the left side of this expression represents the increase rate of the area integral of *T*, while the terms on the right side denote, respectively, the net increase of *T* by diffusion and volumetric sources. Eq. (2.17) may be integrated in time, and then the need for appropriate initial and boundary data (conditions) is apparent.

Thus, FE methods can be viewed as satisfying a differential equation in some average sense over a region of space, and thus as providing regional integral approximation to the original differential equations. Finite Difference (FD) methods, on the other hand, are usually regarded as local point-wise approximation methods based on differential equations.

2.2.2 Commonly Used 2D Finite Elements and Their Shape Functions

In two dimensions, the region is usually discretized by discrete triangular, rectangular or even more general quadrilateral elements. Since the choice of a particular coordinate system influences the amount of algebra required in the formulation, it needs to be chosen wisely [8]. Consider the following:

- Mesh information in this analysis (sub-program) is directly obtained from the remaining main-program VecTor2. As a result, nodal coordinates, element connections as well as their numberings are inherited from the existing code;
- The construction of shape functions that satisfy consistency requirements for linear elements with straight-lined boundaries becomes straightforward. As a result, the concept of *isoparametric* elements is circumvented;
- Integrals that appear in the expressions of the element coefficient matrices and consistent nodal force vector can be carried out in closed form. As a result, numerical quadrature is avoided in the computational scheme.

Given these facts, the standard rectangular Cartesian coordinates are chosen for both the employed bilinear quadrilateral and linear triangular elements when constructing the element shape functions as below. Since no transformations between the coordinate systems are involved, the element matrices can be derived in such a way that is quite different [9] from that commonly reported for *isoparametric* elements.

Despite being included in the more general case of four-node quadrilateral element, the formulation for the rectangular element will be reported separately since they are advantageous in situations where it can be used.

2.2.2.1 Linear Triangular Element

For the typical linear triangular element shown in Fig. 2.1, it is common practice to resort to an *area coordinate* system [9] when formulating the shape function since the shape functions are simply the area coordinates themselves. That is,

$$N_i = x_i = \frac{A_i}{A} \text{ (with counter-clockwise arrangement, i.e. } i = 1,2,3) \tag{2.18}$$

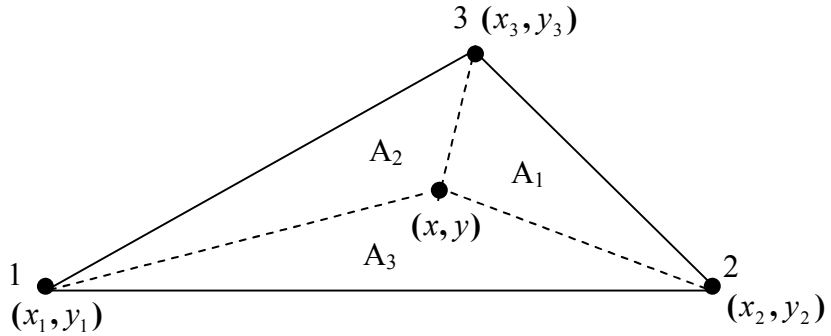


Figure 2.1 An schematic triangular element

Note that the area coordinates $(x_i, i = 1,2,3)$ are not independent because their sum is equal unity. Substituting the area coordinate into Eq. (2.18), one can easily derive the shape functions in Cartesian coordinates as follows:

$$N_i(x,y) = \frac{A_i}{A} = \frac{\det \begin{vmatrix} 1 & x & y \\ 1 & x_j & y_j \\ 1 & x_m & y_m \end{vmatrix}}{\det \begin{vmatrix} 1 & x_i & y_i \\ 1 & x_j & y_j \\ 1 & x_m & y_m \end{vmatrix}} \tag{2.19}$$

2.2.2.2 Rectangular Element

To construct shape functions as well as element coefficient matrices for a rectangular element as shown in Fig. 2.2, one might be tempted to use a polynomial expansion in terms of the standard Cartesian coordinates. A Lagrange polynomial is frequently chosen since the desired interpolation functions can be constructed simply from a tensor product of the one-dimensional counterparts for the *x* and *y* directions respectively.

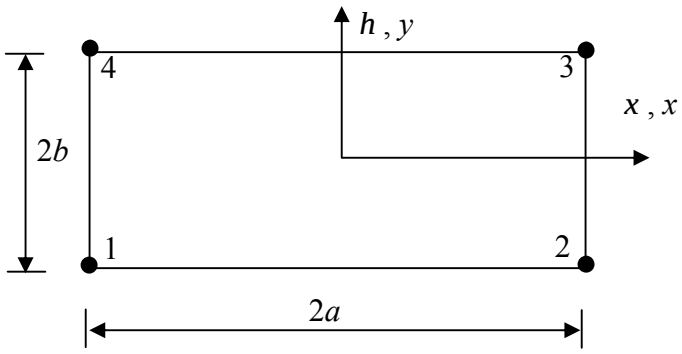


Figure 2.2 A rectangular element in Cartesian coordinates

Consider the linear variation of the nodal displacement vector **u** in two dimensions:

$$\mathbf{u}^{(e)} = N_i u_i \ (i = 1,2,3,4) \tag{2.20}$$

with

$$N_1 = L_1^{(x)} L_1^{(y)},\ N_2 = L_2^{(x)} L_1^{(y)},\ N_3 = L_2^{(x)} L_2^{(y)},\ N_4 = L_1^{(x)} L_2^{(y)} \tag{2.21}$$

where,

$$L_1^{(x)} = \frac{1}{2}(1-x),\ L_2^{(x)} = \frac{1}{2}(1+x),\ L_1^{(y)} = \frac{1}{2}(1-h),\ L_2^{(y)} = \frac{1}{2}(1+h) \tag{2.22}$$

and

$$x = x/a\,,\ h = y/b \tag{2.23}$$

2.2.2.3 Four-Node Quadrilateral Element

The natural coordinates (also called quadrilateral coordinates) for a quadrilateral element are x and h, which are illustrated in Fig. 2.3 for a straight-sided quadrilateral.

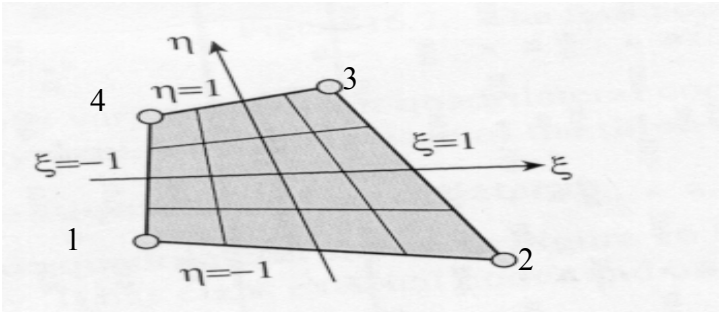


Figure 2.3 A bilinear quadrilateral element

The shape functions for the above 4-node isoparametric quadrilateral element are:

$$N_i(x,h) = \frac{1}{4}(1+xx_i)(1+hh_i) \tag{2.24}$$

where (x_i, h_i) are the nodal coordinates in the natural pattern ($(x,h) \in [-1,1]$). Note that these functions in Eq. (2.24) do vary linearly on quadrilateral lines, but are *not* linear polynomials due to the extra term xh, which is not required for the purpose of linear-complete polynomials. This traditional natural-coordinate-based isoparametric shape function will *only* be used when developing the closed-form element matrix formulation, in which the entries of matrices are eventually derived in terms of nodal rectangular Cartesian coordinates.

2.3 Underlying Finite Difference Temporal Discretization

In transient problems, it will be shown that the resultant equation is a set of first-order ordinary differential equations with respect to the independent variable time *t*. Consequently, the solutions must proceed with increasing time until the results are obtained over a required time level or until a particular temperature level (such as the steady state) is attained.

The simplest and most common procedure for such a transient analysis is to use a two-level finite difference formulation in time, which interpolates time between two successive levels n and $n+1$. That is,

$$\frac{\partial T}{\partial t} = \frac{T_{n+1} - T_n}{\Delta t} \tag{2.25}$$

As it will be shown in Chapter 3, in the case of temperature-dependent properties, one must decide at what time level to evaluate the temperature terms appearing in the coefficient matrices in the resultant equations. Similarly, one can use two-point (linear) interpolation formula as follows:

$$T_{n+r} = (1-r)T_n + rT_{n+1} \tag{2.26}$$

By changing the value of *r* from 0 to 1 in Eq. (2.26), different classical methods in the literature can be identified as follows:

- $r = 0$: the Euler method

The Euler method is a forward finite difference technique. The advancement requires no iteration due to its explicit nature, but the time step is restricted by stability considerations and may be further restricted by nonlinear problems.

- $r = 1$: the Laasonen method

This is a backward finite difference technique with a fully implicit nature.

- $r = 1/2$: the Crank-Nicholson method

The Crank-Nicholson is a center finite difference technique. This is a common choice based on accuracy considerations [10], with the time discretization being second-order correct.

- $r = 2/3$: the Galerkin method

Instead of the finite difference technique, if one employs a Galerkin-weighted residual method with linear 1D interpolation function in time dimension, the value of r in Eq. (2.26) could be proven [11] to be $2/3$. While the final formulation looks similar to those from FD approximations, the Galerkin scheme exhibits considerable computational advantages. It is unconditionally stable and gives less oscillatory errors than the most accurate Crank-Nicholson scheme.

2.4 Temperature-Dependent Material Properties

While concrete is generally a non-homogeneous, anisotropic medium composed of particles of aggregate held together by hydrated cement paste, it can be treated as a homogeneous isotropic material in heat analysis. Despite all that, the temperature dependence of thermal properties has an especially great effect on heat transfer analysis. Moreover, the temperature-dependence of the mechanical properties will significantly affect the follow-up stress and deformation analyses.

2.4.1 Thermal Properties

From the general transient governing equation (i.e. Eq. (2.6)), one can see that, in addition to physical property density r, the two thermal properties involved in the heat analysis are:

- Thermal conductivity *k*: heat flux transmitted through a unit area of a material under a unit temperature gradient. This measures the ability of the material to conduct heat.

- Specific heat *c*: quantity of heat needed to raise the temperature of a unit mass of a material by one degree. This is a measure of the heat capacity of a material.

The temperature-dependent thermal and physical properties make heat analysis nonlinear since the coefficient matrices in the final resultant equation are not constant but dependent on the temperature, which in turn is the unknown to be solved.

While simple problems might be tackled by using the Kirchoff transformation technique [12], most resultant computational schemes will lead to a tremendous number of computations and consequently are quite inefficient [13]. Furthermore, the use of the

Kirchoff transformation requires reverting back to the original physical temperature after solving the equations and this involves the integration of the transformation equations.

Therefore, it is more convenient and simpler to consider the case of variable thermal and physical properties directly in the construction of relevant governing equation and to solve the equation in an iterative manner. Whether properties are functions of temperature or in the form of tiered data companied by interpolation, they give rise to a nonlinear algebraic equations system and the associated complexity is solving the equations. Fortunately, such difficulties can be reasonably bypassed by assigning an average value of each property within each finite element at the current iteration step to simplify the formulation. The detailed derivation will be described in Chapter 3. Therefore, the only task remaining here is to establish a database from which one can read the values of those temperature-dependent properties at different temperature ranges.

Since thermal properties at high temperature are quite difficult to obtain and there are very few data available in the literature (e.g. reference [14-15]), their variation with temperature employed in this proposed implementation scheme are based on the Eurocode and are briefly given below:

$$r(T) = \begin{cases} r(20^\circ C) & T \le 115^\circ C \\ r(20^\circ C)(1 - 0.02(T - 115)/85) & 115^\circ C < T \le 200^\circ C \\ r(20^\circ C)(0.98 - 0.03(T - 200)/200) & 200^\circ C < T \le 400^\circ C \\ r(20^\circ C)(0.95 - 0.07(T - 400)/800) & 400^\circ C < T \le 1200^\circ C \end{cases} \tag{2.27}$$

where $r(T)$ is the density at temperature $T(^\circ C)$; and

$$c(T)=\begin{cases}900 \quad (J/kg\,K) & T\le 100^{\circ}C\\ 900+(T-100) \quad (J/kg\,K) & 100^{\circ}C<T\le 200^{\circ}C\\ 1000+(T-200)/2 \quad (J/kg\,K) & 200^{\circ}C<T\le 400^{\circ}C\\ 1100 \quad (J/kg\,K) & 400^{\circ}C<T\le 1200^{\circ}C\end{cases} \tag{2.28}$$

where $c(T)$ is the specific heat at temperature $T(^{\circ}C)$.

Concerning the conductivity *k*, the initial value at a reference temperature ($20^{\circ}C$) can be used in the interpolation between the upper and lower limits given by Eurocode:

$$k^{upper}=2.0-0.2451(T/100)+0.0107(T/100)^2 \ (W/m\,K) \tag{2.29.a}$$

$$k^{lower}=1.36-0.136(T/100)+0.0057(T/100)^2 \ (W/m\,K) \tag{2.29.b}$$

Also, the experimental data provided by Shin et al. [14], given in Table 2.1, can be used as a back-up choice in the relative codes.

Temperature ($^{\circ}C$)	Density r ($kg\,m^{-3}$)	Conductivity k ($W/m\,K$)	Diffusivity a ($\times 10^{-6}\,m^2 s^{-1}$)	Specific heat c (J/kgK)
20	2252.43	2.194	0.8824	1104
500	2104.97	1.283	0.4505	1354
700	2077.71	1.136	0.4031	1357
900	2057.44	1.027	0.4170	1199
Correlative function of T	$r=0.0001896T^2-0.3980T+2259.62$ $k=1.3647E-6T^2-0.002569T+2.2427$ $a=(9.1639E-7T^2-0.001370T+0.9091)\times 10^{-6}$ $c=k/ar$			

Table 2.1 Thermal Properties Given by Shin et al. [14]

Variation of these modification factors given above are plotted in Fig. 2.4.

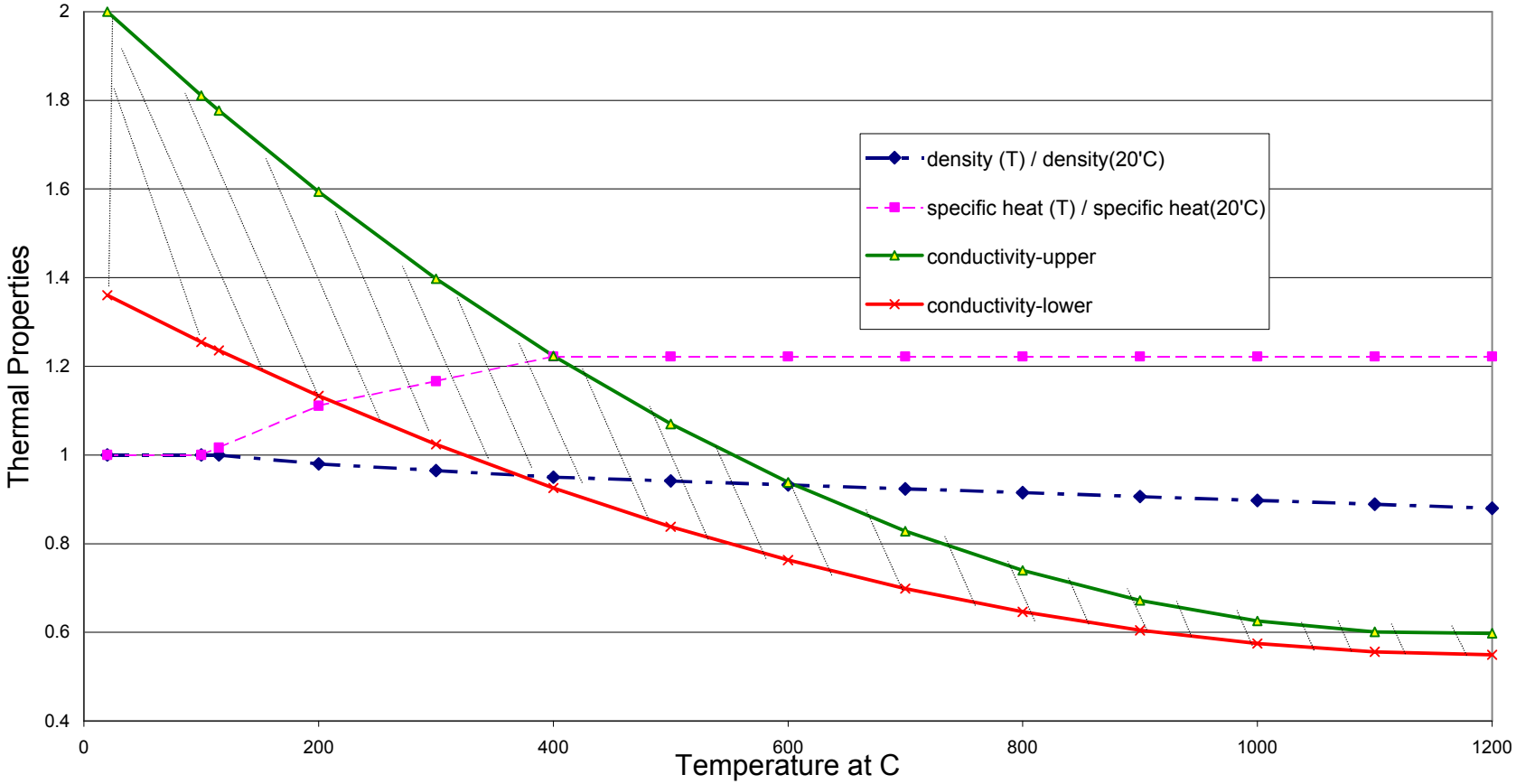


Figure 2.4 Thermal factors at high temperatures

2.4.2 Mechanical Properties

Once the transient temperature distribution is obtained, the structural response can be determined in which the material mechanical properties will dominate. Both strength and stiffness deteriorate significantly under evaluated temperatures. In addition, continuity thermal stresses are induced in indeterminate structures due to thermal expansion and are heavily dependent on structural effective stiffness [16].

The mechanisms governing the chemical reactions and physical changes inside reinforced concrete, and how they affect the mechanical properties, are complicated. Little experimental data is available (e.g. reference [17-18]) and, surprisingly, they are quite different from each other. Therefore, the values for normal-weight concrete and hot-rolled steel in Eurocode are employed in the calculation of the reduction factors. A short description of these data, and the corresponding calculated factors in VecTor2, are given below.

Some characteristics of concrete and reinforcing steel at evaluated temperatures are shown in Table 2.2. The reduction factors for those properties are directly given with the exception of the value of $e_c^{'}$ which includes thermal strains.

If the tensile strength of concrete is to be taken into account, in the absence of more accurate information, the following equation can be used in the calculation of this reduction factor, according to Eurocode.

$$f(T) = \begin{cases} 1.0 & T \le 100^{\circ}C \\ 1.0 - 1.0(T-100)/500 & 100^{\circ}C < T \le 600^{\circ}C \\ 0.0 & 600^{\circ}C < T \le 1200^{\circ}C \end{cases} \tag{2.30}$$

temperature	concrete				steel		
	siliceous aggregates		carbonate aggregates		f_u / f_y^{20}	f_u / f_u^{20}	E_s / E_s^{20}
	$f_c^{'} / f_{c,20}^{'}$	$e_c^{'}$	$f_c^{'} / f_{c,20}^{'}$	$e_c^{'}$			
20	1.00	0.0025	1.00	0.0025	1.0	1.0	1.0
100	1.00	0.0040	1.00	0.0040	1.0	1.0	1.0
200	0.95	0.0055	0.97	0.0055	1.0	0.81	0.90
300	0.85	0.0070	0.91	0.0070	1.0	0.61	0.80
400	0.75	0.0100	0.85	0.0100	1.0	0.42	0.70
500	0.60	0.0150	0.74	0.0150	0.78	0.36	0.60
600	0.45	0.0250	0.60	0.0250	0.47	0.18	0.21
700	0.30	0.0250	0.43	0.0250	0.23	0.07	0.13
800	0.15	0.0250	0.27	0.0250	0.11	0.05	0.09
900	0.08	0.0250	0.15	0.0250	0.06	0.04	0.07
1000	0.04	0.0250	0.06	0.0250	0.04	0.02	0.04
1100	0.01	0.0250	0.02	0.0250	0.02	0.01	0.02
1200	0.00	-	0.00	-	0.00	0.00	0.00

Table 2.2 Some material properties of concrete and reinforcing steel

In the Eurocode, the thermal expansion coefficients have the variation as shown in Fig. 2.5.

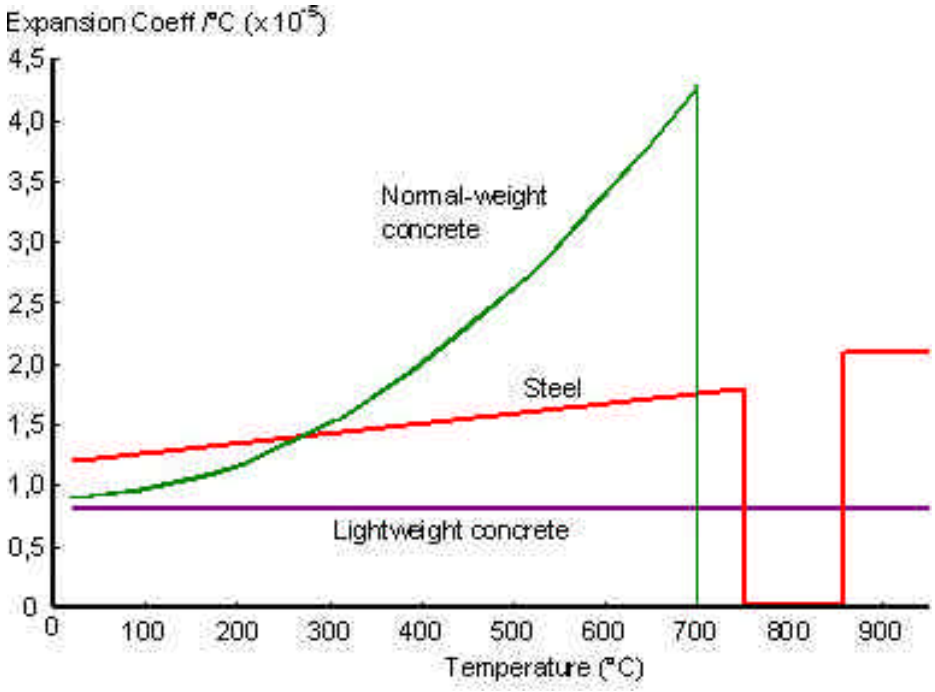


Figure 2.5 Thermal coefficients of concrete and steel

It can be seen that for both concrete and steel, thermal expansion ceases altogether at some levels of temperature. Instead of giving explicit thermal expansion coefficients, the Eurocode provides thermal strains within various temperature ranges directly as thermal elongation is believed to develop *progressively*.

- Concrete with siliceous aggregate:

$$e(T)=\begin{cases}-1.8\times10^{-4}+9\times10^{-6}T+2.3\times10^{-11}T^3 & T\le 700^\circ C\\ 14\times10^{-3} & 700^\circ C<T\le 1200^\circ C\end{cases} \tag{2.31.a}$$

- Concrete with carbonate aggregate:

$$e(T)=\begin{cases}-1.2\times10^{-4}+6\times10^{-6}T+1.4\times10^{-11}T^3 & T\le 805^\circ C\\ 12\times10^{-3} & 805^\circ C<T\le 1200^\circ C\end{cases} \tag{2.31.b}$$

- Reinforcing steel:

$$e(T)=\begin{cases}-2.416\times10^{-4}+1.2\times10^{-5}T+0.4\times10^{-8}T^3 & T\le 750^\circ C\\ 11\times10^{-3} & 700^\circ C<T\le 860^\circ C\\ -6.2\times10^{-3}+2\times10^{-5}T & 860^\circ C<T\le 1200^\circ C\end{cases} \tag{2.32}$$

Note that the above thermal strains are relative to the length at $20^\circ C$. Thus, with a reference at $20^\circ C$ the reduction (increasing in this case) factor could be calculated at:

$$f(T)=\frac{a(T)}{a(20)}=\frac{e(T)/\Delta T}{e(20)_{,T}}=\frac{e(T)/(T-20)}{e(20)_{,T}} \tag{2.33}$$

Variation of these modification factors given above are plotted in Fig. 2.6 and 2.7 for various concrete and steel, respectively.

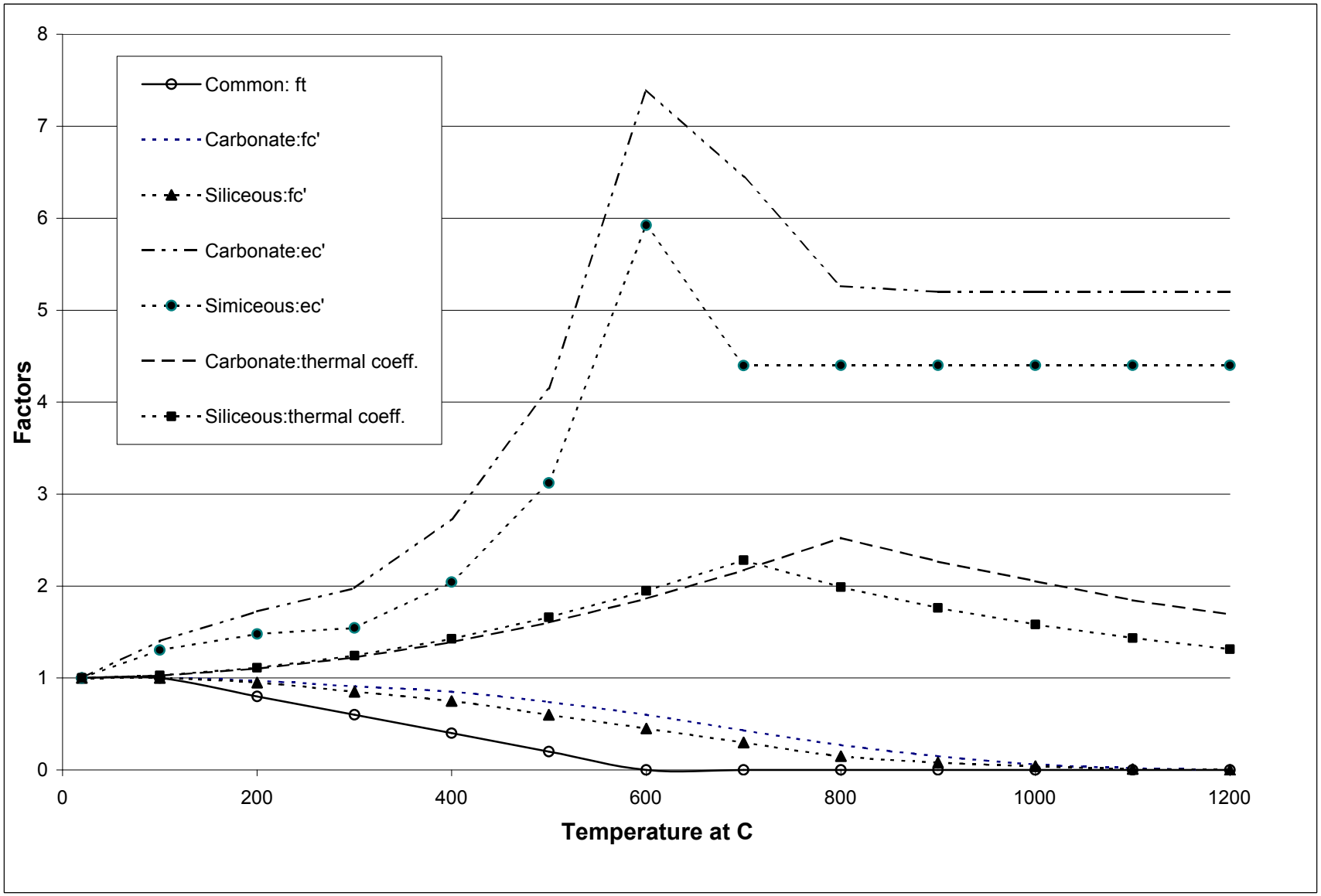


Figure 2.6 Mechanical modification factors for various concrete

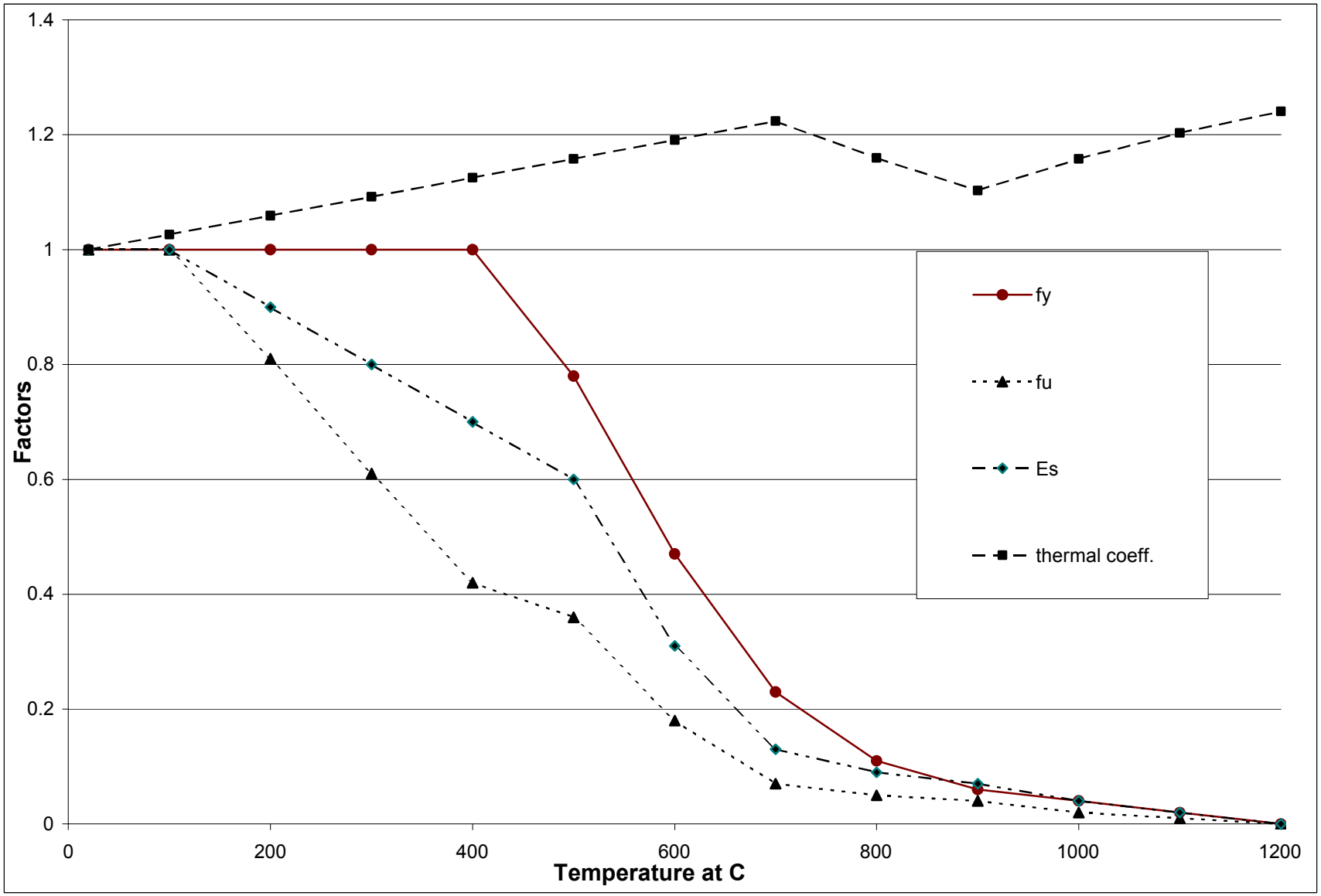


Figure 2.7 Mechanical modification factors for steel

2.5 Heat-of-Hydration Model

The hydration of cement within concrete is an exothermic process. Thermal expansion, and its induced effects such as thermal cracking, may reach a significant level in early age concrete, particularly in massive structures.

The micro-structural development occurring during the hydration of cement is complicated. To this date, precise knowledge of the mechanisms and kinetics of the hydration reactions between cement and water requires further investigation [19]. In addition, various factors, including curing conditions (temperature and age), type of cement, mix proportion of the concrete (such as water/cement ratio), as well as environmental interactions, can influence the rate and the total heat of hydration within concrete structures. Thus, it seems appropriate to treat the heat generated per unit volume during hydration as a material property [20]. At the same time, it has become clear that age-dependent material properties must be *uniquely* [21] tied to the *degree of hydration*, in turn relating to *maturity value* in the field of the *maturity method* [22].

Determination of concrete maturity values requires the knowledge of its time-temperature history. The maturity method is viewed as a successful technique which takes into account the varying effects of concrete temperature and curing time on concrete strength development. While the maturity method has been used conventionally to predict concrete strength gain during curing, its application to concrete technology can go far beyond that [23]. In this sense, it can be applied to any concrete property related to the extent of cement hydration since the maturity method is based on cement hydration kinetics.

In the application of the maturity method, maturity models (or functions) are used to convert the time-temperature curing history of concrete into maturity values. Based on accuracy and wider range of temperature conditions, the best known model is the one proposed by Freiesleben-Hanson and Pedersen [24]. Unlike the early models proposed by Saul [25] and Rastrup [26], this model is based on the Arrhenius equation, which describes the effect of temperature on the rate of chemical reactions, and is of the form:

$$M(t,T) = \int_0^t k \exp\left(\frac{-E_a}{RT_k}\right) dt \tag{2.34}$$

where,

R is the universal gas constant being equal to $8.314 J/(mol \cdot K)$.

T_k is the actual temperature of concrete in degree Kelvin.

E_a is conventionally an apparent activation energy in kJ/mol. It is recently reported [21-25] that E_a is not a true activation energy but, rather, provides a temperature sensitivity factor for the property interested. While the CEB-FIP code uses an average value and ASTM recommends a range, the method accepted by many researchers is of the form:

$$E(T_c) = \begin{cases} 33.5 & T_c \geq 20°C \\ 33.5 + 1.47(20 - T_c) & T_c < 20°C \end{cases} \tag{2.35}$$

k is typically a temperature-dependent rate constant for compression strength development, but in this case, for degree of hydration development. By using a parabolic-dispersion model (the one proposed by Knudsen [27] is commonly employed) to fit the experimental

data, the curve representing degree of hydration versus time can be obtained. It is important to point out that the k value is also property-dependent while it is classically related the compression strength development.

Since the rate constant is affected by temperature, an equivalent quantity called the equivalent time t_e is introduced and defined as follows:

$$t_e = \sum_0^t \exp\left[\frac{E_a}{R}\left(\frac{1}{T_r + 273} - \frac{1}{T + 273}\right)\right]\Delta t \tag{2.36}$$

In this way, time values at which the maturity values, in turn the degree of hydration, have been measured at other temperature can be converted to equivalent times at the reference temperature, in hopes of obtaining a single curve for degree of hydration versus *equivalent* time.

Now one can establish the model to simulate the relationship between the rate of heat of hydration and degree of hydration. There are some models developed in the past decade [28-29]. While numerous efforts have been made to include as many realistic features into those models as possible, only time will tell if the underlying assumption are indeed reasonable. The volumetric heat of hydration in the model, which is based on the work by Wang and Dilger [28] for ordinary Portland cement, is of the form:

$$Q = \begin{cases} 0.5 + 0.54 t_e^{\,0.5} & t_e \le 10.0 \\ 2.2 \exp[-0.0286(t_e - 10.0)] & t_e > 10.0 \end{cases} \tag{2.37}$$

where t_e is the equivalent maturity time of concrete in hours.

While it looks like one can proceed with heat analysis once the above Q is determined, the implementation of heat analysis considering the heat of hydration remains unsolved mainly due to following three facts:

1. Some influencing factors, like water/cement ration and cement content, are highly situation dependent. An accurate model requires a complete data base for which experimental works will be involved. This project mainly focuses on analytical computation.

2. The environmental interactions during heat analysis, technically termed as boundary conditions in FE analysis, will play an important role. That energy transferred between the boundary and the environment is mainly due to convection, thermal irradiation as well as solar radiation. However, only heat conduction is involved in this work.

3. As mentioned previously, the material properties of concrete at early ages will be *highly* age-dependent. Thus, the absence of thermal properties for heat analysis and mechanical properties for the follow-up analyses, such as stress and crack evaluation, makes this temperature calculation (if done) less reliable and useful.

Although no implementation of heat of hydration will be realized in this project, the formulation provided in the proposed computational scheme (described in Chapter 3) does include this aspect for the future possible development.

Chapter 3 Computational Scheme

Although the actual derivation is based on a Taylor expansion, Finite Difference (FD) approximation is based on local, discontinuous shape functions [7] with collocation weighting applied from the weighted-residual method viewpoint. Its conceptual simplicity and ease of implementation entail the loss of accuracy. Moreover, its dependency on a *structured* grid significantly restricts the application range since one-to-one mapping between the physical domain and computational domain is the key feature of such a grid.

The reasons for the success of the FE method are well known [30]: local characteristic of approximation, flexible ability of simulation of complex geometrical domains, and existence of a large set of approximation schemes adapted to various problems but embedded in a unified formulation. The FE method requires only an *unstructured* grid, in which nodes and grid cells (elements) are quasi-random ordered. That is, unlike the structured grid, neighbouring cells or nodes cannot be directly identified by their indices. In addition to its flexibility, the unstructured grid brings in the much easier adaptation of mesh (mesh refinement or mesh regeneration) to the solution domain. Therefore, the FE method is chosen in the current computational scheme.

3.1 Governing Equations with Appropriate Boundary Conditions

Due to the additional dimension introduced by the time variable, the numerical procedures for transient differential equations are quite different from those for steady-state problems.

Therefore, they are usually treated separately. For the purpose of description, some equations in Chapter 2 will be repeated here.

The governing equation in terms of Cartesian coordinates for an isotropic material in steady-state problems is of the form:

$$\nabla \cdot (k \nabla T) + Q = 0 \tag{2.3}$$

while the one for transient problems is given by:

$$\nabla \cdot (k \nabla T) + Q - rc \frac{\partial T}{\partial t} = 0 \tag{2.6}$$

To solve these equations, one has to specify the initial temperature distribution (for transient problems only) and the boundary conditions which might be time dependent and spatially varying.

The initial condition determines the state of the temperature field at time $t = 0$ and can be expressed as:

$$T(x, y, t = 0) = T_0(x, y) \quad \text{in } \Omega \tag{2.8}$$

Spatial boundary conditions are normally of the following types:

- *Dirichlet* (or essential) boundary conditions: $T = \overline{T}(x, y) \quad \text{on } \Gamma_T$ (2.9)

- *Neumann* (or natural) boundary conditions: $-k \frac{\partial T}{\partial n} = \overline{q}_n \quad \text{on } \Gamma_q$ (2.10)

- *Cauchy* (or mixed) boundary conditions: $a(x,y)T + b(x,y)\dfrac{\partial T}{\partial n} = f(x,y)$ (2.11)

The Dirichlet boundary conditions can be exactly treated by modifying the coefficient matrix in the global finite element equations, reflecting the specified value of nodal solutions. The specification of Neumann boundary is a unique [5] feature in FE method, in the sense that the Neumann boundary conditions are explicitly available within the FE formulation (as shown in later section). For elliptic (with respect to spatial discretization for heat conductions) equations, Cauchy boundary conditions (usually corresponding to convective conditions) are not appropriate in FE analysis since they prescribe both the function and its derivative at one spatial location.

3.2 Sequential Steps in the Computational Scheme

This section describes the sequential steps require to derive the final formulation of the finite element analysis by using a standard Galerkin procedure. Mainly, one multiplies the governing equation by a weighting function followed by integrating by parts and making use of Green's theorem in the plane. After that, one has to approximate the temperature field with the interpolation (shape) function chosen and specify the weighting function as the shape function in the integral statement. Once the element coefficient matrices are assembled, one can further rewrite the equation in a matrix system. Thus, particular solution procedures can be applied in solving the resultant equation systems corresponding to the steady-state and transient problems, respectively. A detailed step-by-step computational scheme is summarized below.

3.2.1 Element Discretization

In this step, one discretizes the physical domain into elements which are discrete spatial regions from the subdivision of a continuum. The choice of element, including type, numbering and allocation, involves a trade-off between convenience and complexity. In this scheme, standard triangular, rectangular, and the more general 4-node quadrilateral elements are chosen since the mesh information is directly obtained from the existing code in program VecTor2.

3.2.2 Integral Statement Establishment

In this step, one will need to write the equivalent integral statement for Eq. (2.6) and Eq. (2.10) by assuming that the essential boundary condition in Eq. (2.9) is automatically satisfied by the proper choice of the function T^h in Eq. (2.13).

One can use the Galerkin-weighted residual method to readily get the *strong form* statement as:

$$\int_{\Omega} v\left[\nabla\cdot(k\nabla T)+Q-rc\frac{\partial T}{\partial t}\right]d\Omega+\int_{\Gamma_q}\bar{v}\left[k\frac{\partial T}{\partial n}+\bar{q}\right]d\mathrm{G}=0 \tag{3.1}$$

Here, the introduced function v is the so-called *test function*.

Then, by using the following divergence theorem:

$$v\nabla\cdot(k\nabla T)=\nabla\cdot(vk\nabla T)-(\nabla v)\cdot(k\nabla T) \tag{3.2}$$

and Gauss's theorem (i.e. Eq. (2.16)) for the first term in the strong form equation, one can further obtain:

$$\int_{\Gamma} vk\nabla T\cdot\vec{n}d\Gamma-\int_{\Omega}(\nabla v)\cdot(k\nabla T)d\Omega+\int_{\Omega}vQd\Omega-rc\int_{\Omega}v\frac{\partial T}{\partial t}d\Omega+\int_{\Gamma_q}\bar{v}\left[k\frac{\partial T}{\partial t}+\bar{q}\right]d\Gamma=0 \tag{3.3}$$

Recognizing $\Gamma=\Gamma_T+\Gamma_q$, $\bar{q}_n=-k\dfrac{\partial T}{\partial n}=-(k\nabla T)\cdot\vec{n}$ and making $v=-\bar{v}$ (without loss of generality as both functions are arbitrary), one can finally reach the *weak form* as in Eq. (3.4) in which a lower order of continuity is required in the choice of the trial function T at the price of a higher continuity for test function v:

$$\int_{\Omega}\nabla^{\mathrm{T}}vk\nabla Td\mathrm{O}+rc\int_{\Omega}v\frac{\partial T}{\partial t}d\Omega-\int_{\Omega}vQd\mathrm{O}+\int_{\Gamma_{\mathrm{q}}}v\bar{q}d\mathrm{G}-\int_{\Gamma_{\mathrm{T}}}vk\frac{\partial T}{\partial n}d\mathrm{G}=0 \tag{3.4}$$

It can be seen that if the choice of T is restricted in satisfying the essential boundary condition (i.e. Eq. (2.9)) along Γ_T, the last term in the left-hand-side of Eq. (3.4) can be omitted. Then, Eq. (3.4) can be reduced to:

$$\int_{\Omega}\nabla^{\mathrm{T}}vk\nabla Td\mathrm{O}+rc\int_{\Omega}v\frac{\partial T}{\partial t}d\Omega-\int_{\Omega}vQd\mathrm{O}+\int_{\Gamma_{\mathrm{q}}}v\bar{q}d\mathrm{G}=0 \tag{3.5}$$

So, one can notice that the natural boundary condition (i.e. Eq. (2.10)) along Γ_q is satisfied since no variable T appears in the integrals taken along the boundary Γ_q in Eq. (3.5).

Through the above derivations, it can be seen that in the Galerkin procedure, the initial and boundary conditions (i.e. Eq. (2.8) and (2.9) respectively) do not appear explicitly in the formulations. Thus, the spatial interpolation functions must be chosen so as to satisfy the essential boundary conditions, and a temporal stepping scheme must be started from the initial condition. This does not allow a general formulation. Also, from the original

governing equations (i.e. Eq. (2.3) and (2.6) for steady-state and transient problem respectively), since second derivatives appear in the integrand, the consistency requirement implies that C^1 continuity elements must be used. This is a stringent requirement in practice. Therefore, it is desirable to reduce the order of differentiation to employ C^0 elements and to introduce the boundary conditions directly. This was realized through Eq. (3.5), with a trial (shape) function as constructed in the next step.

3.2.3 Shape Function Construction

The purpose of this step is to construct the shape function as described previously in the form of:

$$T \approx T^h = \sum_i N_i a_i = \mathbf{Na} \tag{2.13}$$

where, shape functions $N_i(x, y)$ postulate a spatial form for the dependent variable T in the element and are related to the number of nodes in that element as well as their numbering system. In turn, the function T is expanded as a spatially weighted summation over all the nodal points in Eq. (2.13). Each term in the summation process represents the contribution from a particular node i and contains terms N_i and a_i term defined for that node. This allows evaluation of the spatial integrals in Eq. (3.5).

Since the transient problem involves the time variable (where the space domain Ω is not subjected to change), the following *partial discretization* is adopted in this scheme:

$$T(x, y, t) \approx T^h = \sum_i N_i(x, y) a_i(t) = \mathbf{Na} \tag{3.6}$$

Clearly, the derivatives of **a** with respect to time *t* will remain in the final approximation, and one can expect the resultant equation will be a set of ordinary differential equations with *t* as the independent variable. Concerning the spatial shape functions $N_i(x, y)$, relevant equations in Section 2.2.2 can be employed for corresponding element types.

3.2.4 Element Matrix Calculation in Closed-Form

This step applies the key component of the Galerkin procedure, which is, substituting the shape functions as shown into Eq. (3.6) and prescribing the test functions as $v = w_j = N_j$. In doing so, Eq. (3.5) will lead to:

$$\int_{\Omega} \nabla^{\mathrm{T}} N_j k \nabla T d\Omega + rc \int_{\Omega} N_j \frac{\partial T}{\partial t} d\Omega - \int_{\Omega} N_j Q d\Omega + \int_{\Gamma_q} N_j \bar{q} d\Gamma = 0 \tag{3.7}$$

with $T(x, y, t) \approx T^h = \sum_i N_i(x, y) a_i(t) = \mathbf{Na}$ being such that the prescribed essential boundary conditions along Γ_T are satisfied. Then, it can be rewritten in a matrix system of ordinary differential equations as follows:

$$\mathbf{Ka} + \mathbf{C}\frac{d\,\mathbf{a}}{dt} + \mathbf{f} = 0 \tag{3.8}$$

where, in heat analysis, **K** is the conductance matrix which is symmetric and diagonally dominant; **C** is the capacitance matrix; **f** is the forcing term due to internal heat resources (e.g. heat of hydration) and natural boundary conditions; and **a** is the nodal vector containing the nodal values of dependent variable *T*. The entries of matrices can be evaluated by:

$$\mathrm{K}_{ij} = \mathrm{K}_{ji} = \int_{\Omega} \nabla^{\mathrm{T}} N_j k \nabla N_i d\mathrm{O} = \int_{\Omega} (\frac{\partial N_j}{\partial x} k \frac{\partial N_i}{\partial x} + \frac{\partial N_j}{\partial y} k \frac{\partial N_i}{\partial y}) d\mathrm{O} \tag{3.9.a}$$

$$\mathrm{C}_{ij} = C_{ji} = \int_{\Omega} N_i \mathbf{r} c N_j d\Omega \tag{3.9.b}$$

$$\mathrm{f}_j = -\int_{\Omega} N_j Q d\mathrm{O} + \int_{\Gamma_q} N_j \bar{q} d\mathrm{G} \tag{3.9.c}$$

In the case of steady-state problems, instead of Eq. (3.8), standard shape functions in Eq. (2.13) (with a_i simply a set of constants) are used. The final equations thus are always of an algebraic form as follows:

$$\mathbf{Ka} + \mathbf{f} = 0 \tag{3.10}$$

from which a unique set of parameters (nodal solution) can be determined.

From the above derivations, it can be seen that one equation of the form in Eq. (3.8) or Eq. (3.10) applies to every nodal point. Consequently, there is no need to specially construct the boundary equations, as often arises with the FD formulations.

It is important to point out that in the above calculation of element coefficient matrices, an average value of the thermal property (k or c) can be assigned to each element based on the current temperature distribution to simplify the formulation. Similarly, parameters Q and r are taken as constant, an average value within an element, as well as $\bar{q}$ over the length of the boundary of the relevant element.

Note that all the above calculations are evaluated in the local individual elements. One can use superscript "e" for elemental coefficient matrices, shape functions, and integral

domains to differentiate the global counterpart if necessary. The matrices in closed-form will be derived for all element types employed in this computational scheme. In the thermal analysis, since there is a single degree of freedom per node and material matrices are scalars (unlike counterpart matrices in traditional FE formulations), if the number of nodes in the element is equal to *n*, both the matrix **K** and **C** will be of the form $n \times n$. Similarly, vector **f** will be $n \times 1$.

3.2.4.1 Linear Triangular Element

For the triangle shown in Fig. 2.1, with the shape function described in Section 2.2.2.1, the following closed-form elemental matrices are calculated as follows:

- The conductance matrix **K**:

$$\mathbf{K}^e = \frac{k^e}{4A^e}\begin{bmatrix} c_1c_1 + d_1d_1 & c_1c_2 + d_1d_2 & c_1c_3 + d_1d_3 \\ c_2c_1 + d_2d_1 & c_2c_2 + d_2d_2 & c_2c_3 + d_2d_3 \\ c_3c_1 + d_3d_1 & c_3c_2 + d_3d_2 & c_3c_3 + d_3d_3 \end{bmatrix} \tag{3.11}$$

where

$$c_i = y_j - y_m \text{ and } d_i = x_m - x_j \tag{3.12}$$

with *i*, *j* and *m* are taken as 1, 2, 3 in a cyclic permutation.

- The capacitance matrix **C**:

$$\mathbf{C}^e = \frac{r^e c^e A^e}{12}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \tag{3.13}$$

- The forcing term **f**:

If side *ij* of the triangle is subjected to the natural boundary condition with a uniform flux q,

$$\mathbf{f}^e = \mathbf{f}_Q^e + \mathbf{f}_q^e = -\frac{Q^e A^e}{3}\begin{bmatrix}1\\1\\1\end{bmatrix} + \frac{q^e L_{ij}^e}{2}\begin{bmatrix}1\\1\\0\end{bmatrix} \tag{3.14}$$

3.2.4.2 Bilinear Rectangular Element

For the rectangle shown in Fig. 2.2, with the shape functions in Section 2.2.2.2, the following closed-form elemental matrices are calculated as follows:

- The conductance matrix **K**:

$$\mathbf{K}^e = \frac{k^e}{6ab}\begin{bmatrix} 2(a^2+b^2) & a^2-2b^2 & -a^2-b^2 & b^2-2a^2 \\ & 2(a^2+b^2) & b^2-2a^2 & -a^2-b^2 \\ & & 2(a^2+b^2) & a^2-2b^2 \\ \text{sym.} & & & 2(a^2+b^2) \end{bmatrix} \tag{3.15}$$

- The capacitance matrix **C**:

$$\mathbf{C}^e = \frac{r^e c^e ab}{9}\begin{bmatrix} 4 & 2 & 1 & 2 \\ & 4 & 2 & 1 \\ & & 4 & 2 \\ \text{sym.} & & & 4 \end{bmatrix} \tag{3.16}$$

- The forcing term **f**:

If edge 1-2 of the rectangle is subjected to the natural boundary condition with a uniform flux q,

$$\mathbf{f}^e = \mathbf{f}_Q^e + \mathbf{f}_q^e = -Q^e ab \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} + \frac{q^e L_{12}^e}{2} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \tag{3.17}$$

3.2.4.3 General Four-Node Quadrilateral Element

For the quadrilateral shown in Fig. 2.3, by the algorithm developed below the closed-form matrices can be directly calculated in terms of the nodal coordinates and thermal properties only.

- The conductance matrix **K**:

The calculation of **K** matrix here is reminiscent of the work in reference [34]. To save computational efforts, one can use symmetry of the matrix and separate the entries into three groups (*B*, *D* and *F*) as follows:

$$\mathbf{K} = \begin{bmatrix} K_{11} & K_{12} & K_{13} & K_{14} \\ & K_{22} & K_{23} & K_{24} \\ & & K_{33} & K_{34} \\ \text{sym.} & & & K_{44} \end{bmatrix} = \begin{bmatrix} B & D & F & D \\ & B & D & F \\ & & B & D \\ & & & B \end{bmatrix} \tag{3.18}$$

All entries in matrix **K** are of the form:

$$K_{ij} = \frac{k^e}{2} \left[\frac{A_2 s_1 + f_1 s_2}{3A_2^2 - f_1^2} + \frac{A_2 t_1 + f_2 t_2}{3A_2^2 - f_2^2} \right] \tag{3.19}$$

A_2 is twice the area of the element, which can be calculated as follows:

$$A_2 = (x_1 - x_3)(y_2 - y_4) - (y_1 - y_3)(x_2 - x_4) \tag{3.20}$$

and

$$f_1 = (x_1 + x_3)(y_2 - y_4) - (y_1 + y_3)(x_2 - x_4) + 2(x_2 y_4 - y_2 x_4) \tag{3.21.a}$$

$$f_2 = (x_3 - x_1)(y_2 + y_4) - (y_3 - y_1)(x_2 + x_4) + 2(x_1 y_3 - y_1 x_3) \tag{3.21.b}$$

All other coefficients will be formulated only for the *parent* entry within each group and remaining entries can then be calculated through a nodal coordinate transformation as follows:

$$x_1, y_1 \Rightarrow x_2 y_2 \Rightarrow x_3 y_3 \Rightarrow x_4 y_4 \Rightarrow x_1, y_1 \tag{3.22}$$

Here, $\Rightarrow$ indicates 'overwrite'.

All detailed calculations are described in Table 3.1.

Group[#]	Parent	s_1	s_2	t_1	t_2	Order[*]
B	K_{11}	$2\begin{Bmatrix}(y_2-y_4)^2\\+(x_2-x_4)^2\end{Bmatrix}$	$-s_1/2$	$(y_2-y_3)^2+(y_3-y_4)^2$ $+(y_4-y_2)^2+(x_2-x_3)^2$ $+(x_3-x_4)^2+(x_4-x_2)^2$	$(y_3-y_4)^2-(y_2-y_3)^2$ $+(x_3-x_4)^2-(x_2-x_3)^2$	K_{11} $\Rightarrow K_{44}$ $\Rightarrow K_{33}$ $\Rightarrow K_{22}$
D	K_{14}	$(y_2-y_4)(2y_1-y_3-y_2)$ $+(x_2-x_4)(2x_1-x_3-x_2)$	$(y_2-y_4)(y_2-y_1)$ $+(x_2-x_4)(x_2-x_1)$	$(y_3-y_1)(2y_4-y_3-y_2)$ $+(x_3-x_1)(2x_4-x_3-x_2)$	$(y_3-y_1)(y_3-y_4)$ $+(x_3-x_1)(x_3-x_4)$	K_{14} $\Rightarrow K_{34}$ $\Rightarrow K_{23}$ $\Rightarrow K_{12}$
F	K_{13}	$-\begin{Bmatrix}(y_2-y_4)^2\\+(x_2-x_4)^2\end{Bmatrix}$	0	$(y_3+y_1)(y_4+y_2)$ $-2(y_2-y_4)^2$ $-2(y_1y_3+y_2y_4)$ $+(x_3+x_1)(x_4+x_2)$ $-2(x_2-x_4)^2$ $-2(x_1x_3+x_2x_4)$	$(y_2-y_4)(y_1-y_2+y_3-y_4)$ $+(x_2-x_4)(x_1-x_2+x_3-x_4)$	K_{13} $\Rightarrow K_{24}$

Legend:

#: grouping is separated based on the freedom characteristics as follows: $\begin{cases}B: \text{Diagonal (freedom on node itself);}\\ D: \text{freedom at adjacent nodes;}\\ F: \text{freedom at opposite nodes.}\end{cases}$.

*: computing sequence when one transforms parent term to remaining entries within each group.

Table 3.1 Calculation of entries in K matrix for four-node quadrilateral element

- The capacitance matrix **C**:

The calculation of matrix **C** is based on the developed MATHEMATICA subroutine which runs symbolic operations mathematically. Also, due to symmetry, only those entries on and above the main diagonal need to be calculated.

$$\mathbf{C}^e = r^e c^e \begin{bmatrix} 1st\ \ Row & & & \\ & 2nd\ \ Row & & \\ & & 3rd\ \ Row & \\ \text{sym.} & & & 4th\ Row \end{bmatrix} \tag{3.23}$$

where the formulation of entries in those four rows is given in Table 3.2.

- The forcing term **f**:

If edge 1-2 of the rectangle is subjected to the natural boundary condition with a uniform flux q,

$$\mathbf{f}^e = \mathbf{f}_Q^e + \mathbf{f}_q^e = -Q^e \frac{A^e}{4} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} + \frac{q^e L_{12}^e}{2} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \tag{3.24}$$

1st Row (four entries: $C_{11}, C_{12}, C_{13}, C_{14}$):

$$\left(\frac{1}{128}\left(-\frac{32x2y1}{3}+\frac{32x4y1}{3}+\frac{32x1y2}{3}-\frac{32x3y2}{9}-\frac{64x4y2}{9}+\frac{32x2y3}{9}-\frac{32x4y3}{9}-\frac{32x1y4}{3}+\frac{64x2y4}{9}+\frac{32x3y4}{9}\right)\right)$$

$$\left(\frac{1}{72}\left(-3x2y1+x3y1+2x4y1+3x1y2-2x3y2-x4y2-x1y3+2x2y3-x4y3-2x1y4+x2y4+x3y4\right)\right)$$

$$\left(\frac{1}{128}\left(-\frac{16x2y1}{9}+\frac{16x4y1}{9}+\frac{16x1y2}{9}-\frac{16x3y2}{9}+\frac{16x2y3}{9}-\frac{16x4y3}{9}-\frac{16x1y4}{9}+\frac{16x3y4}{9}\right)\right)$$

$$\left(\frac{1}{128}\left(-\frac{32x2y1}{9}-\frac{16x3y1}{9}+\frac{16x4y1}{3}+\frac{32x1y2}{9}-\frac{16x3y2}{9}-\frac{16x4y2}{9}+\frac{16x1y3}{9}+\frac{16x2y3}{9}-\frac{32x4y3}{9}-\frac{16x1y4}{3}+\frac{16x2y4}{9}+\frac{32x3y4}{9}\right)\right)$$

2nd Row (three entries: C_{22}, C_{23}, C_{24}):

$$\left(\frac{1}{128}\left(-\frac{32x2y1}{3}+\frac{64x3y1}{9}+\frac{32x4y1}{9}+\frac{32x1y2}{3}-\frac{32x3y2}{3}-\frac{64x1y3}{9}+\frac{32x2y3}{3}-\frac{32x4y3}{9}-\frac{32x1y4}{9}+\frac{32x3y4}{9}\right)\right)$$

$$\left(\frac{1}{128}\left(-\frac{32x2y1}{9}+\frac{16x3y1}{9}+\frac{16x4y1}{9}+\frac{32x1y2}{9}-\frac{16x3y2}{3}+\frac{16x4y2}{9}-\frac{16x1y3}{9}+\frac{16x2y3}{3}-\frac{32x4y3}{9}-\frac{16x1y4}{9}-\frac{16x2y4}{9}+\frac{32x3y4}{9}\right)\right)$$

$$\left(\frac{1}{128}\left(-\frac{16x2y1}{9}+\frac{16x4y1}{9}+\frac{16x1y2}{9}-\frac{16x3y2}{9}+\frac{16x2y3}{9}-\frac{16x4y3}{9}-\frac{16x1y4}{9}+\frac{16x3y4}{9}\right)\right)$$

3rd Row (two entries: C_{33}, C_{34}):

$$\left(\frac{1}{128}\left(-\frac{32x2y1}{9}+\frac{32x4y1}{9}+\frac{32x1y2}{9}-\frac{32x3y2}{3}+\frac{64x4y2}{9}+\frac{32x2y3}{3}-\frac{32x4y3}{3}-\frac{32x1y4}{9}-\frac{64x2y4}{9}+\frac{32x3y4}{3}\right)\right)$$

$$\left(\frac{1}{72}\left(-x2y1-x3y1+2x4y1+x1y2-2x3y2+x4y2+x1y3+2x2y3-3x4y3-2x1y4-x2y4+3x3y4\right)\right)$$

4th Row (one entry: C_{44}):

$$\left(\frac{1}{128}\left(-\frac{32x2y1}{9}-\frac{64x3y1}{9}+\frac{32x4y1}{3}+\frac{32x1y2}{9}-\frac{32x3y2}{9}+\frac{64x1y3}{9}+\frac{32x2y3}{9}-\frac{32x4y3}{3}-\frac{32x1y4}{3}+\frac{32x3y4}{3}\right)\right)$$

Table 3.2 Calculation of entries in C matrix for four-node quadrilateral element

3.2.5 Global Equation Assembly

In this step, the overall system is assembled by starting with zero matrices for **K**, **C** and **f**. Contributions from each element are sequentially added to the contents of the global system matrices. While mathematically governed by compatibility and equilibrium conditions, such an assembly process can be physically interpreted as reconnecting the discrete members back into the complete structure. The key operation of this assembly process is the placement of these contributions. Therefore, the mapping between the local element degree of freedom and global degree of freedom has to be established by using the mesh information.

After all elements are assembled, it is necessary to condense the equations; that is, removing the nodal equations corresponding to the boundary nodes with Dirichlet conditions (i.e. Eq. (2.9)). Thus, the resulting equations are ready for solution.

3.2.6 Equation System Solution

In this step, particular solution procedures will be applied to solve the resultant equation system corresponding to steady-state and transient problems. For transient problems, the system forms a set of first-order *differential* equations in time while a set of *algebraic* equations arises in steady-state problems. So, they are treated separately as follows.

3.2.6.1 Steady-State Problem

If the matrix **K** can be inverted, Eq. (3.10) can be solved by direct methods; for example, by Gaussian elimination scheme. Since the matrix system is generally large, the methods taking advantage of the symmetrical, sparse and banded properties of **K** may lead to many specialized techniques, among which is the iterative method.

Instead of attempting to solve equations directly, iterative methods work with the individual equations which are usually found by solving the terms on the diagonal from Eq. (3.10). Iterative methods start from a guessed initial field, and sequentially improve the field by using successive iterations until that individual equation is satisfied to some extent, in the sense of reaching the tolerance specified in the beginning. It is worthy to note that the similar iterating equations must be modified near the appropriate boundary to account for the prescribed boundary conditions in the FD formulation while it is not the case in the current FE computational scheme.

3.2.6.2 Transient Problem

For transient problems, the resultant equation as in Eq. (3.8) is a set of first-order differential equations with time t as the independent variable. Thus, the solutions have to proceed with increasing time until the results are obtained over a specific time level or until the steady state is attained. In this section, attention is given to the construction of the time discretization since the spatial one is exactly same as that for steady-state problems aforementioned. After setting the time interval, one then can march in time, obtaining the temperature distribution at each time level in terms of the one at the preceding interval.

The most common procedure is to use a finite difference in time with two levels, which was described in Section 2.3. Then, one can approximate the derivative term $\frac{d\,\mathbf{a}}{dt}$ by two-point finite difference as follows:

$$\frac{d\,\mathbf{a}}{dt} = \frac{T_{n+1} - T_n}{\Delta t} \tag{3.25}$$

At the same time, one has to make a decision regarding at what time level to evaluate the temperature in Eq. (3.8). The most popular scheme is to use the *trapezoidal rule* which uses a linear interpolation between levels n and $n+1$:

$$T_{n+r} = (1-r)T_n + rT_{n+1} \tag{3.26}$$

Substituting Eq. (3.25) and Eq. (3.26) into Eq. (3.8) leads to the so-called *generalized mid-point method* [31]:

$$\mathbf{K}(T_{n+r}, t_{n+r})T_{n+r} + \mathbf{C}(T_{n+r}, t_{n+r})\frac{T_{n+1} - T_n}{\Delta t} + \mathbf{f}(T_{n+r}, t_{n+r}) = 0 \tag{3.27}$$

where $t_{n+r} = t_n + r\Delta t$ with subscript *n* representing the *n*th time step.

Eq. (47) can be rearranged as:

$$\left[\frac{\mathbf{C}_{n+r}}{\Delta t} + r\,\mathbf{K}_{n+r}\right]T_{n+1} = \left[\frac{\mathbf{C}_{n+r}}{\Delta t} - (1-r)r\,\mathbf{K}_{n+r}\right](T_n) - \mathbf{f}_{n+r} \tag{3.28}$$

where $\mathbf{C}_{n+r}$ means $\mathbf{C}(T_{n+r}, t_{n+r})$, so too will the $\mathbf{K}_{n+r}$ and $\mathbf{f}_{n+r}$.

As described in Section 2.3, different members of this family can be identified by changing the value of *r* from 0 to 1.0. Both the choice of $r = 1/2$ (for accuracy considerations [10]) and $r = 2/3$ (for stability purpose [11]) will be employed in this computational scheme.

Note that, due to the implicit nature, an iteration loop over each time step is required to maintain accuracy in nonlinear transient problems. That is, the coefficient matrices' evaluation, assembly and solution have to be performed every iteration within each time step, which means nonlinear transient analyses are computationally intensive.

3.3 Numerical Properties of the Computational Scheme

In many situations, questions arise regarding the errors involved in the numerical computations, as well as the consistency, stability and the convergence of the computational scheme. Such matters are discussed in the following sections.

3.3.1 Accuracy

It is important to keep in mind that numerical solutions are only *approximate* solutions. In addition to the errors that might be introduced in the course of development of the solution algorithm in programming and machine round-off in calculations, numerical solutions always include the following three kinds of errors:

- Modeling errors: the difference between actual heat flow problem and the exact solution of the mathematical model described by PDEs. They may be considered negligible since the governing equations (Eq. (2.3) and (2.6)) represent a sufficiently accurate [4] model of the flow. However, it is noticed that modeling errors are also introduced by simplifying the geometry of the solution domain, simplifying initial and boundary conditions, and neglecting of thermal properties' temperature-dependency.

In the current computational scheme, the variable thermal properties will be taken into account. Also, the geometry is limited to the region consisting of straight-lined segments only; consequently the use of triangular and quadrilateral elements will be sufficient in modeling the geometry. During the application of boundary conditions, the concrete surface temperature is assumed the same as the ambient temperature. Although in some case, *skin effect* was found to be significant [32], it is ignored in the current scheme since convection analysis will be involved in this simulation.

- Discretization errors in approximation: the difference between the exact solution of the governing PDEs and exact solution of discrete approximation. Usually, one selects the approximations prior to writing a calculation code so the spatial and temporal grid resolutions are parameters at the user's disposal to control the accuracy.

In the current computational scheme, bilinear quadrilateral and linear triangular elements are employed for spatial approximation. The former is more accurate since it contains additional term *xy* in the interpolation functions. It is important to remark that by using the closed-form coefficient matrices, the spatial integrals are exactly evaluated in this aspect. For the time discretization, the two-level finite difference scheme employed together with the Crank-Nicholson method is of second-order approximation from a Taylor-series truncation error analysis [10]. In the case where more stability is required, then the backup Galerkin scheme is available for use.

- Iteration errors in solution: the difference between the iterative and exact solutions of the algebraic equation system if an iterative process is involved. Usually, iteration is continued until the magnitude of the residual has been reduced to a certain amount and the acceptance criterion has been reached. Although this kind of error has nothing to do with the discretization itself, the effort required to reduce the error to a given magnitude grows as the number of discrete elements increases. Also, the machine round-off error evolves with the iteration calculations. In the current computational scheme, the efficient (in the sense of convergence) Gauss-Seidel iteration scheme is employed, one of the "side" effects of which is the reduction of the round-off errors created since it converges faster than standard schemes.

3.3.2 Consistency

A discretization scheme is called consistent if the discretized equations converge to the given PDEs when both the time step and grid spacing tend to be zero. In other words, the consistency condition of the numerical method requires that the discretization error approaches zero as the mesh is refined.

In the spatial FE discretization, two requirements for the consistency condition are addressed below.

1. Completeness requirement: the element must have enough approximation power to capture the analytical solution in the limit of a mesh refinement process. More precisely, the element shape functions must represent exactly all polynomial terms of order $\leq m$ (m is the *variational index*) in the Cartesian coordinates. In this aspect, the employed linear and bilinear elements are obviously satisfied with $m = 1$, as the field variable T appearing in the weak form integral statement (i.e. Eq. (2.6)) possesses derivatives up to the first order only.

2. Compatibility requirement: the shape functions must provide continuity between elements. Physically, this requirement insures that no material gaps appear as the elements deform. Mathematically, patch shape functions must be C^{m-1} (m as above) continuous between interconnected elements, and C^m piecewise differentiable inside each element. Again, obviously the employed finite element types are satisfied in this aspect.

Concerning the two-level finite difference approximation in time, it is plain to see that the approximation error ($O(\Delta t^2)$) vanishes as $\Delta t \to 0$, and consequently it is consistent.

3.3.3 Stability

Even if the approximations are consistent, the solution of the discretized equation system will not necessarily become the exact solution of the PDEs in the limit of small spatial and temporal step sizes. For this to happen, the solution method has to be stable. A numerical solution method is said to be stable if it does not amplify, without bounds, the errors that appear in the course of numerical solution process.

In the spatial FE discretization, stability is not a property of interpolation functions *per se* but of the employed element as well as its geometrical definition. Mathematically, there are two requirements for the stability condition:

1. Rank Sufficiency: the element stiffness matrix must not possess any zero-energy kinematic modes other than rigid body modes.

2. Jacobian Positiveness: the geometry of the element must preclude the excessive element distortion such that the determinant of the Jacobian matrix remains positive everywhere.

With the choice of bilinear quadrilateral or linear triangular elements, the above stability conditions are naturally satisfied since closed-form evaluation, instead of Gauss quadrature, is employed in the current computational scheme.

Regarding the temporal discretization, the two-level finite difference together with implicit Crank-Nicholson or Galerkin method is unconditionally stable for linear or weakly nonlinear problems and shows good but prudent [31] stability for highly nonlinear problems. Consequently, the coefficient matrices will be evaluated during every round of iteration within each time step in the current computational scheme.

3.3.4 Convergence

The term convergence is used not only in conjunction with error reduction in iterative solution methods, but is also often associated with the convergence of numerical solutions towards a grid-independent solution, in which case it is closely linked [33] to discretization error. Therefore, in an iterative analysis, two different concepts of convergence are required for both discretization and iteration:

1. Discretization: Numerical solutions are said to be convergent if the discretization error approaches zero as the grid spacing and time step are refined. In this sense, the exact solution is approached numerically by mesh refinement. The well-known Lax-Wendroff theorem says that satisfying consistency and stability is the necessary and sufficient condition for convergence. From the previous description on consistency and stability, one can see the current computational scheme will converge in the sense of discretization.

2. Iteration: Iterative processes are said to be convergent if the iterative errors approach zero as the number of iterations increases. In practice, one can specify the criterion for convergence, such as:

$$\left|T^{s+1} - T^{s}\right| \le e \tag{3.29}$$

to determine whether or not to continue to the next round of iteration.

Although the initial guessed distribution of T is a very important factor in the iterative method, convergence with the iterative method does not depend on that initial guess but on the character of the coefficient matrices. In the current computational scheme, the diagonal dominant coefficient matrices guarantee the convergence, and the adoption of Gauss-Seidel method makes it faster than standard iterations.

3.4 Closure

In this chapter, a computational scheme was described step-by-step. With the use of proposed FE computational scheme, some difficulties were solved, which include accounting for temperature-dependent thermal properties, specifying time-varying boundary thermal loads, developing closed-form coefficient matrices, and considering heat of hydration. Also, various numerical properties of this computational scheme were evaluated. In the next chapter, some relative codes will be implemented and presented.

Chapter 4 Code Implementation

This chapter presents the developed subroutine V2HEAT, which implements the proposed computational scheme described in Chapter 3. Also, necessitated by follow-up stress and deformation analyses after temperature calculations, reduction of mechanical properties at evaluated temperatures is calculated and the relative code is implemented in V2TRED. In addition, some auxiliary subroutines are described here as well, which include the code in Mathematica for the closed-form element matrix formulation and the code in MATLAB for temperature distribution contour plot.

The developed codes will then be embedded into program VecTor2, a nonlinear finite element analysis (NLFEA) program for reinforced concrete two-dimensional structures. The behaviour analysis platform incorporated into the VecTor2 enables calculation of the response of reinforced concrete elements subject to in-plane normal and shear stresses in the second-order accuracy. VecTor2 primarily reads three types of input files, i.e. job, structure and load files. VecTor2 outputs binary and ASCII test files for analysis results for which the software Augustus provides graphically post-processing capabilities.

4.1 V2HEAT in FORTRAN

Generally, the subroutine V2HEAT (listed in Appendix C) is developed to facilitate program VecTor2 in performing 2D nonlinear steady-state and transient conduction thermal analysis by using the FE method.

Functionally, V2HEAT compromises between the often conflicting demands of generality and efficiency of applicability and ease of use. Globally, this subroutine has the same breadth of applicability as the FE computational scheme described in Chapter 3. It accounts for the nonlinear temperature-dependent thermal properties for various concrete and steel. Also, it is applicable to time-varying thermal loads. While it is limited in that it currently does not allow for heat-of-hydration, it may be later incorporated by specifying corresponding thermal loading, as formulated in Chapter 3. Furthermore, the same finite element data used in main-program VecTor2 are employed in this heat analysis.

Structurally, V2HEAT consists of one main sub-program and some subsidiary subroutines. The main sub-program is the core which has a controlling function: initializing the calculations according to VecTor2's input, monitoring the progress of calculations performed as they proceed through different time steps, iterating operations within each time step, and producing and restoring the resultant temperature fields at different stages. The subsidiary subroutines perform corresponding particular operations or calculations and are invoked by the main sub-program. Thus, the code has been deliberately structured so that it adapts to different problems.

Schematically, a flowchart to represent the stream of implemented procedures is shown in Fig. 4.1, from which one can see that the loop of iteration for nonlinear thermal properties and the loops of time-advancing for transient analysis are at the core of the whole computational scheme. In practice, the code specifies the criterion of convergence for iteration and uses an external time frame (if any, e.g. time-step analysis model in VecTor2) or internal time interval for time-advancing.

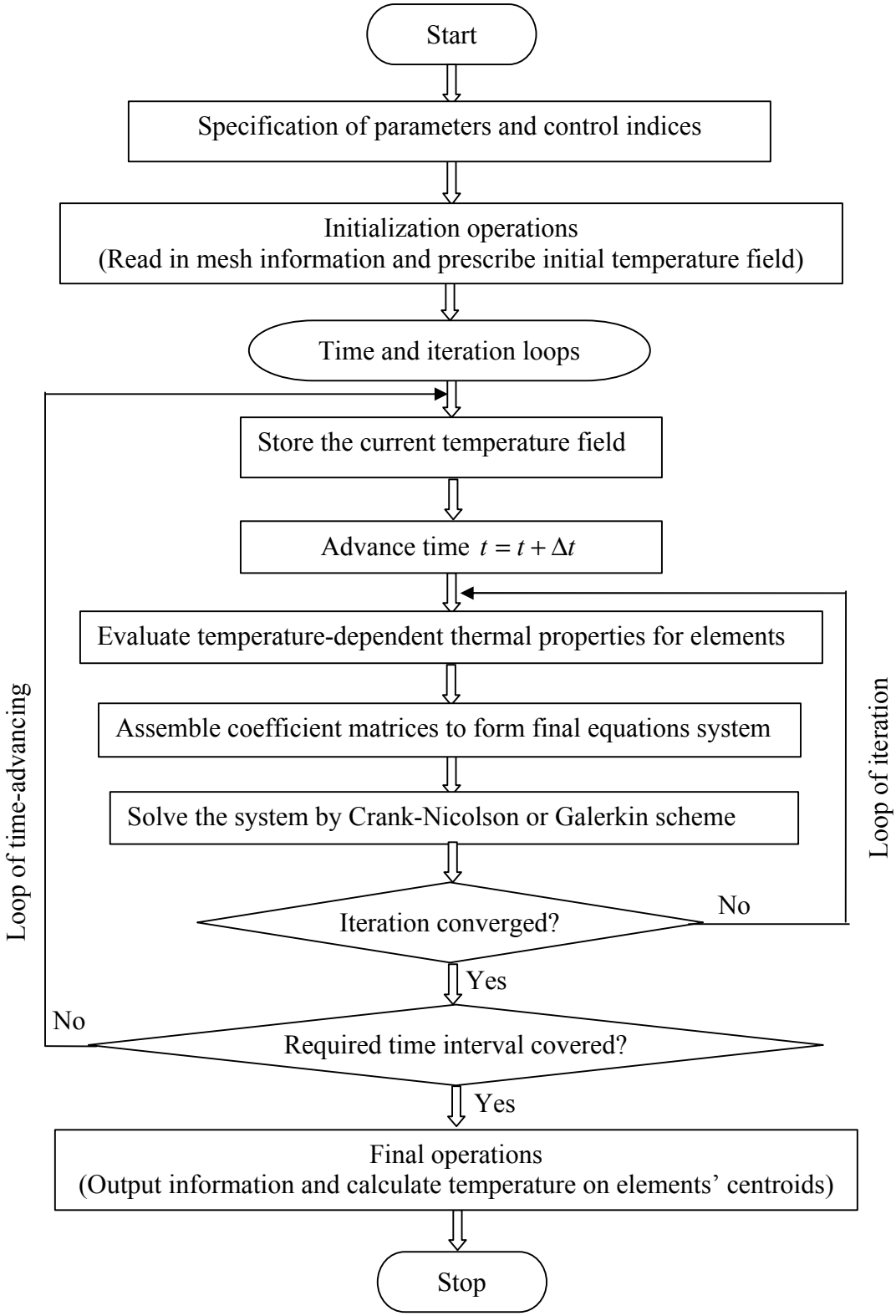


Figure 4.1 Flowchart of V2HEAT

4.2 V2TRED in FORTRAN

In addition to V2HEAT, the subroutine V2TRED is modified and updated to account for variation of mechanical properties at evaluated temperature up to $1200^{\circ}C$. A simple annotated source code is attached in Appendix D. For those properties which are given directly in a tier-by-tier manner, the relative coding is quite straightforward. The only exception is the thermal expansion property. The variation of this property for both concrete and steel was plotted in Section 2.4.2. It can be seen that both concrete and steel's thermal expansion would cease altogether at certain degree of temperature. However, the thermal strain is developed progressively, so zero thermal expansion implies only that no change of thermal strain occurs at that temperature. In addition, the use of formula for calculating thermal strain at high temperatures in VecTor2 does not allow one to specify the thermal expansion reduction (in this case, increasing) factor to be zero (otherwise, the thermal strain will be zero from Eq. (4.1)).

$$e^{thermal} = \Delta T \cdot \alpha(T) = (T - T_{refer}) \cdot f \cdot \alpha_{refer} \tag{4.1}$$

where f is the reduction factor of thermal expansion at temperature T. In fact, the above equation is originally based on the assumption of a constant thermal expansion. In order to keep this equation valid, the use of zero thermal expansion during some range of temperatures is implemented in an implicit way. That is, the $\alpha(T)$ in the equation is taken (interpreted) as the average value of thermal expansion in the temperature range T_{refer} to T, instead of the one in that temperature T. As a result, the reduction factor calculated from the formula above is actually the range-average thermal expansion. In doing so, the real zero thermal expansion and non-zero thermal strain are maintained.

4.3 Closed-Form Element Matrices in MATHEMATICA

VecTor2 models reinforced concrete elements (plain concrete or concrete with smeared reinforcement) by using three types of element: constant strain triangle, standard rectangle and bilinear four-node quadrilateral. All element matrices are formulated in the closed-from in VecTor2, without resorting to the numerical integration. Due to the constant Jacobian, the closed-form formulations can be easily obtained for the triangular and rectangular elements, even with a full (but still symmetrical) material matrix **D** as follows:

$$\mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} \\ & D_{22} & D_{23} & D_{24} \\ & & D_{33} & D_{34} \\ \text{sym.} & & & D_{44} \end{bmatrix} \tag{4.2}$$

While the conductance and capacitance matrices for triangle and rectangle may be available in some literature, they can be easily obtained by using high-level languages, such as MATHEMATICA. In the example of the rectangular element shown in Chapter 2 (i.e. Fig. 2.2), the MATHEMATICA code is given in Figure 4.2 for general stiffness matrix development. The symbolic-form element matrix is also outputted in the same figure, with the assumption that the thickness of the element is unity. The stiffness matrix obtained can then be reduced to the conductance matrix as given in Section 3.2.4.2 since in heat analysis the material matrix is actually a scalar instead of a 3×3 matrix and the degree of freedom on each node is single. The corresponding capacitance matrix development is given in Fig. 4.3. Since the element nodal coordinates in the code are specified as (x_i, y_i) instead of $(\pm a, \pm b)$, it can be actually applied to a four-node quadrilateral element as well.

```
x = {-a, a, a, -a}; y = {-b, -b, b, b};
Nf = {(1 - ξ) * (1 - η), (1 + ξ) * (1 - η),
     (1 + ξ) * (1 + η), (1 - ξ) * (1 + η)} / 4;
dNξ = D[Nf, ξ]; dNη = D[Nf, η];
J11 = dNξ.x; J12 = dNξ.y; J21 = dNη.x; J22 = dNη.y;
J = {{J11, J12}, {J21, J22}};
Jdet = Simplify[J11 * J22 - J12 * J21];
detJ = Simplify[J11 * J22 - J12 * J21];
dNx = (J22 * dNξ - J12 * dNη) / detJ; dNx = Simplify[dNx];
dNy = (-J21 * dNξ + J11 * dNη) / detJ; dNy = Simplify[dNy];
B = {Flatten[Table[{dNx[[i]], 0}, {i, 4}]],
    Flatten[Table[{0, dNy[[i]]}, {i, 4}]],
   Flatten[Table[{dNy[[i]], dNx[[i]]}, {i, 4}]]};
Dmat = {{D11, D12, D13}, {D12, D22, D23}, {D13, D23, D33}};
K = Simplify[Transpose[B].(Dmat.B)];
For [i = 1, i ≤ 8, i++,
  For [j = 1, j ≤ 8, j++,
    K[[i, j]] = Integrate[K[[i, j]], {ξ, -1, 1}];
    K[[i, j]] = Integrate[K[[i, j]], {η, -1, 1}];
   ];];
K1 = K[[1]]; K2 = K[[2]]; K3 = K[[3]]; K4 = K[[4]];
K5 = K[[5]]; K6 = K[[6]]; K7 = K[[7]]; K8 = K[[8]];
Print["Selected row of K Matrix due to space limit"]
Print["K1=", K1 // MatrixForm, "K6=", K6 // MatrixForm];
```

Selected row of K Matrix due to space limit

$$K1=\begin{pmatrix} \frac{\frac{16b^2D11}{3}+8abD13+\frac{16a^2D33}{3}}{16a^2b^2} \\ \frac{4abD12+\frac{16b^2D13}{3}+\frac{16a^2D23}{3}+4abD33}{16a^2b^2} \\ -\frac{\frac{16b^2D11}{3}-\frac{8a^2D33}{3}}{16a^2b^2} \\ -\frac{-4abD12+\frac{16b^2D13}{3}-\frac{8a^2D23}{3}+4abD33}{16a^2b^2} \\ \frac{-\frac{8b^2D11}{3}-8abD13-\frac{8a^2D33}{3}}{16a^2b^2} \\ \frac{-4abD12-\frac{8b^2D13}{3}-\frac{8a^2D23}{3}-4abD33}{16a^2b^2} \\ \frac{\frac{8b^2D11}{3}-\frac{16a^2D33}{3}}{16a^2b^2} \\ -\frac{4abD12-\frac{8b^2D13}{3}+\frac{16a^2D23}{3}-4abD33}{16a^2b^2} \end{pmatrix} \quad K6=\begin{pmatrix} \frac{-4abD12-\frac{8b^2D13}{3}-\frac{8a^2D23}{3}-4abD33}{16a^2b^2} \\ \frac{-\frac{8a^2D22}{3}-8abD23-\frac{8b^2D33}{3}}{16a^2b^2} \\ -\frac{-4abD12-\frac{8b^2D13}{3}+\frac{16a^2D23}{3}+4abD33}{16a^2b^2} \\ -\frac{\frac{16a^2D22}{3}-\frac{8b^2D33}{3}}{16a^2b^2} \\ \frac{4abD12+\frac{16b^2D13}{3}+\frac{16a^2D23}{3}+4abD33}{16a^2b^2} \\ \frac{\frac{16a^2D22}{3}+8abD23+\frac{16b^2D33}{3}}{16a^2b^2} \\ -\frac{4abD12+\frac{16b^2D13}{3}-\frac{8a^2D23}{3}-4abD33}{16a^2b^2} \\ \frac{\frac{8a^2D22}{3}-\frac{16b^2D33}{3}}{16a^2b^2} \end{pmatrix}$$

Figure 4.2 Rectangle's stiffness matrix calculation in Mathematica

```
x = {x1, x2, x3, x4}; y = {y1, y2, y3, y4};
Nf = {{(1 - ξ) * (1 - η), (1 + ξ) * (1 - η), (1 + ξ) * (1 + η), (1 - ξ) * (1 + η)} / 4};
NfT = Transpose[Nf]; dNξ = D[Nf, ξ]; dNη = D[Nf, η];
J11 = dNξ.x; J12 = dNξ.y; J21 = dNη.x; J22 = dNη.y; J = {{J11, J12}, {J21, J22}};
detJ = Simplify[J11 * J22 - J12 * J21]; CM = NfT.Nf;
For [i = 1, i <= 4, i++, For [j = 1, j <= 4, j++,
    CM[[i, j]] = Integrate[detJ * CM[[i, j]], {ξ, -1, 1}, {η, -1, 1}];];];
C1 = CM[[1]]; Print["", C1];
```

$$\left\{\left\{\frac{1}{128}\left(-\frac{32\,x2\,y1}{3}+\frac{32\,x4\,y1}{3}+\frac{32\,x1\,y2}{3}-\frac{32\,x3\,y2}{9}-\frac{64\,x4\,y2}{9}+\frac{32\,x2\,y3}{9}-\frac{32\,x4\,y3}{9}-\frac{32\,x1\,y4}{3}+\frac{64\,x2\,y4}{9}+\frac{32\,x3\,y4}{9}\right)\right\},\right.$$
$$\left\{\frac{1}{72}\left(-3\,x2\,y1+x3\,y1+2\,x4\,y1+3\,x1\,y2-2\,x3\,y2-x4\,y2-x1\,y3+2\,x2\,y3-x4\,y3-2\,x1\,y4+x2\,y4+x3\,y4\right)\right\},$$
$$\left\{\frac{1}{128}\left(-\frac{16\,x2\,y1}{9}+\frac{16\,x4\,y1}{9}+\frac{16\,x1\,y2}{9}-\frac{16\,x3\,y2}{9}+\frac{16\,x2\,y3}{9}-\frac{16\,x4\,y3}{9}-\frac{16\,x1\,y4}{9}+\frac{16\,x3\,y4}{9}\right)\right\},$$
$$\left.\left\{\frac{1}{128}\left(-\frac{32\,x2\,y1}{9}-\frac{16\,x3\,y1}{9}+\frac{16\,x4\,y1}{3}+\frac{32\,x1\,y2}{9}-\frac{16\,x3\,y2}{9}-\frac{16\,x4\,y2}{9}+\frac{16\,x1\,y3}{9}+\frac{16\,x2\,y3}{9}-\frac{32\,x4\,y3}{9}-\frac{16\,x1\,y4}{3}+\frac{16\,x2\,y4}{9}+\frac{32\,x3\,y4}{9}\right)\right\}\right\}$$

```
C3 = CM[[3]]; Print["", C3];
```

$$\left\{\left\{\frac{1}{128}\left(-\frac{16\,x2\,y1}{9}+\frac{16\,x4\,y1}{9}+\frac{16\,x1\,y2}{9}-\frac{16\,x3\,y2}{9}+\frac{16\,x2\,y3}{9}-\frac{16\,x4\,y3}{9}-\frac{16\,x1\,y4}{9}+\frac{16\,x3\,y4}{9}\right)\right\},\right.$$
$$\left\{\frac{1}{128}\left(-\frac{32\,x2\,y1}{9}+\frac{16\,x3\,y1}{9}+\frac{16\,x4\,y1}{9}+\frac{32\,x1\,y2}{9}-\frac{16\,x3\,y2}{3}+\frac{16\,x4\,y2}{9}-\frac{16\,x1\,y3}{9}+\frac{16\,x2\,y3}{3}-\frac{32\,x4\,y3}{9}-\frac{16\,x1\,y4}{9}-\frac{16\,x2\,y4}{9}+\frac{32\,x3\,y4}{9}\right)\right\},$$
$$\left\{\frac{1}{128}\left(-\frac{32\,x2\,y1}{9}+\frac{32\,x4\,y1}{9}+\frac{32\,x1\,y2}{9}-\frac{32\,x3\,y2}{3}+\frac{64\,x4\,y2}{9}+\frac{32\,x2\,y3}{3}-\frac{32\,x4\,y3}{3}-\frac{32\,x1\,y4}{9}-\frac{64\,x2\,y4}{9}+\frac{32\,x3\,y4}{3}\right)\right\},$$
$$\left.\left\{\frac{1}{72}\left(-x2\,y1-x3\,y1+2\,x4\,y1+x1\,y2-2\,x3\,y2+x4\,y2+x1\,y3+2\,x2\,y3-3\,x4\,y3-2\,x1\,y4-x2\,y4+3\,x3\,y4\right)\right\}\right\}$$

Note that the above capacitance matrix excludes the multiplier of material coefficients (e.g. density and specific heat) which are usually taken as constants within each element.

Figure 4.3 Quadrilateral's capacitance matrix calculation in Mathematica

The considerable potential of the Computer Algebra System (CAS), for example Mathematica and Maple, will be further demonstrated when constructing stiffness matrix for a general quadrilateral element. While a code similar to the one shown in Fig. 4.3 can be used in the derivation, the computation would likely never stop proceeding due to the rational Jacobian involved. In other words, the analytical expression for the fully integrated stiffness matrix of a quadrilateral four-node element does not seem to exist. Based on CAS and under the assumption of an isotropic elastic material, Reference [34] describes this stiffness matrix in a closed form by expanding and simplifying the terms in numerical integration summation. Since the CAS has a limited ability to simplify and factorize complex algebraic terms, one would have to further simply the expression produced by CAS in order to arrive at a suitable form for thesis writing or publication. To save effort on editing those expressions by hand, the algorithm in Reference [34] is used. However, to account for the anisotropy of reinforced concrete, the material matrix will be fully populated as shown in Eq. (4.2). Through an analysis done (shown in Fig. 4.4) on the material matrix, the computation algorithm for the stiffness matrix of a general quadrilateral element with a full material matrix is realized by a process of summation. Then, for heat analysis, the conductance matrix can be obtained if one specifies the single degree of freedom on each node and realizes that the material matrix is actually a scalar.

VecTor2 previously divided the quadrilateral element into two triangles, sharing the shortest diagonal as a common edge, analyzed them separately and then took an area average of the strains in each triangle. To alter this 'historical' way of analysis, the subroutine V2STIF is modified based on the developed closed-form stiffness matrix algorithm.

```
Delastic = {{E1, E2, 0}, {E2, E1, 0}, {0, 0, G}}; Dfull = {{C11, C12, C13}, {C12, C22, C23}, {C13, C23, C33}};
Dconnex = {{E1, E2, C13}, {E2, C22, C23}, {C13, C23, G}};
B = {{N1x, 0, N2x, 0, N3x, 0, N4x, 0}, {0, N1y, 0, N2y, 0, N3y, 0, N4y}, {N1y, N1x, N2y, N2x, N3y, N3x, N4y, N4x}};
BT = Transpose[B]; Ktemp = Delastic.B; K = BT.Ktemp; Print["Kelastic=", K // MatrixForm]; Ktemp = Dfull.B; K = BT.Ktemp;
Print["Kfull=", K // MatrixForm]; Ktemp = Dconnex.B; K = BT.Ktemp; Print["Kconnex=", K // MatrixForm];
```

$$
Kconnex = \begin{pmatrix}
N1x(E1\,N1x + C13\,N1y) + N1y(C13\,N1x + G\,N1y) & N1y(G\,N1x + C23\,N1y) + N1x(C13\,N1x + E2\,N1y) & N1x(E1\,N2x + C13\,N2y) + N1y(C13\,N2x + G\,N2y) \\
N1y(E2\,N1x + C23\,N1y) + N1x(C13\,N1x + G\,N1y) & N1y(C23\,N1x + C22\,N1y) + N1x(G\,N1x + C23\,N1y) & N1y(E2\,N2x + C23\,N2y) + N1x(C13\,N2x + G\,N2y) \\
(E1\,N1x + C13\,N1y)\,N2x + (C13\,N1x + G\,N1y)\,N2y & (C13\,N1x + E2\,N1y)\,N2x + (G\,N1x + C23\,N1y)\,N2y & N2x(E1\,N2x + C13\,N2y) + N2y(C13\,N2x + G\,N2y) \\
(C13\,N1x + G\,N1y)\,N2x + (E2\,N1x + C23\,N1y)\,N2y & (G\,N1x + C23\,N1y)\,N2x + (C23\,N1x + C22\,N1y)\,N2y & N2y(E2\,N2x + C23\,N2y) + N2x(C13\,N2x + G\,N2y) \\
(E1\,N1x + C13\,N1y)\,N3x + (C13\,N1x + G\,N1y)\,N3y & (C13\,N1x + E2\,N1y)\,N3x + (G\,N1x + C23\,N1y)\,N3y & (E1\,N2x + C13\,N2y)\,N3x + (C13\,N2x + G\,N2y)\,N3y \\
(C13\,N1x + G\,N1y)\,N3x + (E2\,N1x + C23\,N1y)\,N3y & (G\,N1x + C23\,N1y)\,N3x + (C23\,N1x + C22\,N1y)\,N3y & (C13\,N2x + G\,N2y)\,N3x + (E2\,N2x + C23\,N2y)\,N3y \\
(E1\,N1x + C13\,N1y)\,N4x + (C13\,N1x + G\,N1y)\,N4y & (C13\,N1x + E2\,N1y)\,N4x + (G\,N1x + C23\,N1y)\,N4y & (E1\,N2x + C13\,N2y)\,N4x + (C13\,N2x + G\,N2y)\,N4y \\
(C13\,N1x + G\,N1y)\,N4x + (E2\,N1x + C23\,N1y)\,N4y & (G\,N1x + C23\,N1y)\,N4x + (C23\,N1x + C22\,N1y)\,N4y & (C13\,N2x + G\,N2y)\,N4x + (E2\,N2x + C23\,N2y)\,N4y
\end{pmatrix}
$$

$$
\begin{pmatrix}
N1y\,(G\,N2x + C23\,N2y) + N1x\,(C13\,N2x + E2\,N2y) & N1x\,(E1\,N3x + C13\,N3y) + N1y\,(C13\,N3x + G\,N3y) \\
N1y\,(C23\,N2x + C22\,N2y) + N1x\,(G\,N2x + C23\,N2y) & N1y\,(E2\,N3x + C23\,N3y) + N1x\,(C13\,N3x + G\,N3y) \\
N2y\,(G\,N2x + C23\,N2y) + N2x\,(C13\,N2x + E2\,N2y) & N2x\,(E1\,N3x + C13\,N3y) + N2y\,(C13\,N3x + G\,N3y) \\
N2y\,(C23\,N2x + C22\,N2y) + N2x\,(G\,N2x + C23\,N2y) & N2y\,(E2\,N3x + C23\,N3y) + N2x\,(C13\,N3x + G\,N3y) \\
(C13\,N2x + E2\,N2y)\,N3x + (G\,N2x + C23\,N2y)\,N3y & N3x\,(E1\,N3x + C13\,N3y) + N3y\,(C13\,N3x + G\,N3y) \\
(G\,N2x + C23\,N2y)\,N3x + (C23\,N2x + C22\,N2y)\,N3y & N3y\,(E2\,N3x + C23\,N3y) + N3x\,(C13\,N3x + G\,N3y) \\
(C13\,N2x + E2\,N2y)\,N4x + (G\,N2x + C23\,N2y)\,N4y & (E1\,N3x + C13\,N3y)\,N4x + (C13\,N3x + G\,N3y)\,N4y \\
(G\,N2x + C23\,N2y)\,N4x + (C23\,N2x + C22\,N2y)\,N4y & (C13\,N3x + G\,N3y)\,N4x + (E2\,N3x + C23\,N3y)\,N4y
\end{pmatrix}
$$

$$
\begin{pmatrix}
N1y(G\,N3x + C23\,N3y) + N1x(C13\,N3x + E2\,N3y) & N1x(E1\,N4x + C13\,N4y) + N1y(C13\,N4x + G\,N4y) & N1y(G\,N4x + C23\,N4y) + N1x(C13\,N4x + E2\,N4y) \\
N1y(C23\,N3x + C22\,N3y) + N1x(G\,N3x + C23\,N3y) & N1y(E2\,N4x + C23\,N4y) + N1x(C13\,N4x + G\,N4y) & N1y(C23\,N4x + C22\,N4y) + N1x(G\,N4x + C23\,N4y) \\
N2y(G\,N3x + C23\,N3y) + N2x(C13\,N3x + E2\,N3y) & N2x(E1\,N4x + C13\,N4y) + N2y(C13\,N4x + G\,N4y) & N2y(G\,N4x + C23\,N4y) + N2x(C13\,N4x + E2\,N4y) \\
N2y(C23\,N3x + C22\,N3y) + N2x(G\,N3x + C23\,N3y) & N2y(E2\,N4x + C23\,N4y) + N2x(C13\,N4x + G\,N4y) & N2y(C23\,N4x + C22\,N4y) + N2x(G\,N4x + C23\,N4y) \\
N3y(G\,N3x + C23\,N3y) + N3x(C13\,N3x + E2\,N3y) & N3x(E1\,N4x + C13\,N4y) + N3y(C13\,N4x + G\,N4y) & N3y(G\,N4x + C23\,N4y) + N3x(C13\,N4x + E2\,N4y) \\
N3y(C23\,N3x + C22\,N3y) + N3x(G\,N3x + C23\,N3y) & N3y(E2\,N4x + C23\,N4y) + N3x(C13\,N4x + G\,N4y) & N3y(C23\,N4x + C22\,N4y) + N3x(G\,N4x + C23\,N4y) \\
(C13\,N3x + E2\,N3y)\,N4x + (G\,N3x + C23\,N3y)\,N4y & N4x(E1\,N4x + C13\,N4y) + N4y(C13\,N4x + G\,N4y) & N4y(G\,N4x + C23\,N4y) + N4x(C13\,N4x + E2\,N4y) \\
(G\,N3x + C23\,N3y)\,N4x + (C23\,N3x + C22\,N3y)\,N4y & N4y(E2\,N4x + C23\,N4y) + N4x(C13\,N4x + G\,N4y) & N4y(C23\,N4x + C22\,N4y) + N4x(G\,N4x + C23\,N4y)
\end{pmatrix}
$$

Note: **D**connex-**D**elastic will be the terms in summation process to obtain **D**full.

Figure 4.4 A material matrix analysis for the calculation of the stiffness matrix

4.4 Temperature Distribution Contour Plot in MATLAB

To display the calculation results in graphics, a MATLAB code is developed (as shown in Fig. 4.5) for contour plots of temperature distribution.

```
function feaPlotS(ndS, ndXY, elT, u, scl)
% This function plots the required temperature fields
% Input:
% ndS (nd Temp, ndStress or ndStrain):
% should be a column vector of values to be plotted
% ndXY: nodal coordinates
% elT: element topology
% u: nodal displacements
% scl: scale factor for deformation
hold on;

dXY = ndXY(:,2:3);
if nargin > 3
   for i=1:size(ndXY,1)
      dXY(i,1) = dXY(i,1) + scl * u(2*i -1);
      dXY(i,2) = dXY(i,2) + scl * u(2*i   );
   end
end

% following commands draw the contour plot
for e=1:size(elT,1)
   xy = [dXY(elT(e, 2:5), 1), dXY(elT(e, 2:5), 2)];
   patch(xy(:,1), xy(:,2), ndS(elT(e, 2:5)));
end

%[ms,n]=max(ndS);text(dXY(n,1),dXY(n,2),int2str(n),
'horizontalalignment','center','verticalalignment','middle');

axis equal;
colorbar;
hold off;
```

Figure 4.5 Contour plot of the temperature distribution in MATLAB

Chapter 5 Numerical Corroboration

To assess the performance of the computational scheme presented in Chapter 3 and the code implementation described in Chapter 4, this chapter will carry out four numerical corroborative problems. While different investigated problems are designed to fulfill different objectives, each of them consists of: problem description, computational models, and results and discussion.

5.1 Problem 1: Temperature Profiles

The purpose of Problem 1 is to verify the temperature profile throughout the depth of a cross section. As such, some simplifications are made that bypass the time-varying thermal loads and temperature-dependent thermal properties. Also, no internal heat resource is present in this problem.

- Problem description:

Fig. 5.1 shows the cross section of a long square bar, initially at zero temperature everywhere. A constant temperature $T = 100^{\circ}C$ is suddenly imposed on the upper surface, while temperatures on the remaining surfaces are held at $T = 0^{\circ}C$. The task is to compute the development of the temperature field within the bar's cross section until the final steady-state condition is reached.

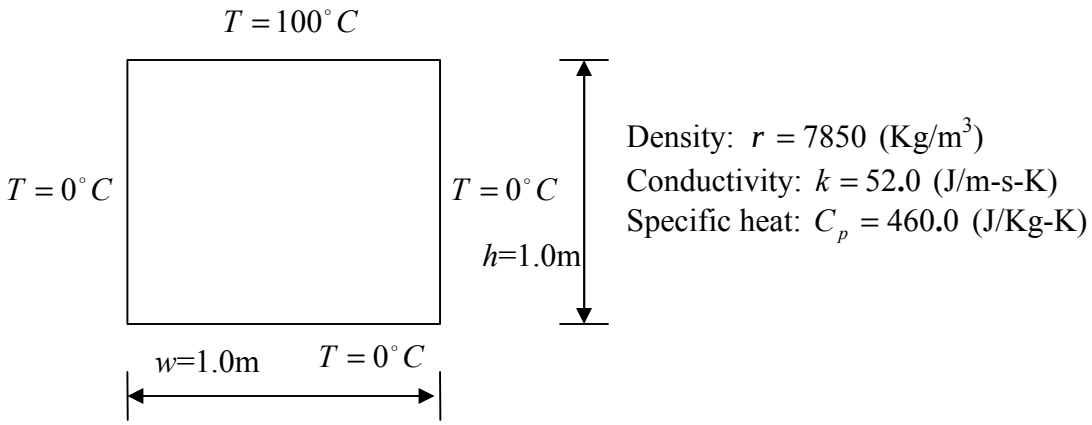


Figure 5.1 Numerical test problem 1

- Computational model:

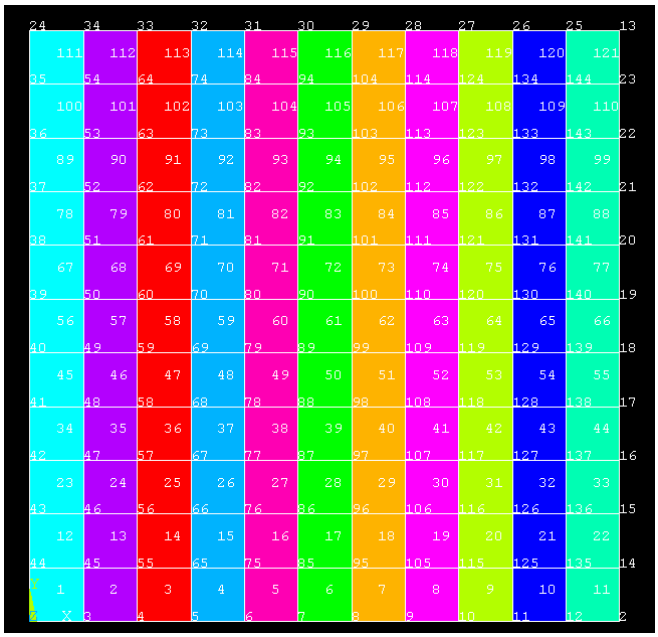


Figure 5.2 The finite element model for Problem 1

As shown in Fig. 5.2, a 11×11 uniform computational grid is employed so that there are $12 \times 12 = 144$ nodes within the model. The model will be analyzed through 100 time steps of 50s duration, with a maximum of 100 iterations allowed at each time step. The criterion for convergence of iterative computations is prescribed to be 1.0×10^{-5}. In addition, the

time stepping factor (i.e. r value in Eq. (3.26)) is set to be 0.5, which corresponds to the most accurate Crank-Nicholson scheme.

- Results and discussion:

The nodal temperature distribution within the structure is generated by the code. By examining the data on the vertical center line, the temperature profiles for both steady-state and transient analyses are obtained and plotted in Fig. 5.3. The graphic contour plots of temperature distribution, generated by a MATLAB code, are also given in Fig. 5.4 (a-e). Based on these results, some observations are discussed below.

1. Temperature profiles (or thermal gradients):

The transient thermal gradients are exceedingly nonlinear shortly after the thermal load is applied, while the steady-state analysis produces a fairly linear one. One can observe from Fig. 5.3 that the transient temperature profiles of listed durations (from 50s to 5000s) gradually approach the steady-state one as time advances. Thus, it can be expected that as time continues to proceed, there is one *ending* stage of transient thermal analyses when all transient effects have diminished and the corresponding temperature profile will be consistent with the one obtained from the steady-state analysis.

2. Thermal stress:

In this problem, continuity thermal stresses will not be induced because of the determinate nature of this problem from a structural viewpoint. However, due to the transient nonlinear thermal gradient, primary thermal stresses will be produced in the section. Nevertheless, the nonlinear thermal gradients approach the linear steady-state condition and, as such, primary thermal stresses in the section will diminish gradually as time advances.

3. Time stepping schemes:

As mentioned in Sections 2.3 and 3.2.6.2, the time stepping factor r can be specified by using either the Crack-Nicholson scheme for accuracy considerations or the Galerkin scheme for stability purposes. In both Fig. 5.3 and Table 5.1, some oscillatory errors are observed during the analyses over a number of starting time steps. However, it appears that such errors diminish as time proceeds. If stability of the solution is a strong concern in the very beginning stages of the analysis, the Galerkin time stepping scheme can be used instead since it produces less oscillatory error than the Crank-Nicholson scheme.

4. Boundary conditions:

The boundary condition that a zero temperature is held on both side edges of the section plays a key role in the resultant temperature distribution. A steady-state analysis on the section without prescribing the above conditions is tested, with the relevant temperature distribution given in Fig. 5.4 (f). It turns out that, without these conditions, the heat analysis can be actually reduced to the 1D case since the resultant temperature distribution is in a parallel-strip manner.

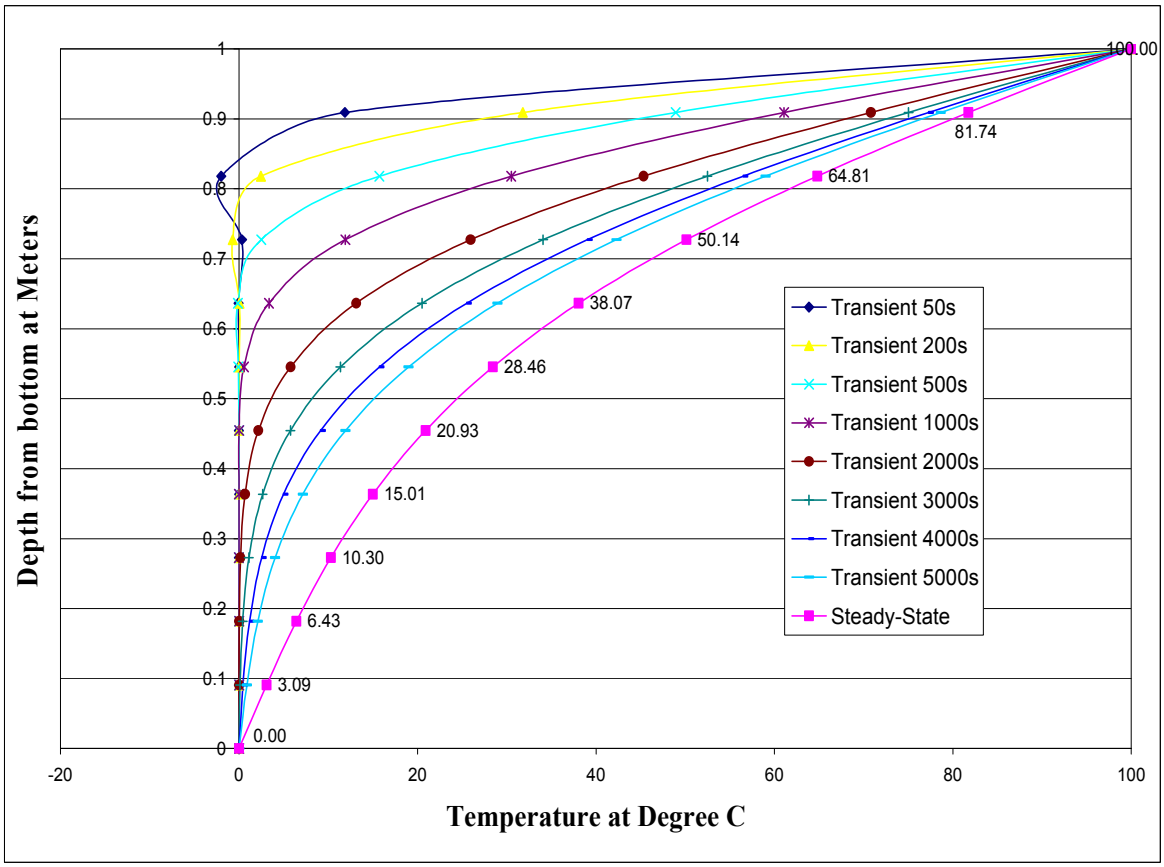
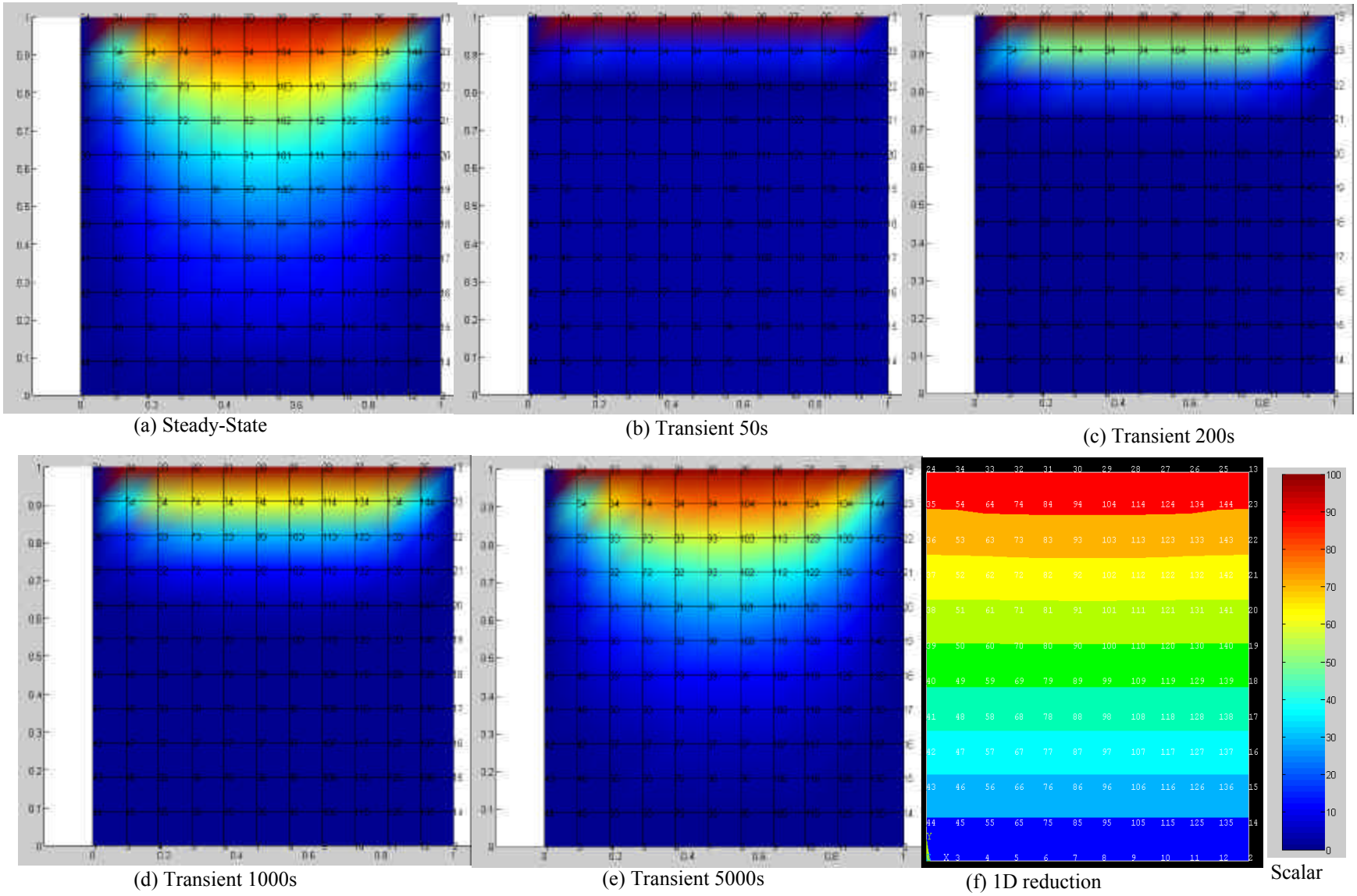


Figure 5.3 Temperature profiles through the depth of the cross section

Node No.	Depth (m)	Nodal Temperature at Degree C								
		Steady State	Transient							
			50s	200s	500s	1000s	2000s	3000s	4000s	5000s
30	1	100.0	1.00E+2	1.00E+2	1.00E+2	100.0	100.0	100.0	100.0	100.0
94	10/11	81.7	1.19E+1	3.18E+1	4.90E+1	61.1	70.8	75.0	77.3	78.7
93	9/11	64.8	-1.99E+0	2.44E+0	1.57E+1	30.5	45.4	52.5	56.5	59.0
92	8/11	50.1	3.35E-1	-6.98E-1	2.49E+0	11.9	26.0	34.1	39.1	42.3
91	7/11	38.1	-5.62E-2	4.93E-2	-1.15E-1	3.4	13.1	20.5	25.5	29.0
90	6/11	28.5	9.43E-3	1.34E-2	-7.32E-2	0.6	5.8	11.4	15.7	19.0
89	5/11	20.9	-1.58E-3	-6.04E-3	8.98E-3	0.0	2.2	5.8	9.2	11.9
88	4/11	15.0	2.66E-4	1.42E-3	1.81E-3	0.0	0.7	2.7	5.0	7.1
87	3/11	10.3	-4.45E-5	-2.24E-4	-7.10E-4	0.0	0.2	1.1	2.5	4.0
86	2/11	6.4	7.46E-6	1.41E-5	6.74E-5	0.0	0.0	0.4	1.2	2.1
85	1/11	3.1	-1.22E-6	5.28E-6	1.70E-5	0.0	0.0	0.1	0.5	0.9
7	0	0.0	0.00E+0	0.00E+0	0.00E+0	0.0	0.0	0.0	0.0	0.0

Data in shaded columns (over 50s, 200s and 500s) are in scientific numbers to show the oscillation involved.

Table 5.1 Nodal temperatures on the center line in Problem 1



(a) Steady-State (b) Transient 50s (c) Transient 200s

(d) Transient 1000s (e) Transient 5000s (f) 1D reduction

Figure 5.4 Temperature contour plots in Problem 1

5.2 Problem 2: Accuracy Comparisons

This problem is proposed to compare the accuracy of the results obtained from the developed code with those from ANSYS. While the code runs independently on VecTor2, some essential features of the computational scheme, such as multi-domain, variable-thermal-property, and quadrilateral-element, are involved in the analysis.

- Problem description:

This is a nonlinear transient heat transfer analysis of a simplified casting process (phase change in the solidification process is ignored), which is given in ANSYS Guide [35]. The objective is to track the temperature distribution in the steel casting and the L-shape sand mold, as shown in Fig. 5.5. While convection occurring between the sand mold and the ambient air is neglected, conduction occurring between the steel and the sand mold is analyzed. The material of the sand mold has constant material properties while the casting steel has temperature-dependent thermal conductivity. The detailed material properties are given in Table 5.2.

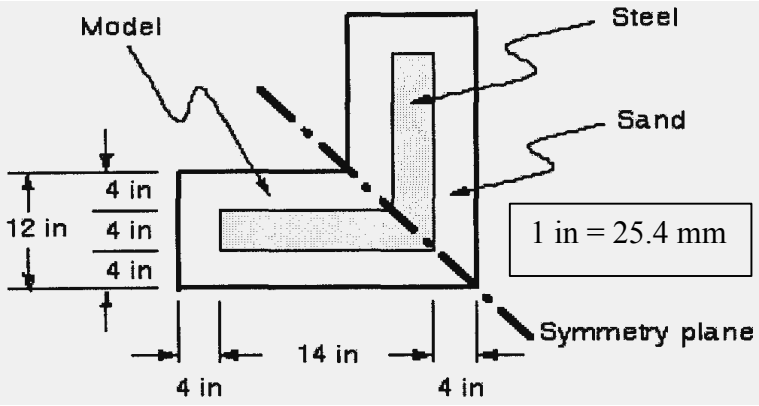


Figure 5.5 Numerical test problem 2

Item	U.S. Customary	S.I. Equivalent
Material properties for sand:		
Conductivity	0.025 $Btu/hr-in-^\circ F$	0.519 W/mK
Density	0.054 lb/in^3	1495 $kg\,m^{-3}$
Specific heat	0.28 $Btu/lb-^\circ F$	1172 J/kgK
Material properties for steel:		
Density:	0.25 lb/in^3	6920 $kg\,m^{-3}$
Specific heat:	0.11 $Btu/lb-^\circ F$	460.6 J/kgK
Conductivity: at $0^\circ F$ $(-17.8^\circ C)$	1.44 $Btu/hr-in-^\circ F$	29.9 W/mK
at $2643^\circ F$ $(1450^\circ C)$	1.54 $Btu/hr-in-^\circ F$	32.0 W/mK
at $2750^\circ F$ $(1510^\circ C)$	1.22 $Btu/hr-in-^\circ F$	25.3 W/mK
at $2875^\circ F$ $(1580^\circ C)$	1.20 $Btu/hr-in-^\circ F$	25.0 W/mK

Table 5.2 Material properties in Problem 2

- Computational model:

A 2D analysis of a one unit thick slice will be performed. Half symmetry is used to reduce the size of the model, in which the lower half is the portion modeled. As the computational mesh in Fig. 5.6 shows, quadrilateral elements are employed in this analysis. Since the initial conditions are the starting point for a transient thermal analysis, one needs to specify both the initial temperatures of the steel casting ($2875^\circ F$) and the sand mold's ambient temperature ($80^\circ F$). In addition, to make the final equation system non-singular, the boundary condition is added in such a way that the temperatures on the four external corners are held at $80^\circ F$. The final time in this analysis is set at 3 hours, with the interval equal to 0.01 hour and the time stepping factor equal to 2/3 which corresponds to the Galerkin method instead of the Crank-Nicholson method tested in Problem 1.

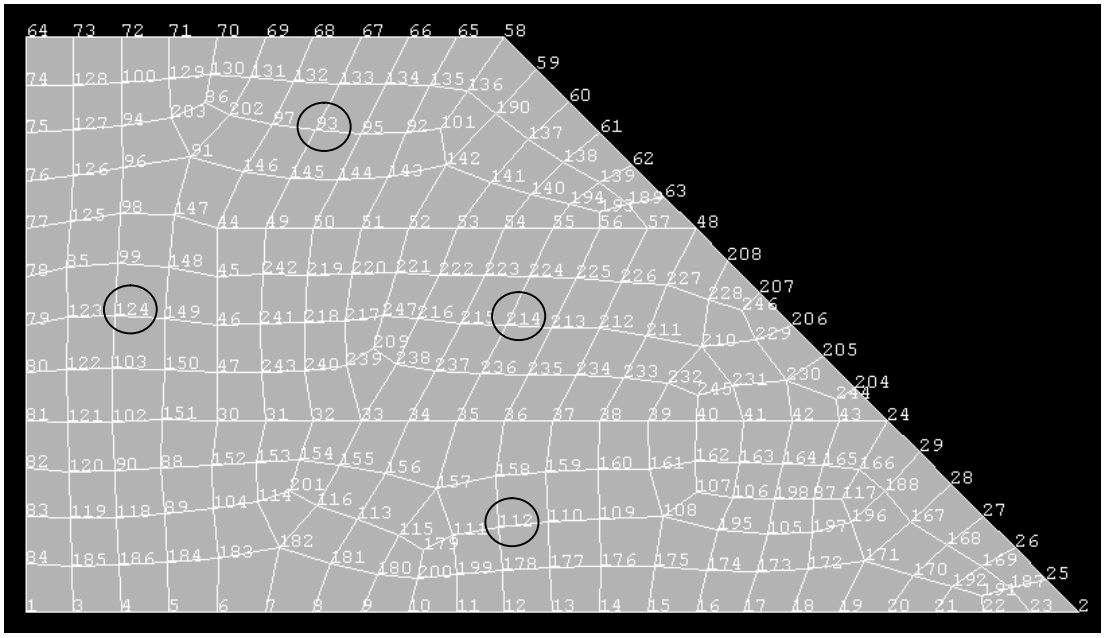


Figure 5.6 The finite element model for Problem 2

- Results and discussion:

While the whole set of nodal temperature data can be directly obtained from the program's output files, due to space limits only selected time steps' results are included, which correspond to time steps equal to 0.25h, 0.5h, 1.0h, 2.0h and 3.0h. For the purpose of comparison, the exact same mesh system (including numbering of nodes and irregular quadrilateral elements) will be used in analyses with both ANSYS and the developed code V2HEAT. Also, an identical time advancing size is used for the same reason. Both a data-list of the nodal temperatures and a contour-plot of its distribution are provided. Some discussion is given below based on the results.

1. Contour plots:

Contour plots of the temperature distribution for both analyses are shown in Fig. 5.7. Due to technique difficulties in programming, the nodal numbering system remains in contour

plots for V2HEAT. Also, the temperature scale shown is lightly different between each other. In spite of these display problems, one can readily see their similarity or closeness in the heat conduction trend. It is likely both of them are capable of capturing the heat conduction involved in this problem.

2. Nodal temperature listings:

In order to examine the difference between the two analyses more precisely, some investigation of local representative (circled in Fig. 5.6) nodal temperatures are given in Table 5.3 for statistic comparison. In addition, graphically compared in Fig. 5.8 are the time-history temperature curves of the model's center (node 214). From both comparisons, V2HEAT is believed to be very similar to ANSYS in accuracy.

3. Imperial units:

Imperial units are used in both analyses. It follows that V2HEAT does not have the restriction on units provided all parameters are consistent in units.

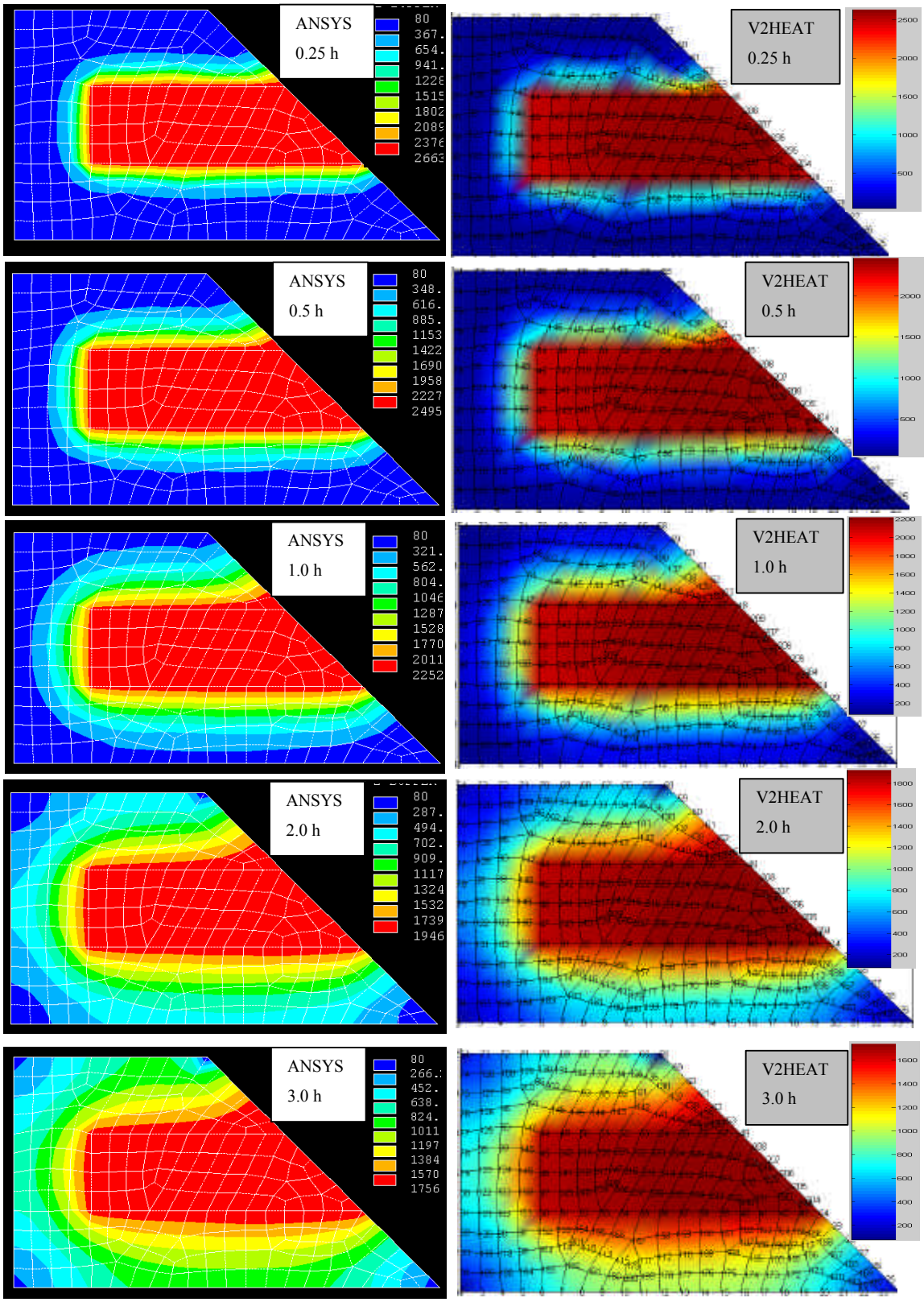


Figure 5.7 Temperature contour plots in Problem 2

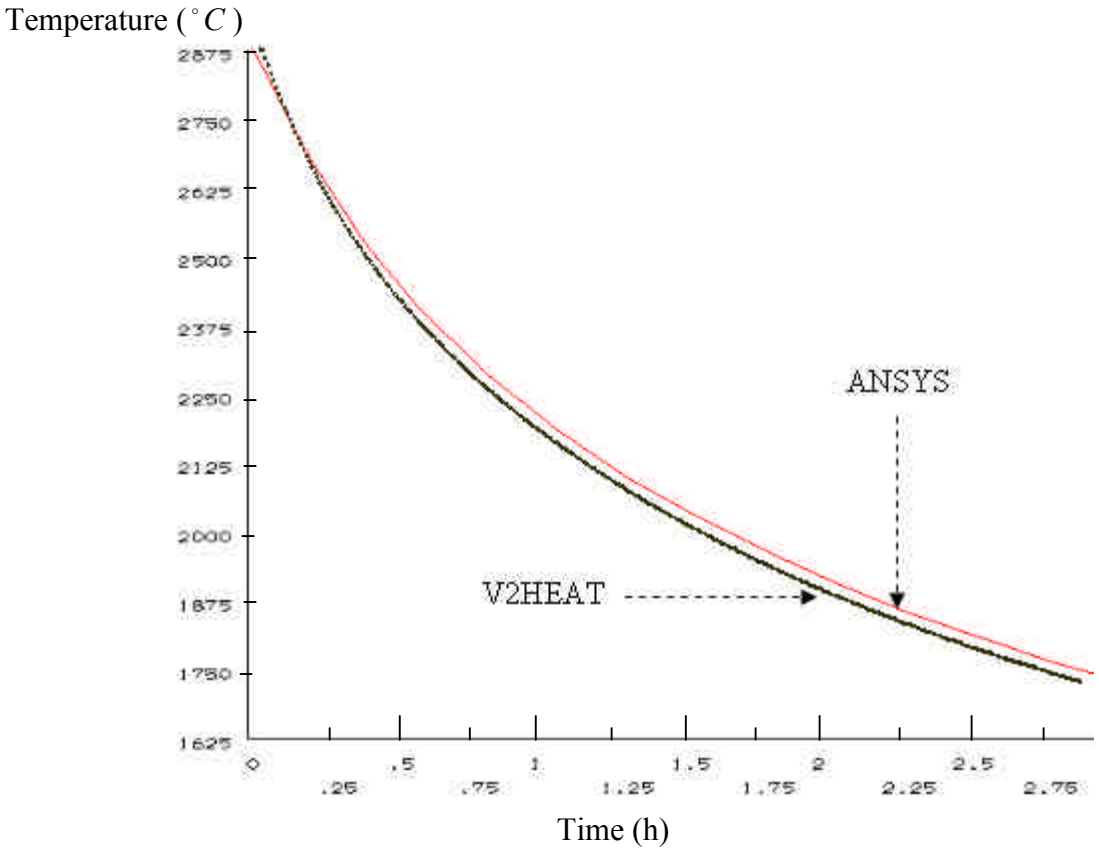


Figure 5.8 Time-history temperature curves of the model's center

Rep. Node Number	T(V2HEAT)/T(ANSYS)				
	0.25 h	0.5 h	1.0 h	2.0 h	3.0 h
214	2652.5/2512.2 =1.056	2469.6/2429.0 =1.017	2222.5/2189.7 =1.015	1924.8/1901.1 =1.012	1740.4/1721.7 =1.011
93	201.42/202.38 =0.995	388.92/456.30 =0.852	674.45/736.84 =0.915	954.07/985.18 =0.969	1086.7/1109.2 =0.980
124	187.52/178.31 =1.052	355.19/410.29 =0.866	613.98/669.21 =0.918	863.65/889.17 =0.971	979.65/995.42 =0.984
112	181.69/161.43 =1.125	349.68/396.52 =0.882	635.30/697.88 =0.910	966.71/1007.2 =0.960	1146.1/1176.3 =0.975
Mean	1.058	0.904	0.940	0.978	0.988
Mean of COV#(%)	(4.356+7.296+4.647+2.051+1.411)/5=3.95				

COV#: coefficient of the variation defined as percentage of ratio (standard deviation/mean).

Table 5.3 Statistical comparisons in Problem 2

5.3 Problem 3: Various Thermal Loads

Unlike Problems 1 and 2, where the code runs independently of VecTor2, this problem requires a structural analysis be performed after the heat analysis capacity is implemented through the incorporated subroutine V2HEAT. Also, the modified V2TRED will allow one to take mechanical properties' temperature dependency into account when determining a section's response to various thermal loads. Taking advantage of VecTor2's built-in realistic constitutive models, one can then expect some structural response to the applied time-varying thermal loads.

- Problem description:

The simply-supported reinforced concrete beam considered is shown in Fig. 5.9, with details of the material properties given in Table 5.4. In this example, two reinforced concrete material types are used. One type models the plain concrete comprising the flange, while the other models the web region of the beam with one smeared reinforcement component representing the stirrup reinforcement. Also, two ductile steel reinforcement material types are utilized to model the longitudinal steel bars. The beam is subjected to gravity load in addition to thermal loads which simulate fire underneath. The analyses are expected to determine both the internal (e.g. crack pattern) and external (e.g. deflection) responses of the beam at intermediate stages and at the conclusion of thermal loading.

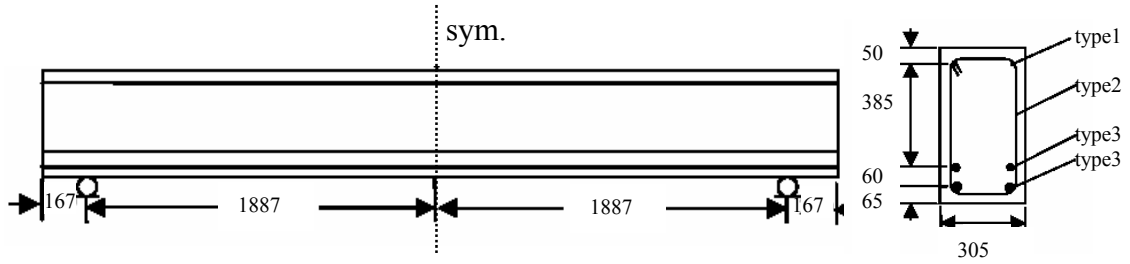


Figure 5.9 Numerical test problem 3

Material properties of concrete				
f_c' (MPa)	f_t' (MPa)	e_c' $(\times 10^{-3})$	E_c (MPa)	a_c $(\times 10^{-6}/^\circ C)$
24.1	1.88	2.00	24100	9.0

Material properties of reinforcement								
Type	f (mm)	A_s mm^2	f_y (MPa)	f_u (MPa)	E_s (MPa)	E_{sh} (MPa)	e_{sh} $(\times 10^{-3})$	a_s $(\times 10^{-6}/^\circ C)$
1	13	253	345	700	200000	2000	5	11.5
2	7.5	$r = 0.099\%$	325	600	200000	2000	5	-
3	29	1282	555	900	200000	2000	5	11.5

Table 5.4 Material details in Problem 3

- Computational models:

As both the structural and loading conditions are symmetrical about the mid-span, only one half of the beam needs to be modeled. Within the model shown in Fig. 5.10, nodes along the symmetrical line are restrained from displacements in the longitudinal direction, and the node at the support is restrained in the transverse direction. While the concrete is modeled by rectangular elements, truss bar elements are used for longitudinal reinforcing bars. All this structural information is prescribed through a *structure* input file, one of three primary kinds of files from which VecTor2 reads input information.

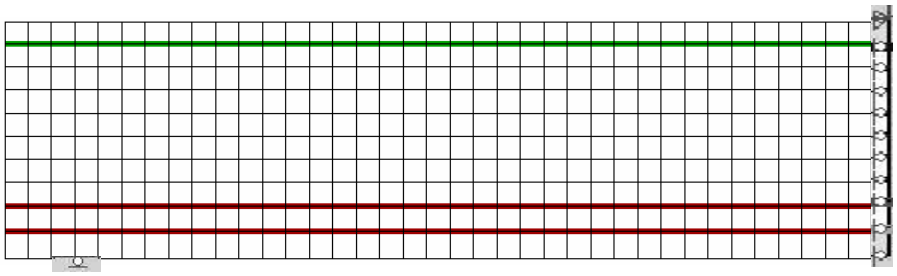


Figure 5.10 The finite element model for Problem 3

The model will be analyzed through 54 time steps of 600s duration (9 hours in total), with a value of 0.5 for time stepping factor *r* and a maximum of 100 iterations allowed at each time step. In VecTor2, time steps proceed through an accumulation regime of the load stage defined in the *job* input file. For the current case, there are 54 load stages with the load increment of 600s. VecTor2 defines various types of thermal loads by prescribing the entry fields of Nodal Temperature Loads in the *load* input file, as provided in Table 5.5. The different fire loads provided are not intended to accurately simulate the fire in reality, but rather to illustrate some possible nodal temperature loads.

While the various thermal loads tested are plotted in Fig. 5.11, listed in Appendix B are the aforementioned VecTor2's input files of this example. Except for the particular thermal load file, all remaining files (job and structure files) are exactly same for all tests of thermal loads.

- Results and discussion:

From the analyses with the various thermal loads, some results and discussion are provided below. For the purpose of visual presentation, the software Augustus© is used to represent the output files from VecTor2 analyses.

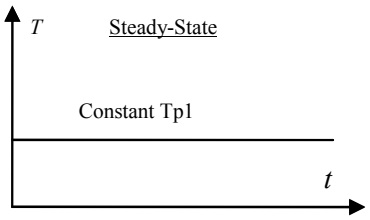
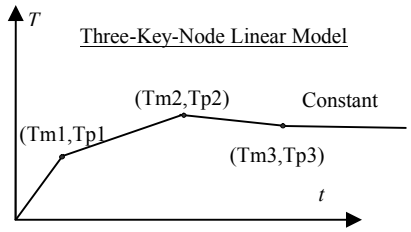
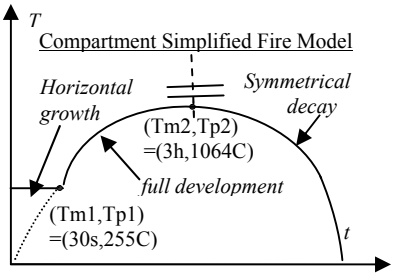
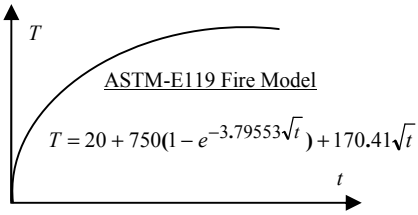
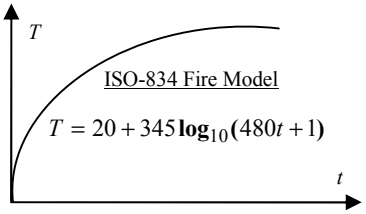
Load Type	Entry Fields in Load File								Load Plots
	Node	Type	Tm1	Tp1	Tm2	Tp2	Tm3	Tp3	
1	+	1		+					
2	+	2	+	+	+	+	+	+	
3	+	3	+	+	+	+			
4	+	4							
5	+	5							
Notes:	1. "+" indicates the field that is used in load definition; 2. Load types 2-5 are transient analyses while type 1 is steady-state one. 3. Full development curve in load type 3 is based on ASTM-E119 during the range of 255C-1064C; and the decay curve is symmetrical with development one about the peak point. 4. Standard fire curves in load types 4 and 5 have no ends.								

Table 5.5 Various fire models in VecTor2

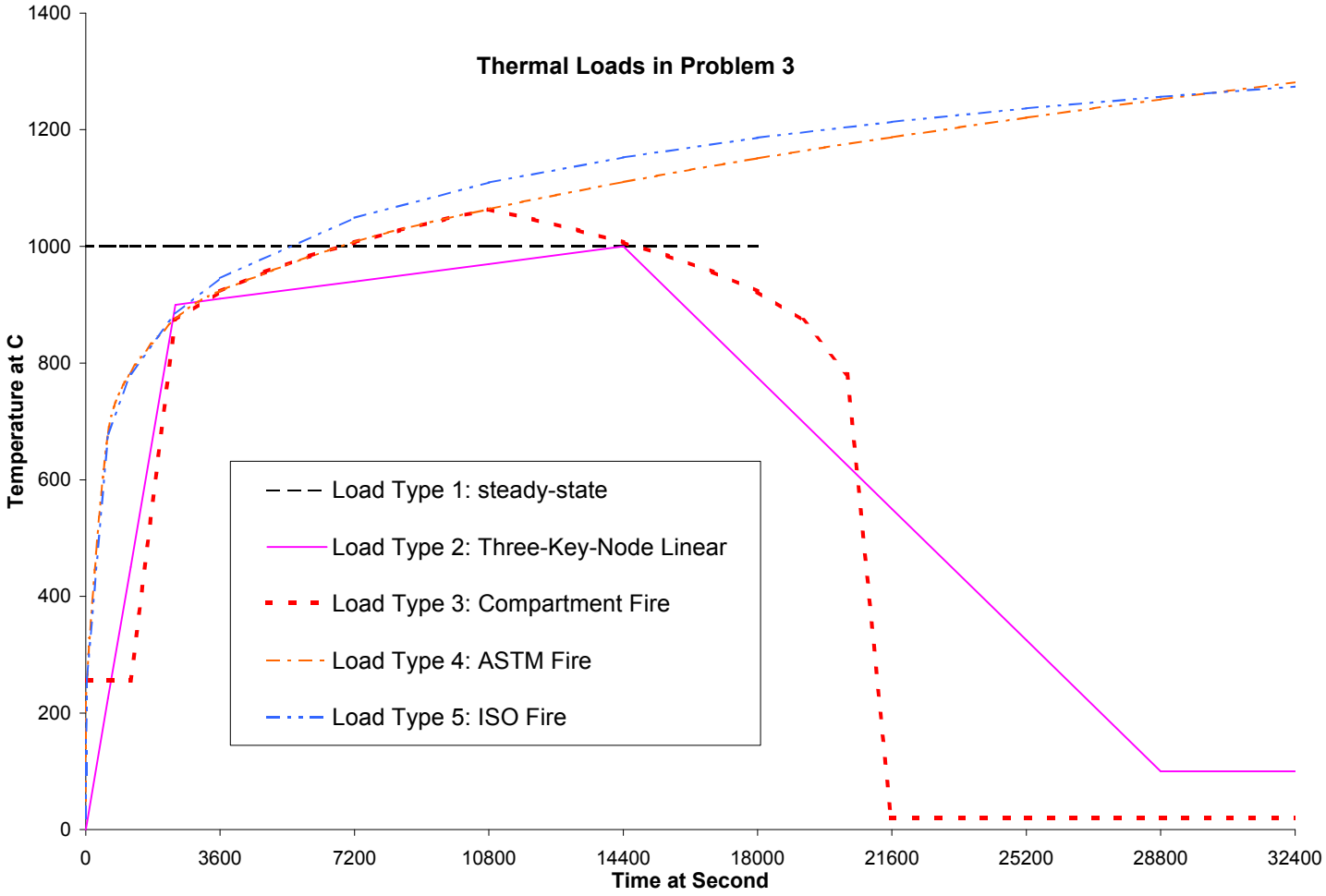


Figure 5.11 Various thermal loads tested in Problem 3

1. Load type 1: steady-state model

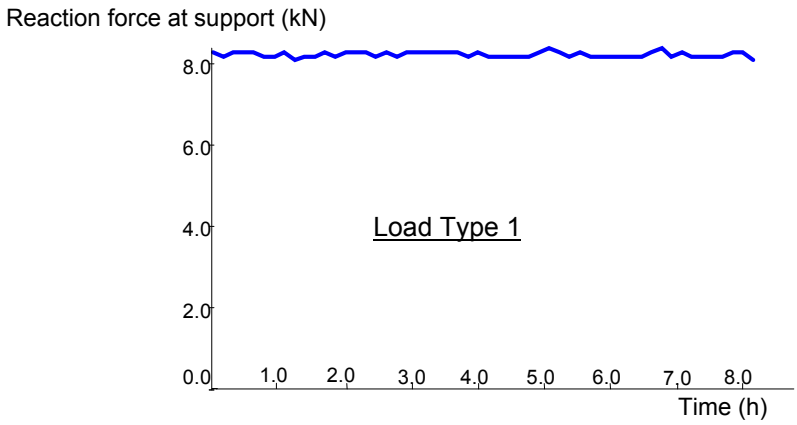


Figure 5.12 Reaction forces at support

Since the beam is statically determinate, no restrained force will result from thermal gradients, regardless of the fire load type. As a result, the reaction force at support in the above figure stays fairly constant, corresponding to the gravity load:

$$2400\times10^{-3}\times9.8\times(305\times560\times2053.5)\times10^{-9}=8.25\text{ kN}$$

For the deflection in the steady-state condition, due to the linear (in theory) thermal strain, the curvature is constant throughout the height of the beam, being equal to:

$$f=a_c\Delta T/h \tag{5.1}$$

where $a_c=f\times a_{c,20}=2.035118\times9.0\times10^{-6}$ and thermal gradient is equal to $1000^\circ C$.

Then, one can use first moment area theorem to calculate the deflection from the curvature diagram as follows:

$$\Delta^{T}_{mid}=f\times\frac{L}{2}\times\frac{L}{4}=\frac{a_c\Delta TL^2}{8h}=\frac{9\times10^{-6}\times2.035118\times1000\times(2053.5\times2)^2}{8\times560}=68.96\text{mm} \tag{5.2}$$

For the deflection due to the gravity load (treaded as a point load at the centroid of the beam), one can calculate the deflection by formula as follows:

$$\Delta_{\text{mid}}^{\text{G}} = \frac{W \times L^3}{48 E_c I_c} = \frac{(2400 \times 9.8 \times 0.305 \times 0.56 \times 4.107) \times (4.107)^3}{48 \times (24100 \times 10^6) \times (0.305 \times 0.56^3 / 12)} = 0.22\text{mm} \quad (5.3)$$

It is observed that the values calculated above (totally $68.96 + 0.22 = 69.2$mm due to both thermal and gravity loads) are very close to the results obtained from VecTor2 (67.9mm), shown in Fig. 5.13 (load type 1) below. The difference is likely due to $a_s < a_c$, causing some internal restraint.

2. Load type 2: three-key-node linear model

During the test of load type 2, the deflection-time curve given in Fig. 5.13 (load type 2) presents a similar overall trend as thermal gradient imposed while there is a lag in time observed between the turning points in plots of the imposed thermal gradient and the deflection curve. This lag is thought to be due to that in the plot of temperature load, the thermal gradient after the turning point is still very high and keeps contributing to the increase of the deflection until a lower stage is reached where the imposed temperature gradient is not sufficient to increase the deflection. This phenomenon is understandable if one considers the case where the transient constant temperature load is imposed but the deflection is increasing as time proceeds.

Also, the net strain (total strain minus the thermal strain) in the concrete at the location of top mid-span of the beam is shown in Fig. 5.14. The strain approaches to zero at the conclusion of the test when the thermal gradient approaching linear steady-state condition.

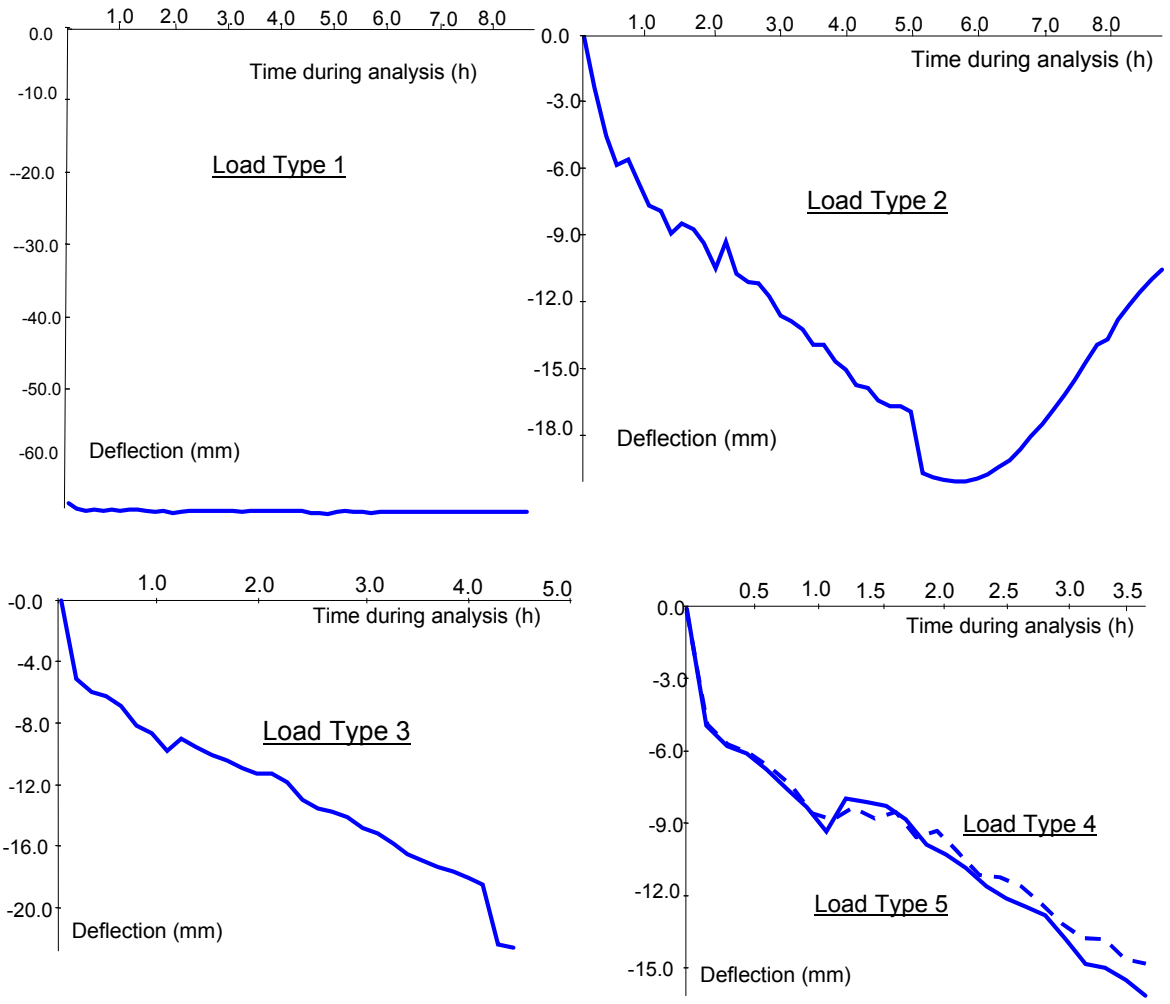


Figure 5.13 Deflection-time curves at beam's mid-span in Problem 3

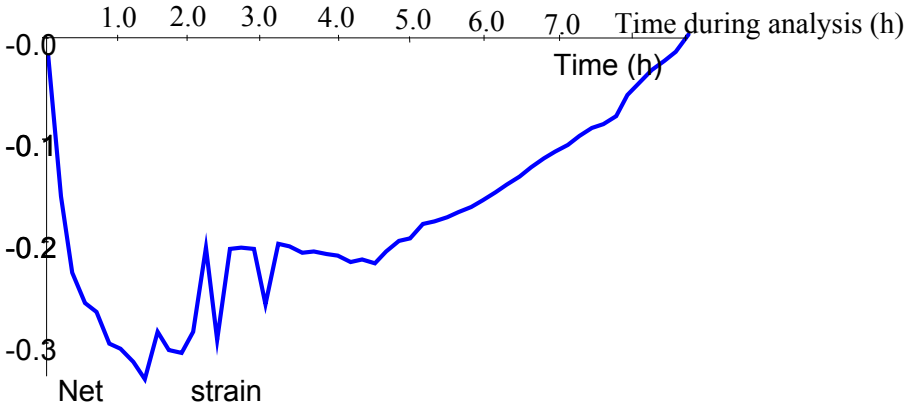


Figure 5.14 Net strains at top mid-span in the test of load type 2

3. Load type 3: compartment fire model

In the deflection curve given in Fig. 5.13 (load type 3), the expected phenomenon of a similar trend as the thermal gradient does not occur in the test of load type 3. It is likely that the structure failed at about 5 hours, when the thermal load is still very high (over $900^{\circ}C$), and hence deflections increase further. In fact, as shown by Augustus, the stiffness in the bottom steel bars is completely lost due to high temperatures there (over $1100^{\circ}C$). The crack width at this moment is around 10 mm. The strain in the stirrup around the left corner is also quite high (around 5%, as shown in Fig. 5.15), which causes the stress in the stirrups to go well beyond the yield stress.

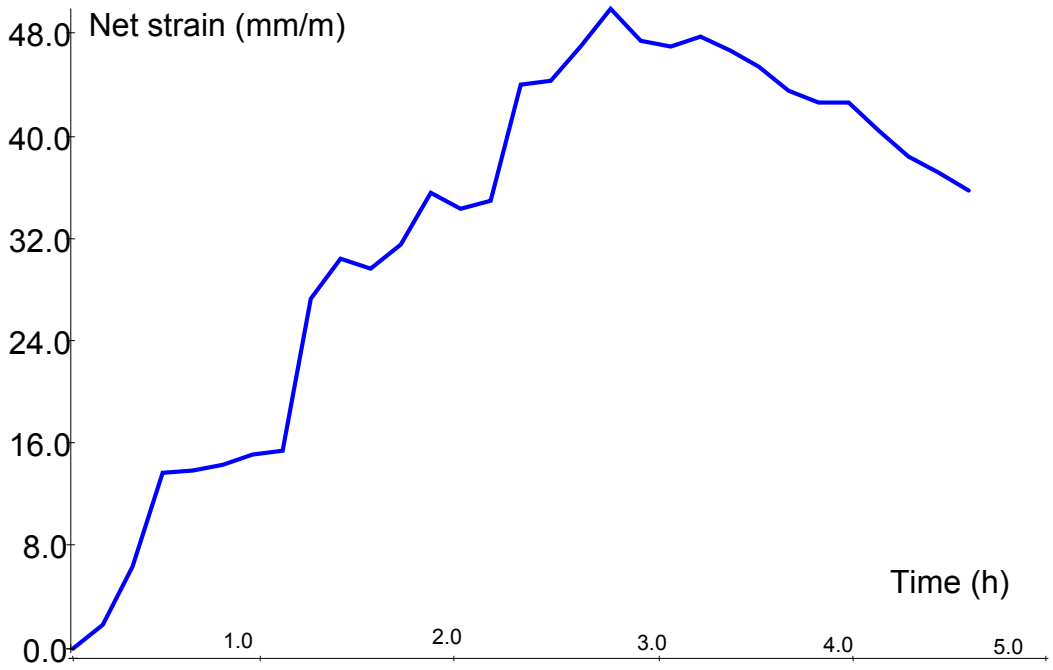


Figure 5.15 Net strains in stirrups in the test of load type 3

4. Load types 4 and 5: standard fire curves

The downward deflections obtained at the center of the beam for both load types 4 and 5 are shown in Fig. 5.13 (load types 4 and 5). They are very close to each other due to both fire models (in terms of thermal gradients imposed) being very similar. Also, the program stops further calculations at about 3.67 hours when the stiffness of steel bars is lost in the bottom cracked concrete region, which produces a structural failure mechanism. The crack width at that time is also over 10 mm and the stress in the stirrups is well beyond the yield stress, as shown in Fig. 5.16.

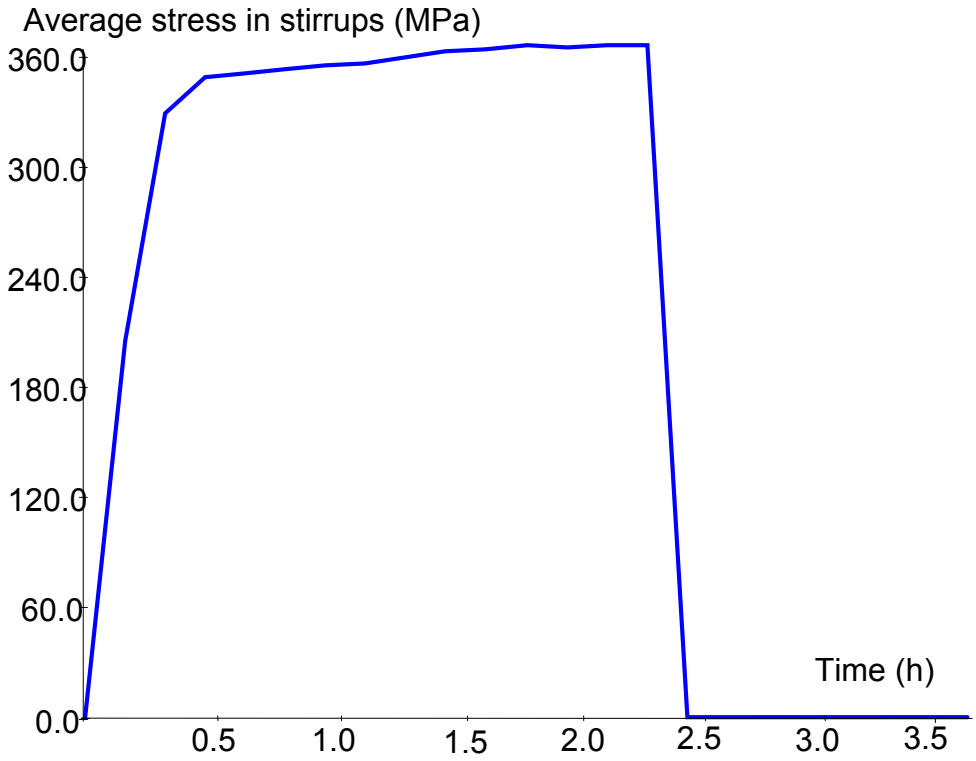


Figure 5.16 Average stresses in stirrups in the test of load type 4

During analyses, with the variation of thermal loads, the beam's crack development changes at different load stages (i.e. time). Although the deformed shapes and crack

patterns are available for each load stage, the one with the maximum crack width on the control chart in Augustus is chosen and shown in Fig. 5.17 for load types 3, 4 and 5. It is observed that they are slightly different, which is thought to be due to all of them occurring at very early load stages (numbers can be found in Fig. 5.17), where the thermal gradients of all loads are not much different from one another as given from Fig. 5.11.

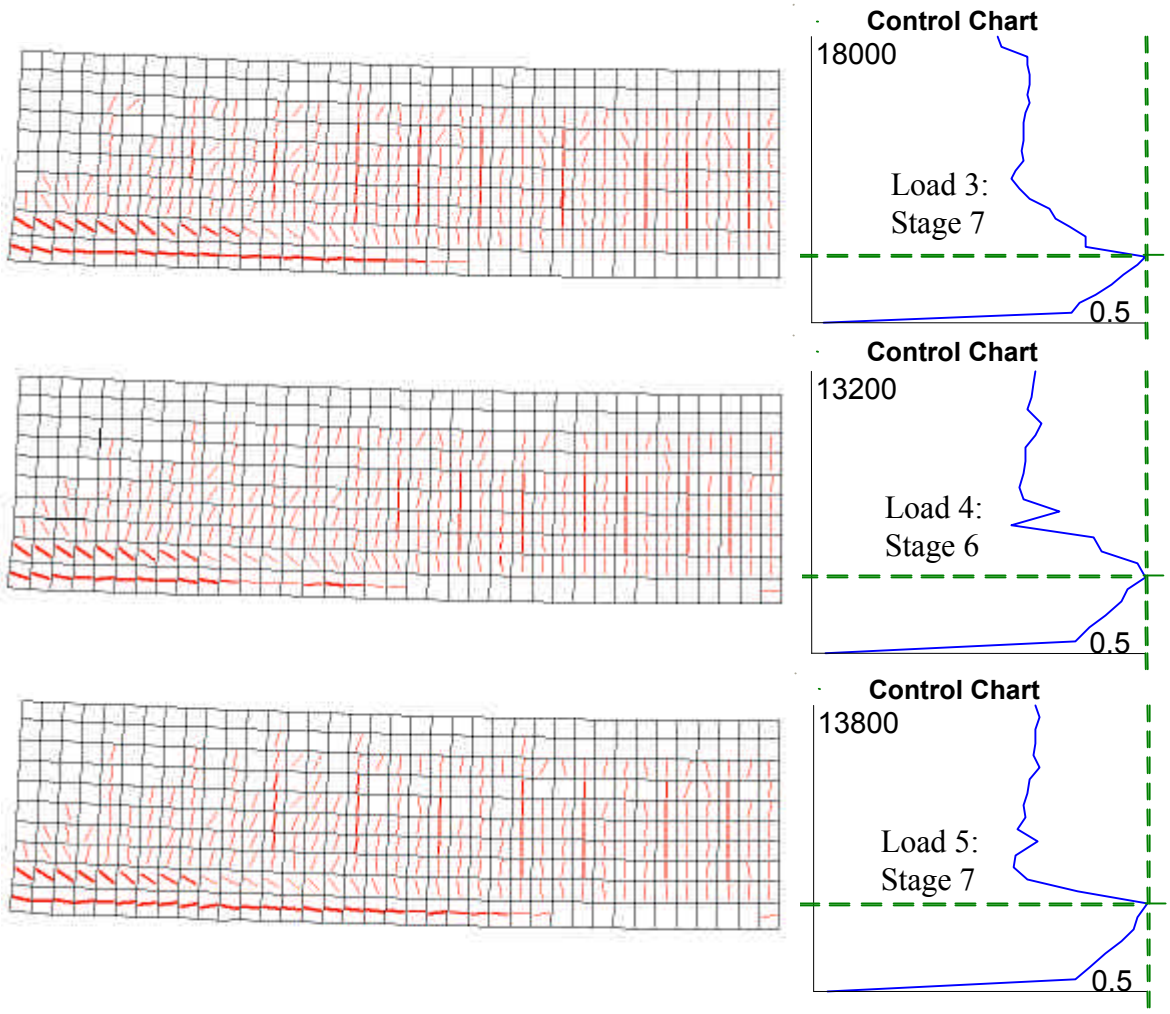


Figure 5.17 Crack patterns with control plots from the Augustus

5. Extended to the one-way slab: a restrained structure

By simply modifying the restraint system on the model, one can test the same thermal loads on a one-way slab, with the left end treated as a continuous beam in this case. This test is not intended to really analyze the behavior of the slab with an appropriate dimension but to see the characteristic of the response of an indeterminate structure to the thermal loads. The load type 2 is selected in this test.

The deflection-time curve obtained at the center of the slab is given in Fig. 5.18. It can be seen that it again presents the similar overall trend as the imposed temperature load plotted in Fig. 3.11. The sharp fluctuations in the curve are found to be due to severe cracking progression within the load stage. As a result, a smaller load incremental (i.e. time stepping size) is required to produce a smoother curve. That actually proves the fact that the high-nonlinear transient heat analysis is computationally intensive.

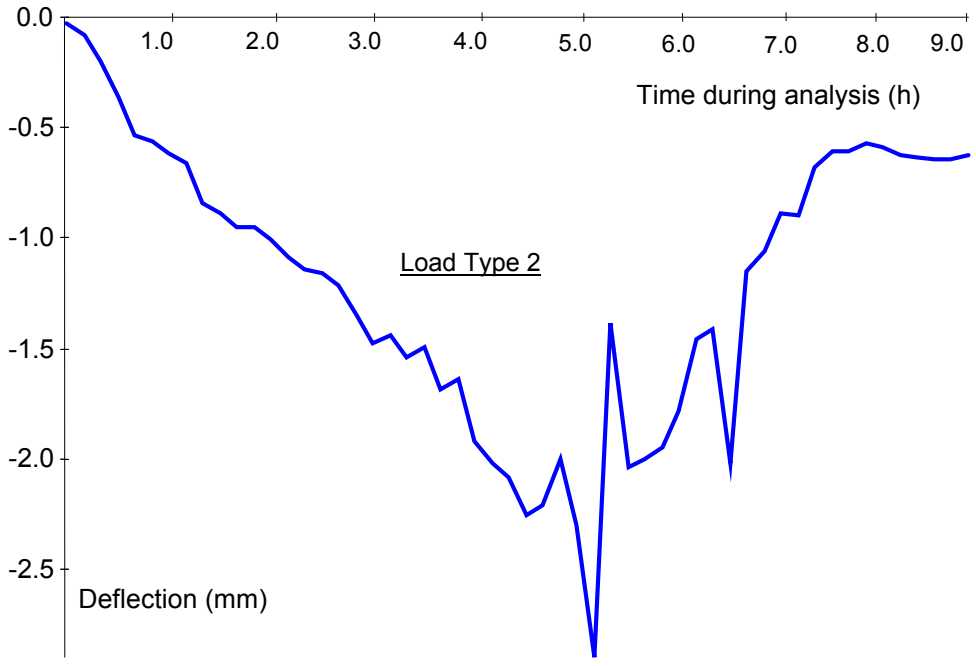


Figure 5.18 Deflection at the center of the slab tested

5.4 Problem 4: A Real Test

This problem is modeled after a specimen tested by Vecchio and Sato [32] at the facilities of Ontario Hydro over 15 years ago. Unlike the original experimental program which covered a diverse range of thermal and mechanical loading conditions and was used to calibrate and validate their proposed theoretical formulations, this problem examines only three models to test the overall performance of the modified VecTor2.

- Problem description:

The test speciman is, overall, a reinforced concrete portal frame consisting of two columns and one beam. The schematic representation of the test model is shown in Fig. 5.19, with specimen details and material properties given in Table 5.6. Two side panels span the interior of the frame to form a tank-like structure, with a flexible silicone water-stop in the gaps between the panels and the frame allowing the frame to be structurally independent of the panels. Water placed in the tank then serves to apply thermal loads, by means of an immersed heater.

Specimen details[1]			Material properties					
			Concrete[3]			Reinforcement		
b	(mm)	800	f'_c	(MPa)	42.4	f_y	(MPa)	448
h	(mm)	300						
A_s'	(-)	4#20M[2]	f_{cr}	(MPa)	3.12	f_u	(MPa)	710
d'	(mm)	55						
A_s	(-)	4#20M	E_c	(MPa)	58980	E_c	(MPa)	217000
d	(mm)	55						
A_v	(-)	#10M	α_c	$(/^\circ C)$	9.86×10^{-6}	α_s	$(/^\circ C)$	12.4×10^{-6}
s	(mm)	150						

1: Reinforcement in both columns and the beam is identical;
2: Area of #10M is $100\,mm^2$; #20M is $300\,mm^2$;
3: Thermal diffusivity is taken as the measured value $0.774 mm^2/s$.

Table 5.6 Specimen details and material properties in Problem 3

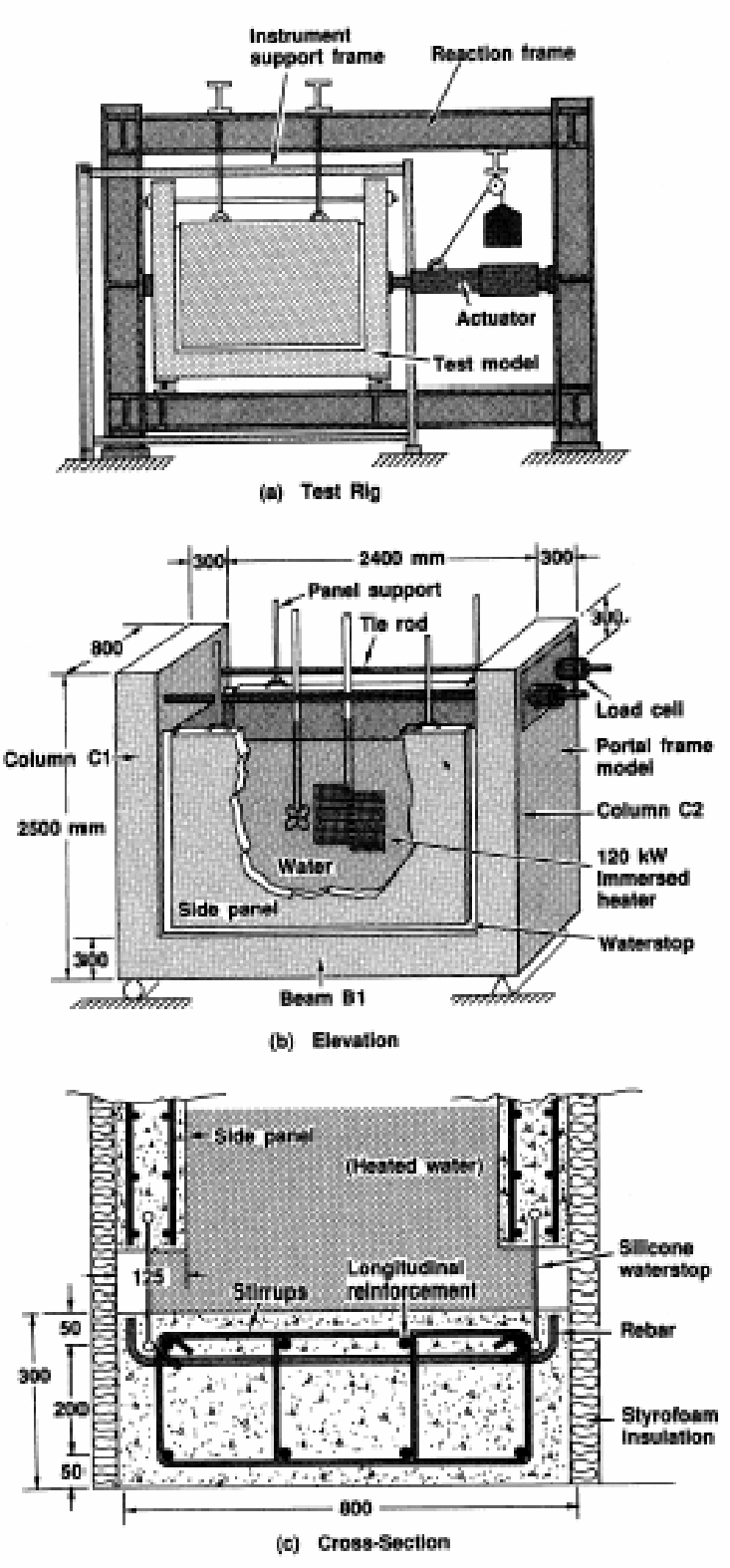


Figure 5.19 Numerical test problem 4 (From reference [32])

- Computational model:

The computational model is shown in Fig. 5.20 (d), in which only half of the structure is modeled due to symmetry.

Three distinct types of tests will be considered. The Type I test (Fig. 5.20 (a)) is conducted with the test model in an unrestrained mode. The temperature load Type I will be maintained for a long period (10 hours) so that both transient and final steady-state conditions can be observed. In the Type II test (Fig. 5.20 (b)), the tie-rod is engaged to render the structure one-degree statically indeterminate. Thus, with the columns restrained from outward deflection by the tie-rod, restraint forces are induced in the tie-rod and hence in the frame. The temperature load for the Type II test will be applied about for 10 hours to disclose the internal crack pattern developed and external restraint forces produced. In the Type III test (Fig. 5.20 (c)), the model is in the unrestrained configuration while a simultaneously acting mechanical load is applied laterally to the column at a location 800 mm above the bottom of the beam. The temperature load Type III will be applied until the ultimate capacity of the structure is attained under the monotonically increased (from zero) mechanical loads.

Instead of focusing on thermal "shock" tests, as in the original experimental program, the thermal loads employed in this problem consist of different increase amounts at varous rates for each test type. The temperature on faces not exposed to water is maintained at the room temperature $15^{\circ}C$. A plot of thermal loads is given in Fig. 5.21.

All VecTor2 input files used in this problem are listed in Appendix C.

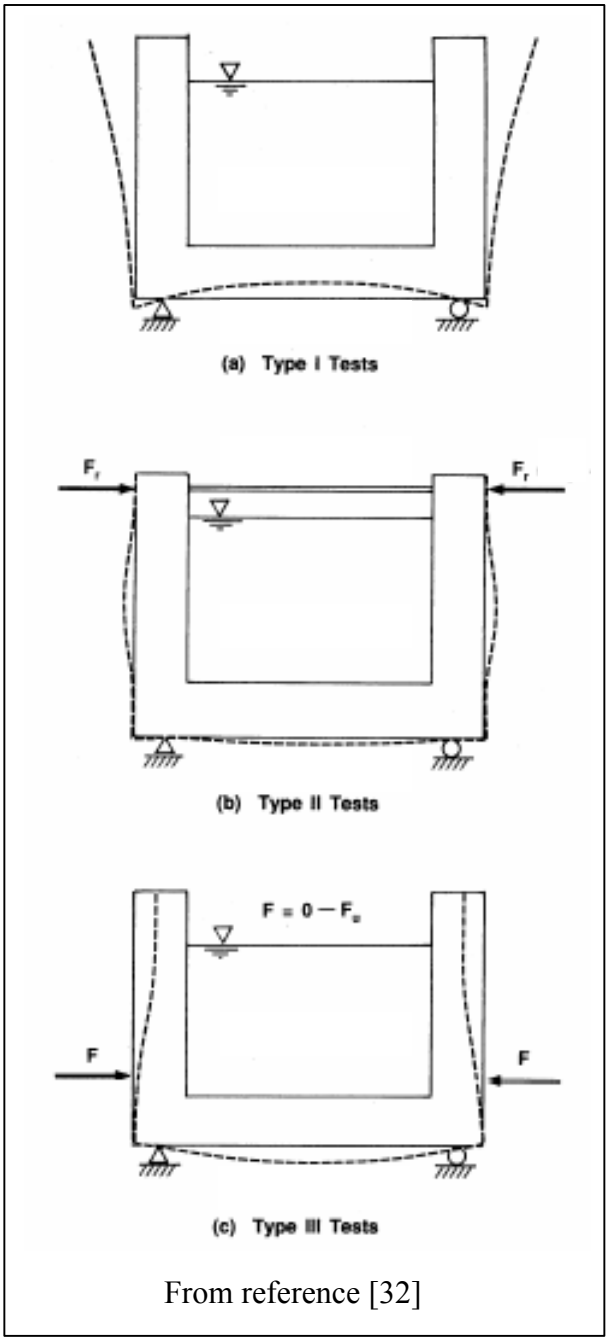


Figure 5.20 Computational models in Problem 4

- Results and discussion:

The results obtained from analyses are discussed below, with different focuses for each type of the test.

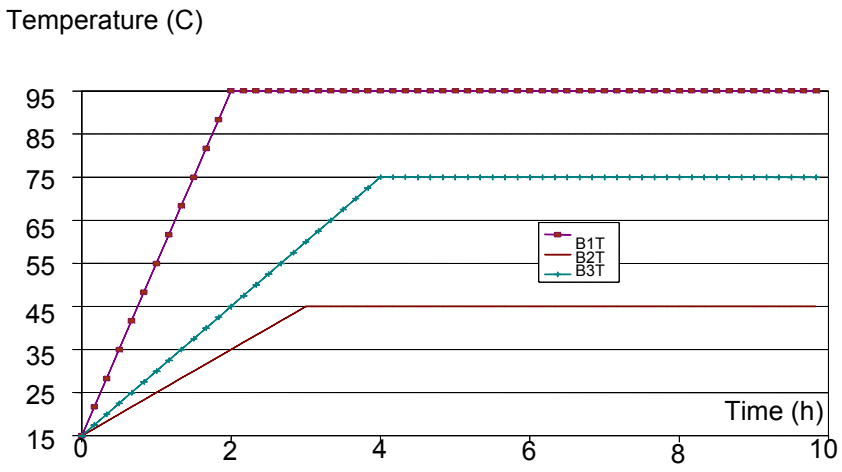


Figure 5.21 Thermal loads tested in Problem 4

1. Type I test:

During Type I testing, highly nonlinear transient thermal gradients are produced within the members in the beginning. In time, they approach a steady-state condition, characterized by a fairly linear gradient thought the depth of the section. The imposed thermal loads result in an upward deflection of the beam relative to its ends and outward deflections of the column relative to its base. The vertical deflection occurring at the mid-span of the beam and the lateral deflection at the top of the column are shown in Fig. 5.22. Both of them are thought to match the trend of the thermal load applied, increasing linearly in the beginning and becoming constant in the end. The deflection at beam's mid-span increases from 2.58 mm to 2.68 mm and the one at the column's top increases from 14.64 mm to 15.43 mm during the time range of 4-10 hours. That indicates the heat flow approaches the steady-state condition under which the deflection will stay unchanged in theory.

Primary thermal stresses are induced in the test model, mainly due to nonlinearity in the thermal gradients shortly after the thermal loads are applied (also from differences in

thermal expansion coefficients between concrete and reinforcement). However, this stress diminishes as the thermal gradients approach the linear steady-state condition. This phenomenon is well presented by Fig. 5.22, in which both concrete's shear and normal stresses within the outer-corner element approach to zero at conclusion of the test.

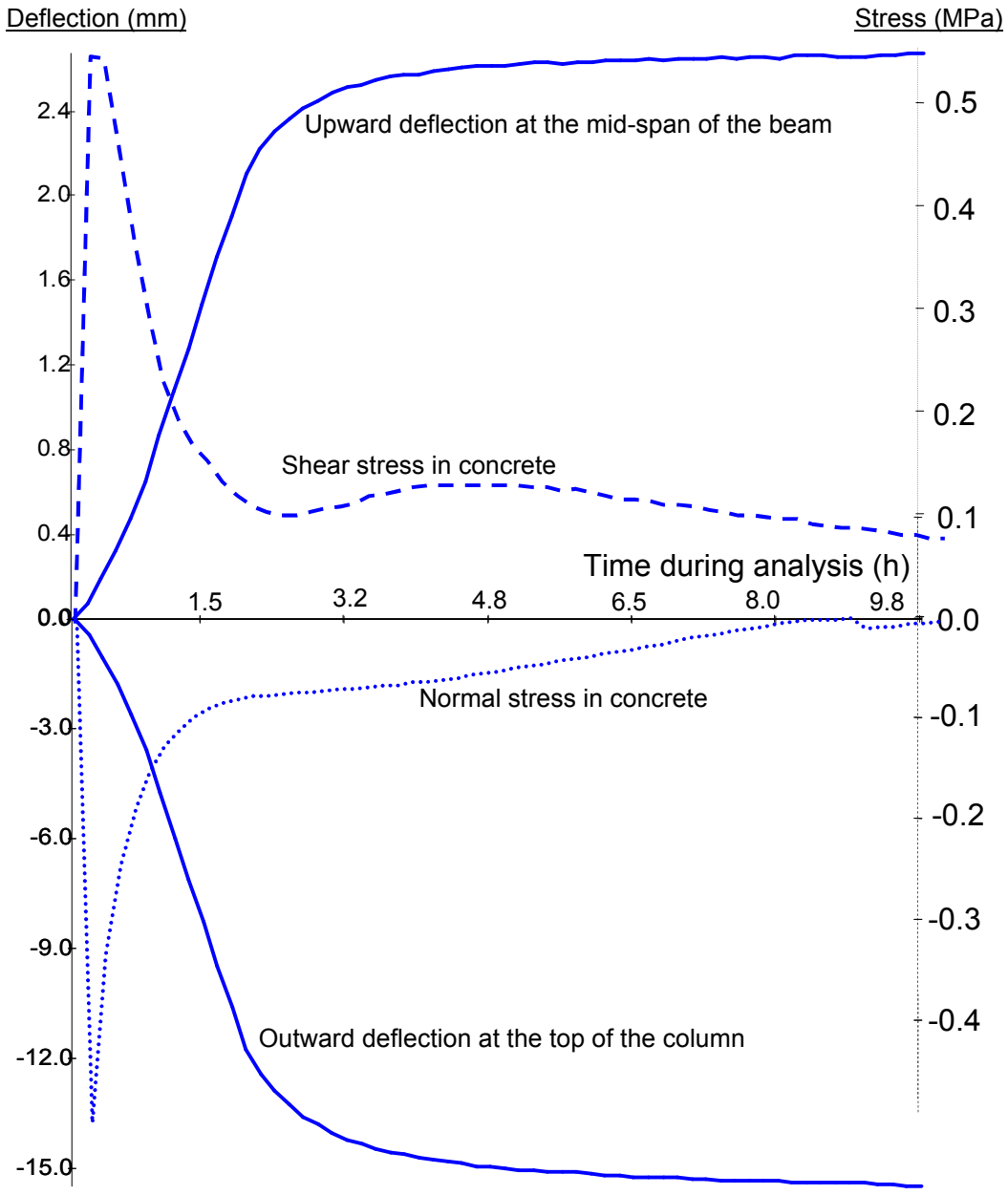


Figure 5.22 Deflections and stresses in Type I test

2. Type II test:

In this test, all three levels of temperature loading are applied on the model. As shown in Fig. 5.23, the restraint forces induced in the model present a similar style of overall trend for all types of load. A higher temperature gradient seems to have a faster increase in the initial stages, while the peak value appears to be lower. Also, the time when peak value of restraint forces occurs are not corresponding to the ones of thermal gradients. In this aspect, a higher temperature load appears to have a further gap between these two peak times. After peaking, the forces then have a large drop. It is likely due to the sudden severe cracking that develops during these times, which renders the structure less stiff. Actually, a lower temperature load 'postpones' the occurring and progression of severe cracking. Consequently, the restraint force can develop a longer time, during which the internal forces can be redistributed better. After dropping, the forces increase marginally (due to increase of temperature loading) until the temperature gradients within structure achieve the highest state. Thereafter, the forces remain essentially constant over the remainder of the test. For example, in the case with load type B2T, a severe cracking progression is observed at the location about 500 mm from the base of the column, as shown in Fig. 5.24. The concrete stress at this location goes up to the tensile strength of the concrete.

In addition to crack conditions, it is found that the restraint force induced is also sensitive to yield conditions within the structure. Within the test of the thermal load Type I on the model, a yield stress in reinforcement is observed during the drop period, at the location around the cracking shown in Fig. 5.25. The yielding of reinforcement can significantly reduce the structure's stiffness, and in turn reduce the restraint force in the statically indeterminate structure.

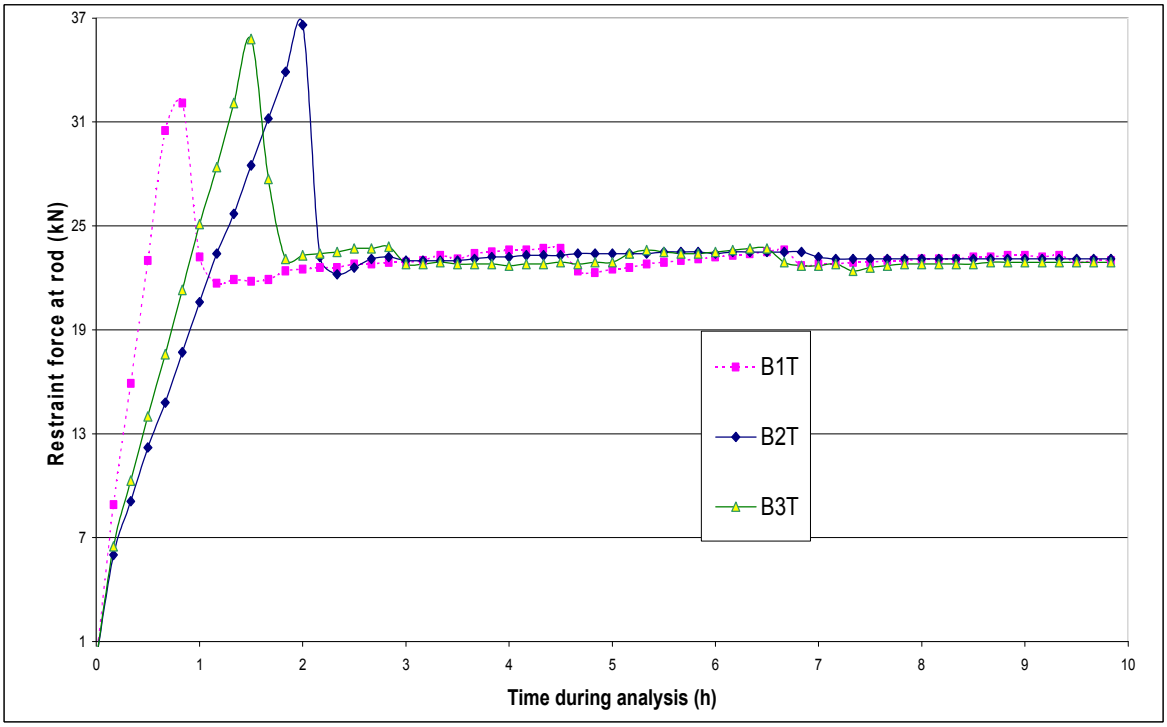


Figure 5.23 Restraint force induced at engaged tie-rods in Type II test

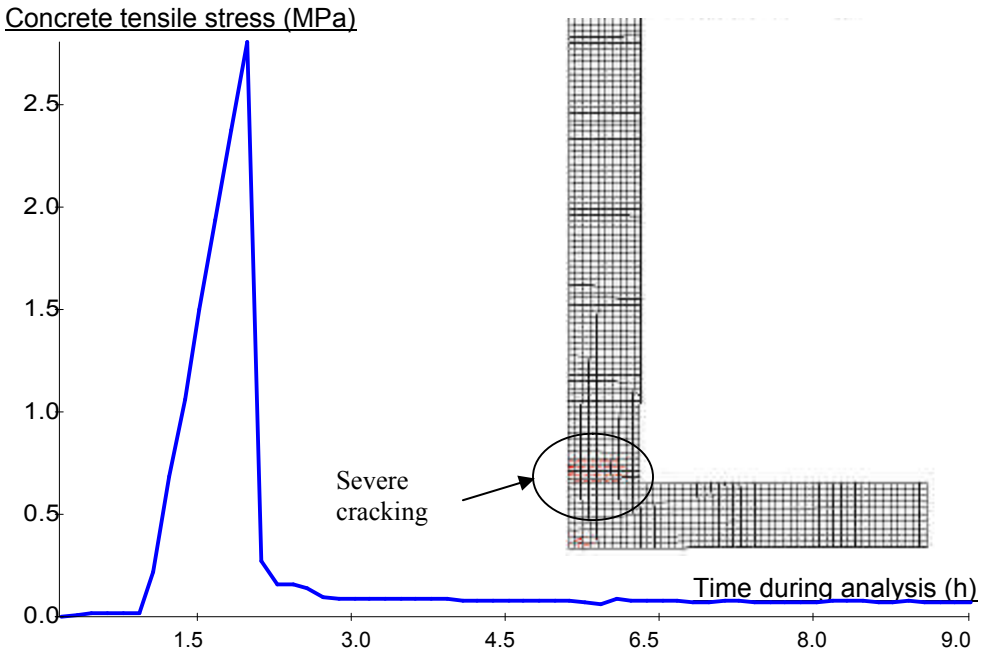


Figure 5.24 Concrete stresses and crack pattern in Type II test

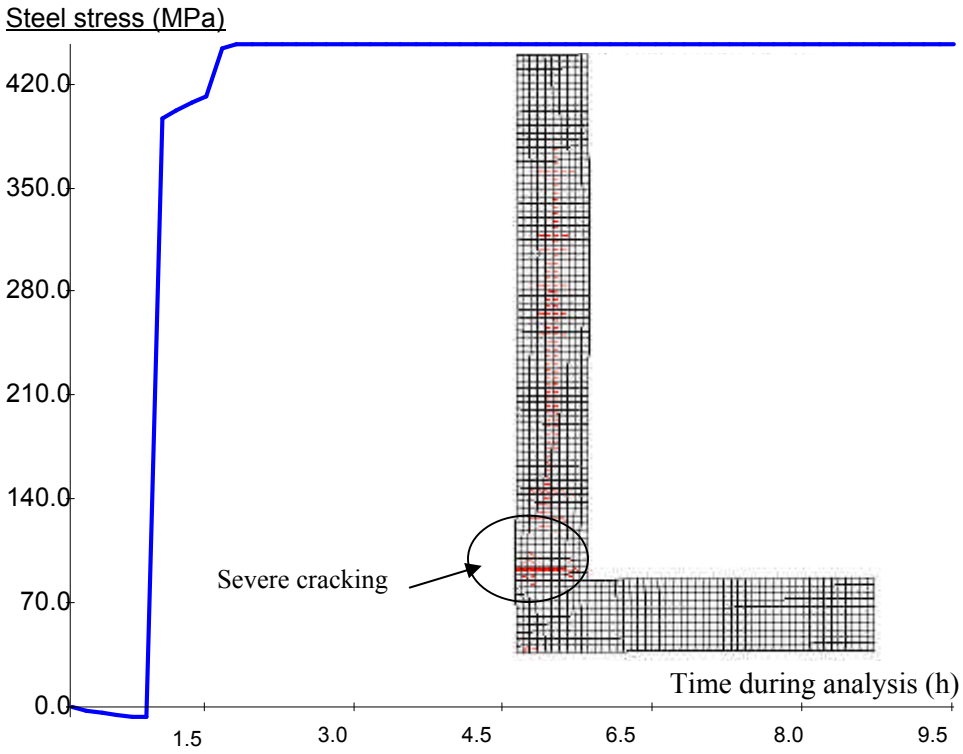


Figure 5.25 Yield stress in reinforcement during Type II test

3. Type III test:

In Type III testing, in addition to the temperature loading, the lateral load applied is monotonically increased until the ultimate capacity of the model is exceeded.

During the test of temperature load Type III on the model, as shown in Fig. 5.26 (type "3R+3T") the load-deformation curve of the structure is linear until the bottom reinforcement in the beam yields at about 1.0 hours. Thereafter, response is essential plastic with a limited increase in the load capacity, mainly due to strain hardening. A response of the model without the thermal load is also plotted in Fig. 5.26, denoted by type "3R+0T". From the comparison, the presence of a thermal load in the case of test model does not appear to reduce the ultimate capacity of the structure very much. However, if the

thermal load is sufficiently high it can crack the structure when the mechanical load is still at a low level. In some cases, the presence of the thermal load may yield the structure much earlier than the case without thermal loads. In addition, a highly nonlinear thermal gradient will induce primary stresses within the structure and reduce the stiffness and strength of the structure to a significant degree.

Noticing that the structure is statically determinate and no force redistribution will then be present, extended tests on restrained model in Type II test are also performed, with and without the thermal load Type III. The load-deformation curves obtained are also plotted in Fig. 5.26, denoted by types "2R+0T" and "2R+3T"

Through the comparisons between types "3R+3T" and "2R+3T", it is observed that the introduced restraint highly increases the structure's ultimate capacity. From the comparisons between the types "2R+1T" and "2R+0T", it is again not found that the presence of the thermal load (even in a high level as B1T) produces a significant influence on the response of the statically indeterminate structure. However, the presence of thermal loads should be always of attention since the cracking and yield of the structure are likely sensitive to the amount of the thermal loads applied.

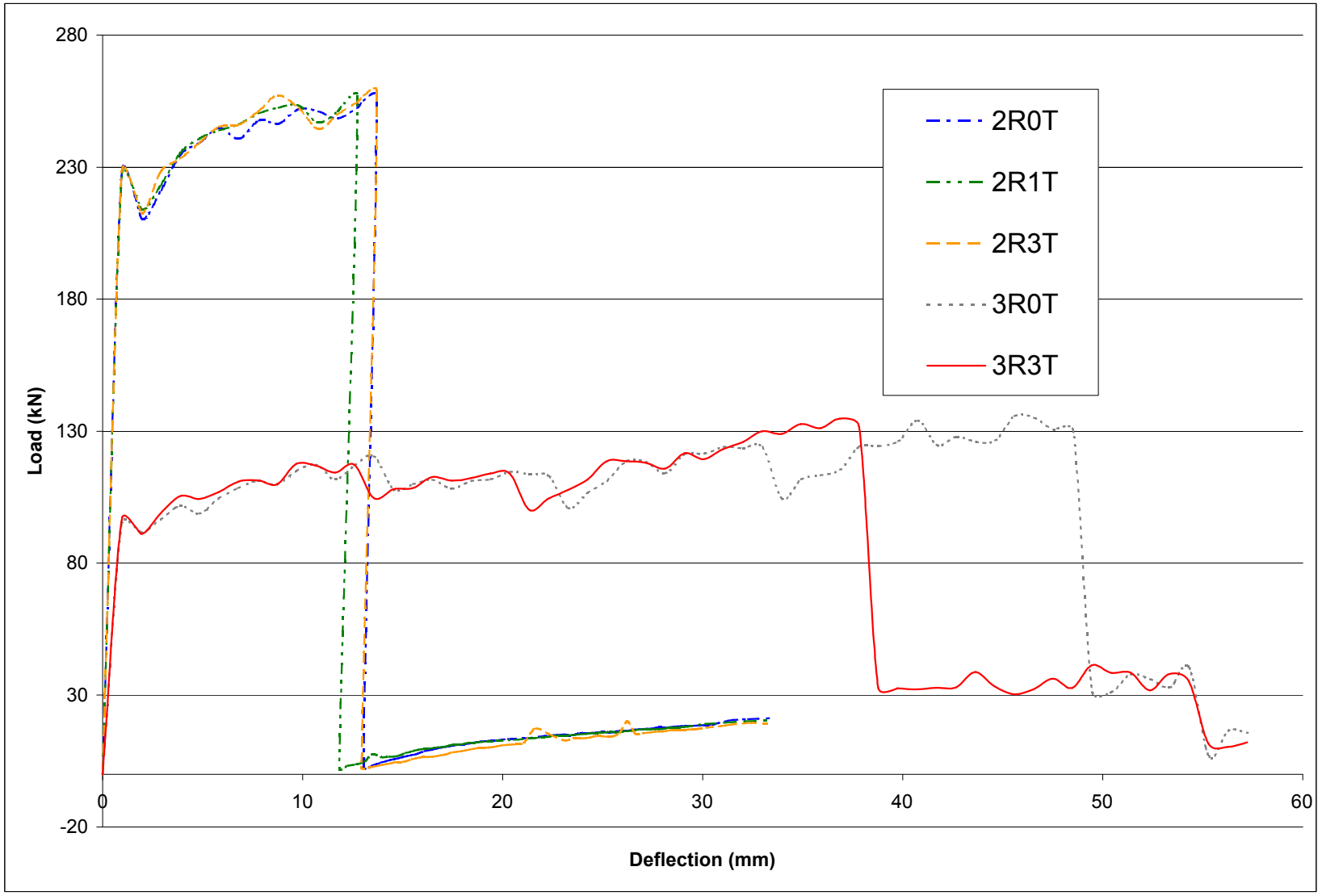


Figure 5.26 Load-deformation curves obtained in Type III test

Chapter 6 Conclusions and Further Research Prospects

6.1 Conclusions

In this work, a 2D transient nonlinear thermal analysis capability is implemented into program VecTor2. Models are created to account for the time-varying thermal loads and the temperature-dependence of the various concrete and steel mechanical and thermal properties that influence structural and thermal analyses. In addition, formulations for considering heat-of-hydration problems and algorithms for calculating the closed-form element stiffness for a quadrilateral element with a fully-populated material stiffness are developed and realized.

With the results obtained from numerical corroboration testing, the developed computational scheme and the implemented code are found to be accurate, stable and reliable. However, considerable improvements are required before they equal the prominence of the static and dynamic analyses within the VecTor suite of program development and in the advanced concrete structural analysis.

6.2 Suggestions

While a wide variety of research is possible in future work, some aspects closely related to this research are suggested below for further investigation and development.

- In the FE analysis:

For practical engineering problems, the subject of error estimation for numerical solutions and the development of adaptive refinement procedure are central. As a nonlinear finite element analysis facility, VecTor suite of programs have a common but challenging step to move forward in this aspect. For all incorporated finite element calculations, including the heat flow analysis, an error estimation process is required to assess the performance of the current discretization and to provide quantitative description on the accuracy of the present solution. Such a description is essential for the refinement process in which a new finite element mesh will be generated to reduce the discretization error by increasing the number of degrees of freedom where the previous analysis is not adequate. In order to do that, a well designed refinement strategy is necessary to define that new spatial and/or temporal discretization in a most economic manner.

- In the heat modeling:

The computational methods have become extremely powerful and sophisticated with great accuracy during the past few decades. However, from the point of view of overall structural reliability, the results of sophisticated computations of structural response are only, in the end, as good as the assumptions for the simulation or simplification of applied loads, which includes fire. Specifically in heat modeling for concrete structures, heat-of-hydration, thermal creep and spalling effect, and multi-phase change remain intensive research topics though much effort was devoted in previous decades. Also, precise knowledge of the mechanism and kinetics of the microstructure for material properties' temperature dependency require further investigation.

- In the VecTor development:

As direct future work, heat flow analysis is required to be incorporated into three-dimensional problems. It is imperative to implement the nonlinear transient FE thermal analysis capacity into the entire VecTor suite of problems, such as VecTor3 for 3D solid and other kinds of structure types. In this extension, attention must be paid to the efficiency of the developed computationally scheme since the highly nonlinear transient analysis is computational intensive. In addition, the window-based input and graphic-based output facilities are also needed to facilitate user interaction and result presentation.

- In experimental work:

Experimental tests are essential for the parameter collecting, formulation corroboration, and theory validation. The setup of various heat models would require a large database to be established. Also, results obtained from laboratory tests could uniquely verify the developed numerical scheme. As an urgent need from the current research, hydration and thermal creep testing as well as high-temperature material property evaluation are exceedingly important.

References

1. Kodur V. (2003), "World Trade Centre Disaster – Building Performance Investigation", CSCE Workshop at Toronto.

2. Vecchio F.J.; Agostino N.; and Angelakos B. (1993), "Reinforced Concrete Slabs Subject to Thermal Loads", Canadian Journal of Civil Engineering, Vol. 20, No. 5, pp. 741-753.

3. Vecchio F.J. and Collins M.P. (1986), "The Modified Compression-Field Theory for Reinforced Concrete Elements subjected to Shear", ACI Journal, Vol. 83, No. 2, pp219-231.

4. Vecchio, F.J. (2000) "Disturbed Stress Field Model for Reinforced Concrete: Formulation", ASCE Journal of Structural Engineering, Vol. 126, No. 8, pp 1070-1077.

5. Chung T.J. (2002), Computational Fluid Dynamics, Cambridge Press, 1012 pp.

6. Sneddon I.N. (1957), Elements of Partial Differential Equations, McGraw-Hill, 327 pp.

7. Blazek J. (2001), Computational Fluid Dynamics, Elsevier, 440 pp.

8. Zienkiewicz O.C. and Taylor R.L. (2000), The Finite Element Method, 5th edition, John Wiley & Sons, 689 pp.

9. Vecchio, F.J. (1990), "Reinforced Concrete Membrane Element Formulation", ASCE Journal of Structural Engineering, Vol. 116, No. 3, pp 730-750.

10. Heinrich J.C. and Pepper D.W. (1999), Intermediate Finite Element Method: Fluid Flow and Heat Transfer Applications, Taylor & Francis, 596 pp.

11. Lewis R.W.; Morgan K.; Thomas H.R.; and Seetharamu K.N. (1996), The Finite Element Method in Heat Transfer Analysis, Wiley, 279 pp.

12. Farassat F. and Myers M.K. (1987), Extension of Kirchhoff's Formula to Radiation from Moving Surfaces, Langley Research Center.

13. Ozisik M.N. (1994), Finite Difference Methods in Heat Transfer, CRC Press, 412 pp.

14. Shin K.Y. et al. (2002), "Thermo-physical Properties and Transient Heat Transfer of Concrete at Elevated Temperatures", Nuclear Engineering and Design, Vol. 212, No. 2, pp233-241.

15. Zhu X.K. and Chao Y.J. (2002), "Effects of Temperature-Dependent Material Properties on Welding Simulation", Computers and Structures, Vol. 80, No. 11, pp967-976.

16. Vecchio F.J. (1987), "Nonlinear Analysis of Reinforced Concrete Frames Subjected to Thermal and Mechanical Loads", ACI Structural Journal, Vol. 84, No. 6, pp492-501.

17. Castillo C. and Durrani A. J. (1990), "Effect of Transient High Temperature on High-Strength Concrete", ACI Material Journal, Vol. 87, No. 1, pp47-53.

18. Lie T.T. and Kodur V.K.R. (1996), "Thermal and Mechanical Properties of Steel-fibre-reinforced Concrete at Elevated Temperatures", Canadian Journal of Civil Engineering, Vol. 23, No. 2, pp511-517.

19. Bentz D.P. (1997), "Three-Dimensional Computer Simulation of Portland Cement Hydration and Microstructure Development", NISTIR 5756, U.S. Department of Commerce.

20. Khan A.A.; Cook W.D.; and Mitchell D. (1998), "Thermal Properties and Transient Thermal Analysis of Structural Members during Hydration", ACI Material Journal, Vol. 95, No. 3, pp293-303.

21. Pane I. and Hansen W. (2002), "Concrete Hydration and Mechanical Properties under Nonisothermal Conditions", ACI Material Journal, Vol. 99, No. 6, pp534-542.

22. Naik T.R. (1992), "Maturity of Concrete: Its Application and Limitations", Advances in Concrete Technology (edited by Malhotra V.M.), pp329-349.

23. Pinto R.C.A.; Hobbs S.V.; and Hover K.C. (2001), "The Maturity Approach in Concrete Technology-Going Beyond Compressive Strength", Recent Advances in Concrete Technology (edited by Malhotra V.M.), pp749-769.

24. Freiesleben-Hanson P. and Pedersen E.J. (1977), "Maturity Computer for Controlled Curing and Hardening of Concrete", *Nordisk Betong*, Vol. 1, pp21-25.

25. Saul A.G.A (1951), "Principles Underlying of the Steam Curing of Concrete at Atmospheric Pressure", Magazine of Concrete Research, Vol. 2, No. 6, pp127-140.

26. Rastrup E. (1954), "Heat of Hydration in Concrete", Magazine of Concrete Research, Vol. 6, No. 17, pp79-92.

27. Knudsen T. (1984), "The Dispersion Model for Hydration of Portland Cement I. General Concepts", Cement and Concrete Research, Vol. 14, pp622-630.

28. Springenschmid R. (1995), "Thermal Cracking in Concrete at Early Ages", Proceedings of the International Symposium held by RILEM, E & FN SPON, 470 pp.

29. Springenschmid R. (1998), "Thermal Cracking in Concrete at Early Ages", RILEM Report 15, E & FN SPON, 348 pp.

30. Lee C.K. and Zhou C.E. (2003), "On Error Estimation and Automatic Adaptive Refinement for Element-Free Galerkin Method, Part 2: Adaptive Refinement", Computers and Structures, In Press.

31. Belytschko T. and Hughes T.J.R. (1983) Computational Methods for Transient Analysis, North-Holland Amsterdam, 523 pp.

32. Vecchio, F.J. and Sato J.A. (1990), "Thermal Gradient Effects in Reinforced Concrete Frame Structures", ACI Structural Journal, Vol. 87, No. 3, pp 262-275.

33. Lohner R. (2001) Applied CFD Techniques, Wiley, 366 pp.

34. Griffiths D.V. (1994) "Stiffness Matrix of the Four-Node Quadrilateral Element in Closed Form", International Journal for Numerical Methods in Engineering, Vol. 37, pp 1027-1038.

35. ANSYS Thermal Analysis Guide. 00858. 2nd Edition, SAS IP Inc.

Appendix A VecTor2 Input Files for Problem 3

A1.JOB

```
                              * * * * * * * * * * * *
                              *     V e c T o r     *
                              *   J O B   D A T A   *
                              * * * * * * * * * * * *

Job Title       (30 char. max.)        : A1
Job File Name   ( 8 char. max.)        : A1
Date            (30 char. max.)        : Nov 08, 2003

STRUCTURE DATA
--------------
Structure Type                         : 2
File Name       ( 8 char. max.)        : A1R

LOADING DATA

------------
No. of Load Stages                     : 54
Starting Load Stage No.                : 1
Load Series ID  ( 5 char. max.)        : A1

Load   File Name         |                    Factors                    |
Case   (8 char max)          Initial    Final    LS-Inc  Type Reps    C-Inc
  1    A1T                   1.000     1.000   600.000    1    2     1.000
  2    A1G                   1.000     1.000     0.000    2    1     0.000
  3    NULL                  0.000    40.000     0.500    1    1     0.000
  4    NULL                  0.000     0.000     0.000    1    1     0.000
  5    NULL                  0.000     0.000     0.000    1    1     0.000

ANALYSIS PARAMETERS
-------------------
Analysis Mode                        (1-2) : 1
Seed File Name                (8 char max) : NULL
Convergence Limit                   (>1.0) : 1.00001
Averaging Factor                    (<1.0) : 0.25
Maximum Iterations                         : 100
Convergence Criteria                 (1-5) : 2
Results Files                        (1-4) : 2
Output Format                        (1-3) : 1

MATERIAL BEHAVIOUR MODELS
-------------------------
Concrete Compression Base Curve      (0-3) : 1
Concrete Compression Post-Peak       (0-3) : 1
Concrete Compression Softening       (0-8) : 1
Concrete Tension Stiffening          (0-5) : 1
Concrete Tension Softening           (0-3) : 1
Concrete Tension Splitting           (1-2) : 1
Concrete Confined Strength           (0-2) : 1
Concrete Dilatation                  (0-1) : 1
Concrete Cracking Criterion          (0-4) : 1
Concrete Crack Slip Check            (0-2) : 1
Concrete Crack Width Check           (0-2) : 1
Concrete Bond or Adhesion            (0-4) : 1
Concrete Creep and Relaxation        (0-1) : 1
Concrete Hysteresis                  (0-3) : 2
Reinforcement Hysteresis             (0-3) : 1
Reinforcement Dowel Action           (0-1) : 1
Reinforcement Buckling               (0-1) : 1
Element Strain Histories             (0-1) : 1
Element Slip Distortions             (0-4) : 1
Strain Rate Effects                  (0-1) : 1
Structural Damping                   (0-1) : 1
Geometric Nonlinearity               (0-1) : 1
Crack Allocation Process             (0-1) : 1
```

A1R.S2R

```
                  * * * * * * * * * * * * * * * * * * *
                  *          V e c T o r 2            *
                  *   S T R U C T U R E    D A T A    *
                  * * * * * * * * * * * * * * * * * * *

                         STRUCTURAL PARAMETERS
                         *********************

Structure Title        (30 char. max.)     : BRESLER SCORDELIS A1
Structure File Name    ( 8 char. max.)     : A1
No. of R.C. Material Types                 : 2
No. of Steel Material Types                : 2
No. of Bond Material Types                 : 0
No. of Rectangular Elements                : 370
No. of Quadrilateral Elements              : 0
No. of Triangular Elements                 : 0
No. of Truss Elements                      : 111
No. of Linkage Elements                    : 0
No. of Contact Elements                    : 0
No. of Joints                              : 418
No. of Restraints                          : 33

                       MATERIAL SPECIFICATIONS
                       ***********************

                       (A) REINFORCED CONCRETE
                       -----------------------
<NOTE:> FOR RECTANGULAR, QUADRILATERAL AND TRIANGULAR ELEMENTS ONLY

CONCRETE
--------
MAT  Ns    T    f'c  [ f't    Ec     e0     Mu   Cc   Agg   Dens   Kc ] [Sx  Sy]
TYP  #    mm    MPa    MPa    MPa    me          /C    mm  kg/m3  mm2/s  mm  mm
  1   0   305  24.1   1.88  24100  2.00   0.15   0    15     0     0    0    0
  2   1   305  24.1   1.88  24100  2.00   0.15   0    15     0     0    0    0
/
REINFORCEMENT COMPONENTS
-----------------------
MAT SRF  DIR  As  Db  Fy   Fu    Es    Esh    esh   Cs   Dep
TYP TYP  deg  %   mm  MPa  MPa   MPa   MPa    me    /C   me
  2   1  90. 0.099  7.5  325  600  200000  2000  5  0  0
/
                              (B) STEEL
                              ---------
<NOTE:> TO BE USED FOR TRUSS ELEMENTS ONLY
MAT  REF    AREA   Db    Fy    Fu    Es    Esh   esh   Cs   Dep
TYP  TYP     mm2   mm    MPa   MPa   MPa   MPa    me   /C    me
1   1    1282    29     555   900  200000  2000  5   11.5E-6  0
2   1     253    13     345   700  200000  2000  5   11.5E-6  0
/
                              (C) BOND
                              --------
<NOTE:> TO BE USED FOR EXTERIOR/INTERIOR BONDED ELEMENTS
MAT  REF  { Ao   U1    U2    U3    S1    S2    S3 }/{ CPF   Cmin   No.  HOOK }
TYP  TYP   mm^2  MPa   MPa   MPa   mm    mm    mm     0-1    mm    LYR   0/1
/
                         ELEMENT INCIDENCES
                         ******************

                       (A) RECTANGULAR ELEMENTS
                       ------------------------
<<<<< FORMAT >>>>>
ELMT INC1 INC2 INC3 INC4  [ #ELMT d(ELMT) d(INC) ] [ #ELMT d(ELMT) d(INC) ] /
  1   1   2  40  39   37  1  1  10  37  38 /
/
                      (B) QUADRILATERAL ELEMENTS
                      --------------------------
<<<<< FORMAT >>>>>
ELMT INC1 INC2 INC3 INC4  [ #ELMT d(ELMT) d(INC) ] [ #ELMT d(ELMT) d(INC) ] /
/
```

```
                        (C) TRIANGULAR ELEMENTS
                        -----------------------
<<<<< FORMAT >>>>>
ELMT INC1 INC2 INC3  [ #ELMT d(ELMT) d(INC)] [ #ELMT d(ELMT) d(INC) ] /
/
                          (D) TRUSS ELEMENTS
                          ------------------
<<<<< FORMAT >>>>>
ELMT INC1 INC2 [ #ELMT d(ELMT) d(INC)] [ #ELMT d(ELMT) d(INC) ]  /
371  39  40  37  1  1 /
408  77  78  37  1  1 /
445 343 344  37  1  1 /
/
                          (E) LINKAGE  ELEMENTS
                          ---------------------
<<<<< FORMAT >>>>
ELMT INC1 INC2 [ #ELMT d(ELMT) d(INC) ] [ #ELMT d(ELMT) d(INC) ]
 /
                          (F) CONTACT  ELEMENTS
                          ---------------------
<<<<< FORMAT >>>>
ELMT INC1 INC2 INC3 INC4 [ #ELMT d(ELMT) d(INC) ] [ #ELMT d(ELMT) d(INC) ]
/
                        MATERIAL TYPE ASSIGNMENT
                        ************************
<<<<< FORMAT >>>>>
ELMT  MAT  ACT  [ #ELMT d(ELMT)] [ #ELMT d(ELMT) ] /
  1  1  1   37 1 /
 38  2  1  296 1 /
334  1  1   37 1 /
371  1  1   74 1 /
445  2  1   37 1 /
/
                           COORDINATES
                           ***********
<NOTE:> UNITS:  mm
<<<<< FORMAT >>>>>
NODE X Y  [ #NODES d(NODES)  d(X) d(Y) ] [ #NODES d(NODES)  d(X) d(Y) ] /
  1    0.    0.   38  1  55.5  0.  2  38  0.  65. /
 77    0.  125.   38  1  55.5  0.  7  38  0.  55. /
343    0.  510.   38  1  55.5  0.  2  38  0.  50. /
/
                        SUPPORT RESTRAINTS
                        ******************
<NOTE:> CODE:  '0' FOR NO RESTRAINT; '1' FOR RESTRAINT
<<<<< FORMAT >>>>>
NODE  X-RST  Y-RST [ #NODE d(NODE) ]  /
  1 1 1 11 38/
 38 1 0 11 38/
/


<NOTES:>
```

A1G.L2R

```
                        * * * * * * * * * * * * *
                        *     V e c T o r 2     *
                        *   L O A D   D A T A   *
                        * * * * * * * * * * * * *

                          LOAD CASE PARAMETERS
                          ********************

Structure Title      (30 char. max.)     : B/S Beams
Load Case Title      (30 char. max.)     : 100 kN
Load Case File Name   (8 char. max.)     : A1
No. of Loaded Joints                     : 0
No. of Prescribed Support Displacements  : 0
No. of Elements with Gravity Forces      : 370
No. of Elements with Temperature Change  : 0
No. of Elements with Concrete Prestrain  : 0
No. of Elements with Ingress Pressure    : 0
No. of Nodes with Thermal Load           : 0
No. of Nodes with Lumped Masses          : 0
No. of Nodes with Impulse Forces         : 0
Ground Acceleration Record  (0-1)        : 0

                             JOINT LOADS
                             ***********
<NOTE:>  UNITS:  KN
<<<<< FORMAT >>>>>
NODE    Fx    Fy    [ #NODE d(NODE) d(Fx) d(Fy) ] /
/
                         SUPPORT DISPLACEMENTS
                         *********************
<NOTE:> UNITS: MM
<<<<< FORMAT >>>>>
JNT  DOF  DISPL   [ #JNT d(JNT) ] /
/
                            GRAVITY LOADS
                            *************
<NOTE:>  UNITS:  KG/M3
<<<<< FORMAT >>>>>
ELMT  DENS  GX  GY  [#ELMT d(ELMT)] [ #ELMT d(ELMT)] /
  1   2400   0   1   370  1 /
/
                           TEMPERATURE LOADS
                           *****************
<NOTE:>  UNITS:  C
<<<<< FORMAT >>>>>
ELMT   TEMP   [ #ELMT d(ELMT) d(TEMP) ] [ #ELMT d(ELMT) d(TEMP) ] /
/
                          CONCRETE PRESTRAINS
                          *******************
<NOTE:>  UNITS:  me
<<<<< FORMAT >>>>>
ELMT  STRAIN   [ #ELMT d(ELMT) d(STRAIN) ] [ #ELMT d(ELMT) d(STRAIN) ]  /
/
                           INGRESS PRESSURES
                           *****************
<NOTE:>  UNITS:  MPa
<<<<< FORMAT >>>>>
ELMT PRESSURE  [ #ELMT d(ELMT) d(PRS) ] [ #ELMT d(ELMT) d(PRS) ]  /
/
                          NODAL THERMAL LOADS
                          *******************
<NOTE:>  UNITS:  Sec, Degrees C
<<<<< FORMAT >>>>>
NODE  TYPE  Tm1 Tp1  Tm2 Tp2  Tm3 Tp3  [#NODE d(NODE)] [#NODE d(NODE)]  /
/
                            LUMPED MASSES
                            *************
<NOTE:>  UNITS:  kg, m/s
<<<<< FORMAT >>>>>
NODE  DOF-X  DOF-Y  MASS  GF-X  GF-Y  Vo-X  Vo-Y  [ #NODE d(NODE) ] /
```

```
/
                            IMPULSE FORCES
                            **************
<NOTE:>  UNITS:  Sec, kN
<<<<< FORMAT >>>>>
NODE  DOF  T1   F1   T2   F2   T3   F3   T4   F4  [ #NODE d(NODE) ] /
/
                           GROUND ACCELERATION
                           *******************
<NOTE:>  UNITS:  Sec, m/s2
<<<<< FORMAT >>>>>
TIME   ACC-X   ACC-Y
/


<NOTES:>
```

A1T.L2R

```
                        * * * * * * * * * * * * *
                        *     V e c T o r 2     *
                        *   L O A D   D A T A   *
                        * * * * * * * * * * * * *

                          LOAD CASE PARAMETERS
                          ********************

Structure Title      (30 char. max.)     : B/S Beams
Load Case Title      (30 char. max.)     : 100 kN
Load Case File Name   (8 char. max.)     : A1
No. of Loaded Joints                     : 0
No. of Prescribed Support Displacements  : 0
No. of Elements with Gravity Forces      : 0
No. of Elements with Temperature Change  : 0
No. of Elements with Concrete Prestrain  : 0
No. of Elements with Ingress Pressure    : 0
No. of Nodes with Thermal Load           : 76
No. of Nodes with Lumped Masses          : 0
No. of Nodes with Impulse Forces         : 0
Ground Acceleration Record  (0-1)        : 0

                             JOINT LOADS
                             ***********
<NOTE:>  UNITS:  KN
<<<<< FORMAT >>>>>
NODE    Fx    Fy    [ #NODE d(NODE) d(Fx) d(Fy) ] /
/
                         SUPPORT DISPLACEMENTS
                         *********************
<NOTE:> UNITS: MM
<<<<< FORMAT >>>>>
JNT  DOF  DISPL   [ #JNT d(JNT) ] /
/
                             GRAVITY LOADS
                             *************
<NOTE:>  UNITS:  KG/M3
<<<<< FORMAT >>>>>
ELMT  DENS  GX  GY  [#ELMT d(ELMT)] [ #ELMT d(ELMT)] /
/
                            TEMPERATURE LOADS
                            *****************
<NOTE:>  UNITS:  C
<<<<< FORMAT >>>>>
ELMT   TEMP   [ #ELMT d(ELMT) d(TEMP) ] [ #ELMT d(ELMT) d(TEMP) ] /
/
                           CONCRETE PRESTRAINS
                           *******************
<NOTE:>  UNITS:  me
<<<<< FORMAT >>>>>
ELMT  STRAIN   [ #ELMT d(ELMT) d(STRAIN) ] [ #ELMT d(ELMT) d(STRAIN) ]  /
/
                            INGRESS PRESSURES
                            *****************
<NOTE:>  UNITS:  MPa
<<<<< FORMAT >>>>>
ELMT PRESSURE  [ #ELMT d(ELMT) d(PRS) ] [ #ELMT d(ELMT) d(PRS) ]  /
/
```

Type 1: Steady-State Model

```
                           NODAL THERMAL LOADS
                           *******************
<NOTE:>  UNITS:  Sec, Degrees C
<<<<< FORMAT >>>>>
NODE  TYPE  Tm1 Tp1  Tm2 Tp2  Tm3 Tp3  [#NODE d(NODE)] [#NODE d(NODE)]  /
  1  1    0 1000  43200  1000  45000 10    38 1 /
381  1    100   0   43200    0   45000  0    38 1 /
/
```

Type 2: Three-Key-Node Linear Model

```
                        NODAL THERMAL LOADS
                        ******************
<NOTE:>  UNITS:  Sec, Degrees C
<<<<< FORMAT >>>>>
NODE  TYPE  Tm1 Tp1  Tm2 Tp2  Tm3 Tp3  [#NODE d(NODE)] [#NODE d(NODE)]  /
  1  2    2400 900  14400  1000  28800 100    38 1 /
381  2    2400   0   14400    0   28800  0    38 1 /
/
```

Type 3: Compartment Simplified Fire Model

```
                        NODAL THERMAL LOADS
                        ******************
<NOTE:>  UNITS:  Sec, Degrees C
<<<<< FORMAT >>>>>
NODE  TYPE  Tm1 Tp1  Tm2 Tp2  Tm3 Tp3  [#NODE d(NODE)] [#NODE d(NODE)]  /
  1  3    30 255  10800  1064  32400 1    38 1 /
381  2    100   0   10800    0   32400  0    38 1 /
/
```

Type 4: ASTM-E119 Fire Model

```
                        NODAL THERMAL LOADS
                        ******************
<NOTE:>  UNITS:  Sec, Degrees C
<<<<< FORMAT >>>>>
NODE  TYPE  Tm1 Tp1  Tm2 Tp2  Tm3 Tp3  [#NODE d(NODE)] [#NODE d(NODE)]  /
  1  4    100 255  10800  1064  4900 10    38 1 /
381  2    100   0   10800    0   32400  0    38 1 /
/
```

Type 5: ISO-834 Fire Model

```
                        NODAL THERMAL LOADS
                        ******************
<NOTE:>  UNITS:  Sec, Degrees C
<<<<< FORMAT >>>>>
NODE  TYPE  Tm1 Tp1  Tm2 Tp2  Tm3 Tp3  [#NODE d(NODE)] [#NODE d(NODE)]  /
  1  5    100 255  10800  1064  32400 10    38 1 /
381  2    100   0   10800    0   32400  0    38 1 /
/

                          LUMPED MASSES
                          *************
<NOTE:>  UNITS:  kg, m/s
<<<<< FORMAT >>>>>
NODE  DOF-X  DOF-Y  MASS  GF-X  GF-Y  Vo-X  Vo-Y  [ #NODE d(NODE) ] /
/
                         IMPULSE FORCES
                         **************
<NOTE:>  UNITS:  Sec, kN
<<<<< FORMAT >>>>>
NODE  DOF  T1   F1   T2   F2   T3   F3   T4   F4  [ #NODE d(NODE) ] /
/
                        GROUND ACCELERATION
                        *******************
<NOTE:>  UNITS:  Sec, m/s2
<<<<< FORMAT >>>>>
TIME   ACC-X   ACC-Y
/


<NOTES:>
```

Appendix B VecTor2 Input Files for Problem 4

VECTOR-I.JOB

```
                          * * * * * * * * * * * *
                          *     V e c T o r     *
                          *   J O B   D A T A   *
                          * * * * * * * * * * * *

Job Title       (30 char. max.)        : Frame
Job File Name   ( 8 char. max.)        : FI
Date            (30 char. max.)        : Nov 13, 2003

STRUCTURE DATA
--------------
Structure Type                         : 2
File Name       ( 8 char. max.)        : F1R

LOADING DATA
------------
No. of Load Stages                     : 60
Starting Load Stage No.                : 1
Load Series ID  ( 5 char. max.)        : FI

Load   File Name         |                    Factors                    |
Case   (8 char max)          Initial    Final    LS-Inc  Type Reps    C-Inc
  1    F1T                   1.000     1.000     600.0    1    2     1.000
  2    FG                    1.000     1.000     0.000    2    1     0.000
  3    NULL                  0.000    40.000     0.500    1    1     0.000
  4    NULL                  0.000     0.000     0.000    1    1     0.000
  5    NULL                  0.000     0.000     0.000    1    1     0.000

ANALYSIS PARAMETERS
-------------------
Analysis Mode                      (1-2) : 1
Seed File Name                (8 char max) : NULL
Convergence Limit                 (>1.0) : 1.00001
Averaging Factor                  (<1.0) : 0.5
Maximum Iterations                       : 50
Convergence Criteria               (1-5) : 2
Results Files                      (1-4) : 2
Output Format                      (1-3) : 1

MATERIAL BEHAVIOUR MODELS
-------------------------
Concrete Compression Base Curve    (0-3) : 1
Concrete Compression Post-Peak     (0-3) : 1
Concrete Compression Softening     (0-8) : 1
Concrete Tension Stiffening        (0-5) : 1
Concrete Tension Softening         (0-3) : 1
Concrete Tension Splitting         (1-2) : 1
Concrete Confined Strength         (0-2) : 1
Concrete Dilatation                (0-1) : 1
Concrete Cracking Criterion        (0-4) : 1
Concrete Crack Slip Check          (0-2) : 1
Concrete Crack Width Check         (0-2) : 1
Concrete Bond or Adhesion          (0-4) : 1
Concrete Creep and Relaxation      (0-1) : 1
Concrete Hysteresis                (0-3) : 2
Reinforcement Hysteresis           (0-3) : 1
Reinforcement Dowel Action         (0-1) : 1
Reinforcement Buckling             (0-1) : 1
Element Strain Histories           (0-1) : 1
Element Slip Distortions           (0-4) : 1
Strain Rate Effects                (0-1) : 1
Structural Damping                 (0-1) : 1
Geometric Nonlinearity             (0-1) : 1
Crack Allocation Process           (0-1) : 1

<NOTES:>
```

VECTOR-II.JOB

```
                              * * * * * * * * * * * *
                              *     V e c T o r     *
                              *   J O B   D A T A   *
                              * * * * * * * * * * * *

Job Title       (30 char. max.)        : Frame
Job File Name   ( 8 char. max.)        : FII
Date            (30 char. max.)        : Nov 13, 2003

STRUCTURE DATA
--------------
Structure Type                         : 2
File Name       ( 8 char. max.)        : F2R

LOADING DATA
------------
No. of Load Stages                     : 60
Starting Load Stage No.                : 1
Load Series ID  ( 5 char. max.)        : FII

Load   File Name         |                    Factors                    |
Case   (8 char max)          Initial    Final    LS-Inc  Type Reps    C-Inc
  1    F2T                   1.000     1.000     600.0    1    2     1.000
  2    FG                    1.000     1.000     0.000    2    1     0.000
  3    NULL                  0.000    40.000     0.500    1    1     0.000
  4    NULL                  0.000     0.000     0.000    1    1     0.000
  5    NULL                  0.000     0.000     0.000    1    1     0.000

ANALYSIS PARAMETERS
-------------------
Analysis Mode                      (1-2) : 1
Seed File Name                (8 char max) : NULL
Convergence Limit                 (>1.0) : 1.00001
Averaging Factor                  (<1.0) : 0.5
Maximum Iterations                       : 50
Convergence Criteria               (1-5) : 2
Results Files                      (1-4) : 2
Output Format                      (1-3) : 1

MATERIAL BEHAVIOUR MODELS
-------------------------
Concrete Compression Base Curve    (0-3) : 1
Concrete Compression Post-Peak     (0-3) : 1
Concrete Compression Softening     (0-8) : 1
Concrete Tension Stiffening        (0-5) : 1
Concrete Tension Softening         (0-3) : 1
Concrete Tension Splitting         (1-2) : 1
Concrete Confined Strength         (0-2) : 1
Concrete Dilatation                (0-1) : 1
Concrete Cracking Criterion        (0-4) : 1
Concrete Crack Slip Check          (0-2) : 1
Concrete Crack Width Check         (0-2) : 1
Concrete Bond or Adhesion          (0-4) : 1
Concrete Creep and Relaxation      (0-1) : 1
Concrete Hysteresis                (0-3) : 2
Reinforcement Hysteresis           (0-3) : 1
Reinforcement Dowel Action         (0-1) : 1
Reinforcement Buckling             (0-1) : 1
Element Strain Histories           (0-1) : 1
Element Slip Distortions           (0-4) : 1
Strain Rate Effects                (0-1) : 1
Structural Damping                 (0-1) : 1
Geometric Nonlinearity             (0-1) : 1

Crack Allocation Process           (0-1) : 1

<NOTES:>
```

VECTOR-III.JOB

```
                              * * * * * * * * * * * *
                              *     V e c T o r     *
                              *   J O B   D A T A   *
                              * * * * * * * * * * * *

Job Title       (30 char. max.)        : Frame
Job File Name   ( 8 char. max.)        : FIII
Date            (30 char. max.)        : Nov 13, 2003

STRUCTURE DATA
--------------
Structure Type                         : 2
File Name       ( 8 char. max.)        : F3R

LOADING DATA
------------
No. of Load Stages                     : 60
Starting Load Stage No.                : 1
Load Series ID  ( 5 char. max.)        : FIII

Load   File Name         |                    Factors                    |
Case   (8 char max)          Initial    Final    LS-Inc  Type Reps    C-Inc
  1    F3T                   1.000     1.000     600.0    1    2     1.000
  2    FD                    0.000     60.00     1.000    1    1     0.000
  3    NULL                  0.000    40.000     1.000    1    1     0.000
  4    NULL                  0.000     0.000     0.000    1    1     0.000
  5    NULL                  0.000     0.000     0.000    1    1     0.000

ANALYSIS PARAMETERS
-------------------

Analysis Mode                          (1-2) : 1
Seed File Name                 (8 char max) : NULL
Convergence Limit                     (>1.0) : 1.00001
Averaging Factor                      (<1.0) : 0.5
Maximum Iterations                           : 50
Convergence Criteria                   (1-5) : 2
Results Files                          (1-4) : 2
Output Format                          (1-3) : 1

MATERIAL BEHAVIOUR MODELS
-------------------------
Concrete Compression Base Curve        (0-3) : 1
Concrete Compression Post-Peak         (0-3) : 1
Concrete Compression Softening         (0-8) : 1
Concrete Tension Stiffening            (0-5) : 1
Concrete Tension Softening             (0-3) : 1
Concrete Tension Splitting             (1-2) : 1
Concrete Confined Strength             (0-2) : 1
Concrete Dilatation                    (0-1) : 1
Concrete Cracking Criterion            (0-4) : 1
Concrete Crack Slip Check              (0-2) : 1
Concrete Crack Width Check             (0-2) : 1
Concrete Bond or Adhesion              (0-4) : 1
Concrete Creep and Relaxation          (0-1) : 1
Concrete Hysteresis                    (0-3) : 2
Reinforcement Hysteresis               (0-3) : 1
Reinforcement Dowel Action             (0-1) : 1
Reinforcement Buckling                 (0-1) : 1
Element Strain Histories               (0-1) : 1
Element Slip Distortions               (0-4) : 1
Strain Rate Effects                    (0-1) : 1
Structural Damping                     (0-1) : 1
Geometric Nonlinearity                 (0-1) : 1
Crack Allocation Process               (0-1) : 1

<NOTES:>
```

B1T.L2R

```
                        * * * * * * * * * * * * *
                        *     V e c T o r 2     *
                        *   L O A D   D A T A   *
                        * * * * * * * * * * * * *

                          LOAD CASE PARAMETERS
                          ********************

Structure Title      (30 char. max.)     : frame
Load Case Title      (30 char. max.)     : temp
Load Case File Name   (8 char. max.)     : F1T
No. of Loaded Joints                     : 0
No. of Prescribed Support Displacements  : 0
No. of Elements with Gravity Forces      : 0
No. of Elements with Temperature Change  : 0
No. of Elements with Concrete Prestrain  : 0
No. of Elements with Ingress Pressure    : 0
No. of Nodes with Thermal Load           : 268
No. of Nodes with Lumped Masses          : 0
No. of Nodes with Impulse Forces         : 0
Ground Acceleration Record  (0-1)        : 0

                             JOINT LOADS
                             ***********
<NOTE:>  UNITS:  KN
<<<<< FORMAT >>>>>
NODE    Fx    Fy    [ #NODE d(NODE) d(Fx) d(Fy) ] /
/
                         SUPPORT DISPLACEMENTS
                         *********************
<NOTE:> UNITS: MM
<<<<< FORMAT >>>>>
JNT  DOF  DISPL   [ #JNT d(JNT) ] /
/
                            GRAVITY LOADS
                            *************
<NOTE:>  UNITS:  KG/M3
<<<<< FORMAT >>>>>
ELMT  DENS  GX  GY  [#ELMT d(ELMT)] [ #ELMT d(ELMT)] /
/
                           TEMPERATURE LOADS
                           *****************
<NOTE:>  UNITS:  C
<<<<< FORMAT >>>>>
ELMT   TEMP   [ #ELMT d(ELMT) d(TEMP) ] [ #ELMT d(ELMT) d(TEMP) ] /
/
                          CONCRETE PRESTRAINS
                          *******************
<NOTE:>  UNITS:  me
<<<<< FORMAT >>>>>
ELMT  STRAIN   [ #ELMT d(ELMT) d(STRAIN) ] [ #ELMT d(ELMT) d(STRAIN) ]  /
/
                           INGRESS PRESSURES
                           *****************
<NOTE:>  UNITS:  MPa
<<<<< FORMAT >>>>>
ELMT PRESSURE  [ #ELMT d(ELMT) d(PRS) ] [ #ELMT d(ELMT) d(PRS) ]  /
/
                          NODAL THERMAL LOADS
                          *******************
<NOTE:>  UNITS:  Sec, Degrees C
<<<<< FORMAT >>>>>
NODE  TYPE  Tm1 Tp1  Tm2 Tp2  Tm3 Tp3  [#NODE d(NODE)] [#NODE d(NODE)]  /
521 2 1 15 7200 95 28800 95 41 1 /
572 2 1 15 7200 95 28800 95 64 11 /
1276 2 1 15 7200 15 28800 15 10 11 /
1366 2 1 15 7200 15 28800 15 9 1 /
562 2 1 15 7200 15 28800 15 74 11 /
103 2 1 15 7200 15 28800 15 9 51 /
2 2 1 15 7200 15 28800 15 1 1 /
```

```
1 2 1 15 7200 15 28800 15 51 2 /
102 2 1 15 7200 15 28800 15 9 51 /
/
                            LUMPED MASSES
                            *************
<NOTE:>  UNITS:  kg, m/s
<<<<< FORMAT >>>>>
NODE  DOF-X  DOF-Y  MASS  GF-X  GF-Y  Vo-X  Vo-Y  [ #NODE d(NODE) ] /
/
                           IMPULSE FORCES
                           **************
<NOTE:>  UNITS:  Sec, kN
<<<<< FORMAT >>>>>
NODE  DOF  T1   F1   T2   F2   T3   F3   T4   F4  [ #NODE d(NODE) ] /
/
                          GROUND ACCELERATION
                          *******************
<NOTE:>  UNITS:  Sec, m/s2
<<<<< FORMAT >>>>>
TIME   ACC-X   ACC-Y
/


<NOTES:>
```

B2T.L2R

```
                        * * * * * * * * * * * * *
                        *     V e c T o r 2     *
                        *   L O A D   D A T A   *
                        * * * * * * * * * * * * *

                          LOAD CASE PARAMETERS
                          ********************

Structure Title      (30 char. max.)     : frame
Load Case Title      (30 char. max.)     : temp
Load Case File Name   (8 char. max.)     : F2T
No. of Loaded Joints                     : 0
No. of Prescribed Support Displacements  : 0
No. of Elements with Gravity Forces      : 0
No. of Elements with Temperature Change  : 0
No. of Elements with Concrete Prestrain  : 0
No. of Elements with Ingress Pressure    : 0
No. of Nodes with Thermal Load           : 268
No. of Nodes with Lumped Masses          : 0
No. of Nodes with Impulse Forces         : 0
Ground Acceleration Record  (0-1)        : 0

                             JOINT LOADS
                             ***********
<NOTE:>  UNITS:  KN
<<<<< FORMAT >>>>>
NODE    Fx    Fy    [ #NODE d(NODE) d(Fx) d(Fy) ] /
/
                         SUPPORT DISPLACEMENTS
                         *********************
<NOTE:> UNITS: MM
<<<<< FORMAT >>>>>
JNT  DOF  DISPL   [ #JNT d(JNT) ] /
/
                            GRAVITY LOADS
                            *************
<NOTE:>  UNITS:  KG/M3
<<<<< FORMAT >>>>>
ELMT  DENS  GX  GY  [#ELMT d(ELMT)] [ #ELMT d(ELMT)] /
/
                           TEMPERATURE LOADS
                           *****************
<NOTE:>  UNITS:  C
<<<<< FORMAT >>>>>
ELMT   TEMP   [ #ELMT d(ELMT) d(TEMP) ] [ #ELMT d(ELMT) d(TEMP) ] /
/
                          CONCRETE PRESTRAINS
                          *******************
<NOTE:>  UNITS:  me
<<<<< FORMAT >>>>>
ELMT  STRAIN   [ #ELMT d(ELMT) d(STRAIN) ] [ #ELMT d(ELMT) d(STRAIN) ]  /
/
                           INGRESS PRESSURES
                           *****************
<NOTE:>  UNITS:  MPa
<<<<< FORMAT >>>>>
ELMT PRESSURE  [ #ELMT d(ELMT) d(PRS) ] [ #ELMT d(ELMT) d(PRS) ]  /
/
                          NODAL THERMAL LOADS
                          *******************
<NOTE:>  UNITS:  Sec, Degrees C
<<<<< FORMAT >>>>>
NODE  TYPE  Tm1 Tp1  Tm2 Tp2  Tm3 Tp3  [#NODE d(NODE)] [#NODE d(NODE)]  /
521 2 1 15 10800 45 28800 45 41 1 /
572 2 1 15 10800 45 28800 45 64 11 /
1276 2 1 15 7200 15 28800 15 10 11 /
1366 2 1 15 7200 15 28800 15 9 1 /
562 2 1 15 7200 15 28800 15 74 11 /
103 2 1 15 7200 15 28800 15 9 51 /
2 2 1 15 7200 15 28800 15 1 1 /
```

```
1 2 1 15 7200 15 28800 15 51 2 /
102 2 1 15 7200 15 28800 15 9 51 /
/
                            LUMPED MASSES
                            *************
<NOTE:>  UNITS:  kg, m/s
<<<<< FORMAT >>>>>
NODE  DOF-X  DOF-Y  MASS  GF-X  GF-Y  Vo-X  Vo-Y  [ #NODE d(NODE) ] /
/
                           IMPULSE FORCES
                           **************
<NOTE:>  UNITS:  Sec, kN
<<<<< FORMAT >>>>>
NODE  DOF  T1   F1   T2   F2   T3   F3   T4   F4  [ #NODE d(NODE) ] /
/
                          GROUND ACCELERATION
                          *******************
<NOTE:>  UNITS:  Sec, m/s2
<<<<< FORMAT >>>>>
TIME   ACC-X   ACC-Y
/


<NOTES:>
```

B3T.L2R

```
                        * * * * * * * * * * * * *
                        *     V e c T o r 2     *
                        *   L O A D   D A T A   *
                        * * * * * * * * * * * * *

                          LOAD CASE PARAMETERS
                          ********************

Structure Title      (30 char. max.)     : frame
Load Case Title      (30 char. max.)     : temp
Load Case File Name   (8 char. max.)     : F3T
No. of Loaded Joints                     : 0
No. of Prescribed Support Displacements  : 0
No. of Elements with Gravity Forces      : 0
No. of Elements with Temperature Change  : 0
No. of Elements with Concrete Prestrain  : 0
No. of Elements with Ingress Pressure    : 0
No. of Nodes with Thermal Load           : 268
No. of Nodes with Lumped Masses          : 0
No. of Nodes with Impulse Forces         : 0
Ground Acceleration Record  (0-1)        : 0

                             JOINT LOADS
                             ***********
<NOTE:>  UNITS:  KN
<<<<< FORMAT >>>>>
NODE    Fx    Fy    [ #NODE d(NODE) d(Fx) d(Fy) ] /
/
                         SUPPORT DISPLACEMENTS
                         *********************
<NOTE:> UNITS: MM
<<<<< FORMAT >>>>>
JNT  DOF  DISPL   [ #JNT d(JNT) ] /
/
                            GRAVITY LOADS
                            *************
<NOTE:>  UNITS:  KG/M3
<<<<< FORMAT >>>>>
ELMT  DENS  GX  GY  [#ELMT d(ELMT)] [ #ELMT d(ELMT)] /
/
                           TEMPERATURE LOADS
                           *****************
<NOTE:>  UNITS:  C
<<<<< FORMAT >>>>>
ELMT   TEMP   [ #ELMT d(ELMT) d(TEMP) ] [ #ELMT d(ELMT) d(TEMP) ] /
/
                          CONCRETE PRESTRAINS
                          *******************
<NOTE:>  UNITS:  me
<<<<< FORMAT >>>>>
ELMT  STRAIN   [ #ELMT d(ELMT) d(STRAIN) ] [ #ELMT d(ELMT) d(STRAIN) ]  /
/
                           INGRESS PRESSURES
                           *****************
<NOTE:>  UNITS:  MPa
<<<<< FORMAT >>>>>
ELMT PRESSURE  [ #ELMT d(ELMT) d(PRS) ] [ #ELMT d(ELMT) d(PRS) ]  /
/
                          NODAL THERMAL LOADS
                          *******************
<NOTE:>  UNITS:  Sec, Degrees C
<<<<< FORMAT >>>>>
NODE  TYPE  Tm1 Tp1  Tm2 Tp2  Tm3 Tp3  [#NODE d(NODE)] [#NODE d(NODE)]  /
521 2 1 15 14400 75 28800 75 41 1 /
572 2 1 15 14400 75 28800 75 64 11 /
1276 2 1 15 7200 15 28800 15 10 11 /
1366 2 1 15 7200 15 28800 15 9 1 /
562 2 1 15 7200 15 28800 15 74 11 /
103 2 1 15 7200 15 28800 15 9 51 /
2 2 1 15 7200 15 28800 15 1 1 /
```

```
1 2 1 15 7200 15 28800 15 51 2 /
102 2 1 15 7200 15 28800 15 9 51 /
/
                            LUMPED MASSES
                            *************
<NOTE:>  UNITS:  kg, m/s
<<<<< FORMAT >>>>>
NODE  DOF-X  DOF-Y  MASS  GF-X  GF-Y  Vo-X  Vo-Y  [ #NODE d(NODE) ] /
/
                           IMPULSE FORCES
                           **************
<NOTE:>  UNITS:  Sec, kN
<<<<< FORMAT >>>>>
NODE  DOF  T1   F1   T2   F2   T3   F3   T4   F4  [ #NODE d(NODE) ] /
/
                          GROUND ACCELERATION
                          *******************
<NOTE:>  UNITS:  Sec, m/s2
<<<<< FORMAT >>>>>
TIME   ACC-X   ACC-Y
/


<NOTES:>
```

BD.L2R

```
                        * * * * * * * * * * * * *
                        *     V e c T o r 2     *
                        *   L O A D   D A T A   *
                        * * * * * * * * * * * * *

                          LOAD CASE PARAMETERS
                          ********************

Structure Title      (30 char. max.)     : Enter Structure Title
Load Case Title      (30 char. max.)     : Enter load case title
Load Case File Name   (8 char. max.)     : FD
No. of Loaded Joints                     : 0
No. of Prescribed Support Displacements  : 1
No. of Elements with Gravity Loads       : 0
No. of Elements with Temperature Loads   : 0
No. of Elements with Concrete Prestrain  : 0
No. of Elements with Ingress Pressure    : 0
No. of Element Surfaces w/ Thermal Load  : 0
No. of Nodes with Lumped Masses          : 0
No. of Nodes with Impulse Forces         : 0
Ground Acceleration Record  (0-1)        : 0

                             JOINT LOADS
                             ***********
<NOTE:>  UNITS:  KIPS OR KN
<<<<< FORMAT >>>>>
NODE    Fx    Fy    [ #NODE d(NODE) d(Fx) d(Fy) ] /
/
                         SUPPORT DISPLACEMENTS
                         *********************
<NOTE:> UNITS: MM OR IN
<<<<< FORMAT >>>>>
JNT  DOF  DISPL   [ #JNT d(JNT) ] /
738 1 1.000/
/
                            GRAVITY LOADS
                            *************
<NOTE:>  UNITS:  KG/M3
<<<<< FORMAT >>>>>
ELMT  DENS  GX  GY  [#ELMT d(ELMT)] [ #ELMT d(ELMT)] /
/
                           TEMPERATURE LOADS
                           *****************
<NOTE:>  UNITS:  F OR C
<<<<< FORMAT >>>>>
ELMT   TEMP   [ #ELMT d(ELMT) d(TEMP) ] [ #ELMT d(ELMT) d(TEMP) ] /
/
                          CONCRETE PRESTRAINS
                          *******************
<NOTE:>  UNITS:  me
<<<<< FORMAT >>>>>
ELMT  STRAIN   [ #ELMT d(ELMT) d(STRAIN) ] [ #ELMT d(ELMT) d(STRAIN) ]  /
/
                           INGRESS PRESSURES
                           *****************
<NOTE:>  UNITS:  MPa
<<<<< FORMAT >>>>>
ELMT PRESSURE  [ #ELMT d(ELMT) d(PRS) ] [ #ELMT d(ELMT) d(PRS) ]  /
/
                         SURFACE THERMAL LOADS
                         *********************
<NOTE:>  UNITS:  Sec, Degrees C
<<<<< FORMAT >>>>>
NODE1 NODE2  Tm1 Tp1  Tm2 Tp2  Tm3 Tp3  [#SURF d(NODE)] [#SURF d(NODE)]  /
/
                            LUMPED MASSES
                            *************
<NOTE:>  UNITS:  kg, m/s
<<<<< FORMAT >>>>>
NODE  DOF-X  DOF-Y  MASS  GF-X  GF-Y  Vo-X  Vo-Y  [ #NODE d(NODE) ] /
```

```
/
                            IMPULSE FORCES
                            **************
<NOTE:>  UNITS:  Sec, kN
<<<<< FORMAT >>>>>
NODE  DOF  T1   F1   T2   F2   T3   F3   T4   F4  [ #NODE d(NODE) ] /
/
                           GROUND ACCELERATION
                           *******************
<NOTE:>  UNITS:  Sec, G
<<<<< FORMAT >>>>>
TIME   ACC-X   ACC-Y
/


<<< LOAD FILE NOTES >>>
```

BG.L2R

```
                        * * * * * * * * * * * * *
                        *     V e c T o r 2     *
                        *   L O A D   D A T A   *
                        * * * * * * * * * * * * *

                          LOAD CASE PARAMETERS
                          ********************

Structure Title      (30 char. max.)     : Frame
Load Case Title      (30 char. max.)     : Gravity
Load Case File Name   (8 char. max.)     : BG
No. of Loaded Joints                     : 0
No. of Prescribed Support Displacements  : 0
No. of Elements with Gravity Forces      : 1240
No. of Elements with Temperature Change  : 0
No. of Elements with Concrete Prestrain  : 0
No. of Elements with Ingress Pressure    : 0
No. of Nodes with Thermal Load           : 0
No. of Nodes with Lumped Masses          : 0
No. of Nodes with Impulse Forces         : 0
Ground Acceleration Record  (0-1)        : 0

                             JOINT LOADS
                             ***********
<NOTE:>  UNITS:  KN
<<<<< FORMAT >>>>>
NODE    Fx    Fy    [ #NODE d(NODE) d(Fx) d(Fy) ] /
/
                         SUPPORT DISPLACEMENTS
                         *********************
<NOTE:> UNITS: MM
<<<<< FORMAT >>>>>
JNT  DOF  DISPL   [ #JNT d(JNT) ] /
/
                            GRAVITY LOADS
                            *************
<NOTE:>  UNITS:  KG/M3
<<<<< FORMAT >>>>>
ELMT  DENS  GX  GY  [#ELMT d(ELMT)] [ #ELMT d(ELMT)] /
  1   1   0   1   1240  1 /
/
                           TEMPERATURE LOADS
                           *****************
<NOTE:>  UNITS:  C
<<<<< FORMAT >>>>>
ELMT   TEMP   [ #ELMT d(ELMT) d(TEMP) ] [ #ELMT d(ELMT) d(TEMP) ] /
/
                          CONCRETE PRESTRAINS
                          *******************
<NOTE:>  UNITS:  me
<<<<< FORMAT >>>>>
ELMT  STRAIN   [ #ELMT d(ELMT) d(STRAIN) ] [ #ELMT d(ELMT) d(STRAIN) ]  /
/
                           INGRESS PRESSURES
                           *****************
<NOTE:>  UNITS:  MPa
<<<<< FORMAT >>>>>
ELMT PRESSURE  [ #ELMT d(ELMT) d(PRS) ] [ #ELMT d(ELMT) d(PRS) ]  /
/
                          NODAL THERMAL LOADS
                          *******************
<NOTE:>  UNITS:  Sec, Degrees C
<<<<< FORMAT >>>>>
NODE  TYPE  Tm1 Tp1  Tm2 Tp2  Tm3 Tp3  [#NODE d(NODE)] [#NODE d(NODE)]  /
/
                            LUMPED MASSES
                            *************
<NOTE:>  UNITS:  kg, m/s
<<<<< FORMAT >>>>>
NODE  DOF-X  DOF-Y  MASS  GF-X  GF-Y  Vo-X  Vo-Y  [ #NODE d(NODE) ] /
```

```
/
                            IMPULSE FORCES
                            **************
<NOTE:>  UNITS:  Sec, kN
<<<<< FORMAT >>>>>
NODE  DOF  T1   F1   T2   F2   T3   F3   T4   F4  [ #NODE d(NODE) ] /
/
                           GROUND ACCELERATION
                           *******************
<NOTE:>  UNITS:  Sec, m/s2
<<<<< FORMAT >>>>>
TIME   ACC-X   ACC-Y
/


<NOTES:>
```

Appendix C V2HEAT Source Code

```
      SUBROUTINE HEAT(MOPT,TTIME,STIME,DTIME)
C     ---------------------------------------
C     This subprogram computes element temperatures based on two-
C     dimensional transient nonlinear heat flow analysis in which
C     bilinear 4-node quadrilateral and linear 3-node triangular
C     elements are employed.
C
      PARAMETER (MELM=5000,MTYP=25,MJNT=5200,MBOUN=299,MPROF=99999)
      IMPLICIT REAL (A-H,O-Z)
      COMMON /STR PARA/STRID,SUNITS,NRC,NS,NB,NREC,NQUAD,NTRIG,NTRUSS,
     *        NLINK,NCONT,NJ,NR,NELEMT
      COMMON /CONC SPECIFICATIONS/FC,FT,EC,E0,MU,CEC,T,AGG,
     *        SCRX,SCRY,SMA,URF,DENS,DFSV,NSC,TBS
      COMMON /LTYPE/ LTYP
      COMMON /INCIDENCES/INC,MAT
      COMMON /COORDINATES/X,Y
      COMMON /THERMAL1/ TEMPN,TEMPL,TEMP
      COMMON /THERMAL3/ NTHER,NLOAD,TLOAD
      COMMON /REDUCTION/ TRC,TRS
      COMMON /QUAD/ QINC,QAGS,THK,AGS,CGX,CGY

      INTEGER MOP
      INTEGER NRC,NS,NB,NREC,NQUAD,NTRIG,NTRUSS,NLINK,NCONT
      INTEGER NJ,NR,NELEMT
      INTEGER INC(MELM,6),MAT(MELM),LTYP(MELM)
      INTEGER NSC(MTYP)
      INTEGER NTHER,NLOAD(MJNT)
      INTEGER QINC(MELM,4)
      REAL QAGS(MELM,2),THK(MELM)
      REAL AGS(MELM),CGX(MELM),CGY(MELM)
      REAL FC(MTYP),FT(MTYP),EC(MTYP),MU(MTYP),CEC(MTYP),T(MTYP)
      REAL SCRX(MTYP),SCRY(MTYP),SMA(MTYP),AGG(MTYP)
      REAL URF(MTYP),DENS(MTYP),E0(MTYP),DFSV(MTYP)
      REAL TBS(MTYP)
      REAL X(MJNT),Y(MJNT)
      REAL TEMPL(MELM),TEMP(MELM),TEMPN(MJNT)
      REAL TLOAD(MJNT)
      REAL TTIME,STIME,DTIME
      REAL TRC(MELM,4), TRS(MELM,4)
      CHARACTER*30 STRID
      CHARACTER*10 SUNITS

C     INTRODUCED IN PART 1
      INTEGER NNPEL(MELM),NCONC(MELM,4),MTYPE(MELM),ECONC(MELM)
      INTEGER NITER,NMATR,ITRAN,NNPFC,NPOIN,NELEM
      REAL ALPHA,TOLER,RELAX,COORD(MJNT,2),DEN(MTYP),DIFS(MTYP)
C     INTRODUCED IN PART 2
      INTEGER NFIXB,NFIXD(MBOUN),IFFIX(MJNT)
      INTEGER NPROF,NCOLM(MJNT),NDIAG(MJNT)
      REAL FIXED(MBOUN),TEMPR(MJNT),TFIXD(MJNT)
C     INTRODUCED IN PART 3 OR 4
      REAL ASTIF(4,4),AMASS(4,4),TLAST(MJNT),GSTIF(MPROF),EFORC(4)
      REAL FORCE(MJNT),TEMP1(4,4),RVECT(4)
      INTEGER NDFEL(4)
C     LOCAL VARIABLES
      INTEGER I,IELEM,INOD,IMAT,IBC,J,JELEM,KELEM
C
C     PART  11111            INITIALIZATION OPERATION              11111
C
C *** MESH INFORMATION ADAPTION
      NNPFC=2
      NPOIN=NJ
```

```
      NMATR=NRC

      IELEM=0
      DO 4 JELEM=1,NELEMT
        IF ((LTYP(JELEM).LE.3) .AND. (LTYP(JELEM).GE.1)) THEN
          IELEM=IELEM+1
          ECONC(IELEM)=JELEM
        ENDIF
  4   CONTINUE
      NELEM=IELEM

      DO 1 IELEM=1,NELEM
        JELEM=ECONC(IELEM)
        IF ((LTYP(JELEM).EQ.1).OR.(LTYP(JELEM).EQ.2)) THEN
         NNPEL(IELEM)=4
        ELSEIF (LTYP(JELEM).EQ.3) THEN
         NNPEL(IELEM)=3
        ENDIF
      MTYPE(IELEM)=MAT(JELEM)
  1   CONTINUE

      DO 2 IELEM=1,NELEM
        JELEM=ECONC(IELEM)
        DO 3 INOD=1,NNPEL(IELEM)
         IF (LTYP(JELEM).NE.2) THEN
          NCONC(IELEM,INOD)=INC(JELEM,INOD)
         ELSE
          NCONC(IELEM,INOD)=QINC(JELEM,INOD)
         ENDIF
  3     CONTINUE
  2   CONTINUE

      DO INOD=1,NPOIN
        COORD(INOD,1)=X(INOD)/1000
        COORD(INOD,2)=Y(INOD)/1000
      END DO

      IF (NELEM.NE.NREC+NQUAD+NTRIG) THEN
      WRITE (*,*) 'MESH-IN PROCESS IN V2HEAT IS WRONG!'
      STOP
      END IF

      IF (TTIME.LE.1.0E-3) THEN
       DO I=1,NPOIN
        TEMPR(I)=0.0
       END DO
       GOTO 501
      ENDIF

C *** CONTROAL INDICES SPECIFICATION

      !ALPHA:TIME STEPPING FACTOR CORRESPONDS TO:
C           1/2: CRANK-NICOLSON SCHEME (ACCURATE)
C           2/3: GALERKIN SCHEME (STABLE)
      ALPHA=2.0/3
      NITER=100
      TOLER=1.E-6
      !RELAX:RELAXATION FACTOR FOR NONLINEAR PROBLEM
      RELAX=1.0
C *** MATERIAL PROPERTIES ADAPTION
      DO IMAT=1,NMATR
      DEN(IMAT)=DENS(IMAT)
      DIFS(IMAT)=DFSV(IMAT)
      END DO
C *** ADD INTERNAL HEAT RESOURCE HERE IF APPLICABLE
C
      IF (MOPT.EQ.1) THEN
        ITRAN=0
      ELSE
        ITRAN=1
      ENDIF
```

```
C
C     PART 22222     PRESCRIBE INITIAL AND BOUNDARY CONDITIONS        22222
C
C *** DIRICHLET BOUNDARY CONDITIONS (FIXED TEMPERATURE)
C     TREATED AS NODAL THERMAL LOAD IN DR. VECCHIO'S DATE DESIGN
      NFIXB=NTHER
      DO 205 IBC=1,NTHER
      NFIXD(IBC)=NLOAD(IBC)
      FIXED(IBC)=TLOAD(IBC)
 205  CONTINUE
C *** ADD NEUMANN BOUNDARY CONDITIONS (FIXED FLUX) HERE IF APPLICABLE
C *** INITIAL TEMPERATURE FIELD FOR NON-LINEAR AND/OR TRANSIENT PROBLEMS
      DO 206 INOD=1,NPOIN
      TEMPR(INOD)=TEMPN(INOD)
 206  CONTINUE
      DO 207 IBC=1,NFIXB
      TEMPR(NFIXD(IBC))=FIXED(IBC)
 207  CONTINUE
C *** SET UP ARRAYS FOR FIXED BCs AND INITIALISE VECTOR NCOLM
      DO 208 I=1,NPOIN
      IFFIX(I)=0
      NCOLM(I)=1
 208  CONTINUE
      DO 209 I=1,NFIXB
      NOD=NFIXD(I)
      IFFIX(NOD)=1
      TFIXD(NOD)=FIXED(I)
 209  CONTINUE
C *** SET UP THE VECTOR NCOLM,NDIAG AND CALCULATE NPROF
      CALL DIAGNL(MELM,NELEM,NPOIN,NNPEL,MJNT,NCOLM,
     -            NCONC,NDIAG,NPROF,IFFIX)
      IF (ITRAN.EQ.1) GOTO 300
C
C     PART 33333          PERFORM  STEADY-STATE  ANALYSIS             33333
C
      CALL STEADY(MELM,MJNT,MTYP,MPROF,TOLER,RELAX,
     -            ITRAN,NITER,NELEM,NPOIN,NNPEL,
     -            NPROF,NDFEL,NCONC,NDIAG,IFFIX,TFIXD,
     -            ASTIF,AMASS,GSTIF,EFORC,FORCE,TLAST,
     -            TEMPR,COORD,DEN,DIFS,MTYPE)
      GOTO 501
C
C     PART 44444              PERFORM  TRANSIENT  ANALYSIS            44444
C
 300  CONTINUE
      CALL TRANSI(MELM,MJNT,MPROF,TOLER,ITRAN,NITER,
     -            NELEM,NPOIN,NNPEL,NPROF,NDFEL,NCONC,
     -            NDIAG,IFFIX,TFIXD,ASTIF,AMASS,GSTIF,
     -            EFORC,FORCE,TLAST,TEMPR,COORD,DTIME,
     -            ALPHA,TEMP1,RVECT,MTYP,MTYPE,DEN,DIFS)
C
C     PART 55555     OUTPUT RESULTS(TEMP. ON NODES AND CENTROIDS)     55555
C
C     RESTORE RESULTS IN 'TEMPN' FROM 'TEMPR'
 501  CONTINUE
      DO INOD=1,NPOIN
      TEMPN(INOD)=TEMPR(INOD)
C      IF (TEMPN(INOD).LT.1.E-6) TEMPN(INOD)=0.0
      END DO
C     CALCULATE TEMPERATURE ON CENTROID OF ELEMENTS
      IF (STIME.LT.TTIME) GOTO 502
      IELEM=0
      DO 8 JELEM=1,NELEMT
        IF ((LTYP(JELEM).LE.4) .AND. (LTYP(JELEM).GE.1)) THEN
          IELEM=IELEM+1
          ECONC(IELEM)=JELEM
        ENDIF
  8   CONTINUE
      NELEM=IELEM
      DO 5 IELEM=1,NELEM
        JELEM=ECONC(IELEM)
```

```
        IF ((LTYP(JELEM).EQ.1).OR.(LTYP(JELEM).EQ.2)) THEN
         NNPEL(IELEM)=4
        ELSEIF (LTYP(JELEM).EQ.3) THEN
         NNPEL(IELEM)=3
        ELSEIF (LTYP(I).EQ.4) THEN
         NNPEL(IELEM)=2
        ENDIF
  5   CONTINUE
      DO 6 IELEM=1,NELEM
        JELEM=ECONC(IELEM)
        DO 7 INOD=1,NNPEL(IELEM)
         IF (LTYP(JELEM).NE.2) THEN
          NCONC(IELEM,INOD)=INC(JELEM,INOD)
         ELSE
          NCONC(IELEM,INOD)=QINC(JELEM,INOD)
         ENDIF
  7     CONTINUE
  6   CONTINUE
      IF (NELEM.NE.NREC+NQUAD+NTRIG+NTRUSS) THEN
      WRITE (*,*) 'MESH-OUT PROCESS IN V2HEAT IS WRONG!'
      STOP
      END IF
      DO 92 I=1,NELEM
       JELEM=ECONC(I)
       TEMPL(I)=0.0
       DO 93 J=1,NNPEL(I)
        NOD=NCONC(I,J)
        TEMPL(JELEM)=TEMPL(JELEM)+TEMPN(NOD)/NNPEL(I)
  93   CONTINUE
  92  CONTINUE
C
 502  RETURN
      END

      SUBROUTINE DIAGNL(MELM,NELEM,NPOIN,NNPEL,MJNT,NCOLM,
     -                  NCONC,NDIAG,NPROF,IFFIX)
C     ------------------------------------------------------
C                                                                       DIAGNL
C     Set up the vector 'NDIAG' storing the diagonal dof number for     DIAGNL
C     each column of the global stiffness matrix.                       DIAGNL
C                                                                       DIAGNL
C
      IMPLICIT REAL(A-H,O-Z)
      INTEGER NCONC(MELM,4),NCOLM(MJNT),NDIAG(MJNT),
     -          IFFIX(MJNT),NNPEL(MELM)
      DO 45 I=1,NELEM
      N1=NPOIN
       DO 46 J=1,NNPEL(I)
       K=NCONC(I,J)
       IF (IFFIX(K).EQ.1) GOTO 47
       IF (K.LT.N1) N1=K
   47  CONTINUE
   46  CONTINUE
       DO 48 J=1,NNPEL(I)
       K=NCONC(I,J)
       IF (IFFIX(K).EQ.1) GOTO 49
       N2=K-N1+1
       IF (NCOLM(K).LT.N2) NCOLM(K)=N2
   49  CONTINUE
   48  CONTINUE
   45 CONTINUE
      NDIAG(1)=1
      DO 50 J=2,NPOIN
       NDIAG(J)=NDIAG(J-1)+NCOLM(J)
   50 CONTINUE
      NPROF=NDIAG(NPOIN)
C
      RETURN
      END

      SUBROUTINE STEADY (MELM,MJNT,MTYP,MPROF,TOLER,RELAX,
```

```
     -                  ITRAN,NITER,NELEM,NPOIN,NNPEL,
     -                  NPROF,NDFEL,NCONC,NDIAG,IFFIX,TFIXD,
     -                  ASTIF,AMASS,GSTIF,EFORC,FORCE,TLAST,
     -                  TEMPR,COORD,DEN,DIFS,MTYPE)
C     ------------------------------------------------------
C                                                                       STEADY
C     This subroutine performs steady state analysis                    STEADY
C                                                                       STEADY
      IMPLICIT REAL(A-H,O-Z)
      REAL ASTIF(4,4),AMASS(4,4),DEN(MTYP),DIFS(MTYP)
      REAL EFORC(4),XCORD(4),YCORD(4),TFIXD(MJNT),COORD(MJNT,2),
     -     GSTIF(MPROF),FORCE(MJNT),TLAST(MJNT),TEMPR(MJNT)
      INTEGER IFFIX(MJNT),NCONC(MELM,4),NDFEL(4),NDIAG(MJNT),
     -        NNPEL(MELM),MTYPE(MELM)
C
C ***  PERFORM  STEADY STATE  ANALYSIS
C
      DO 119 ITER=1,NITER
      JITER=ITER
      DO 120 I=1,NPOIN
      FORCE(I)=0.0
      TLAST(I)=TEMPR(I)
  120 CONTINUE
      DO 121 I=1,NPROF
      GSTIF(I)=0.0
  121 CONTINUE
      DO 122 IELEM=1,NELEM
      IMATR=MTYPE(IELEM)
      CALL PROPTY(IELEM,MELM,MJNT,NNPEL,TEMPR,DEN,DIFS,MTYP,
     -            NCONC,DENST,CONDT,DIFST,CAPCT,IMATR)
      DO 123 I=1,NNPEL(IELEM)
      NDFEL(I)=NCONC(IELEM,I)
      ID=NDFEL(I)
      EFORC(I)=0.0
      XCORD(I)=COORD(ID,1)
      YCORD(I)=COORD(ID,2)
      DO 124 J=1,NNPEL(IELEM)
      ASTIF(I,J)=0.0
  124 CONTINUE
  123 CONTINUE
C
C *** CONSTRUCT ELEMENT STIFFNESS MATRIX
C
      CALL STIFFN (MELM,IELEM,NNPEL,ITRAN,DENST,CONDT,
     -             CAPCT,ASTIF,AMASS,XCORD,YCORD)
C
C *** CALCULATE NEUMANN BOUNDARY CONDITION EFFECTS ON THE FORCE VECTOR
C
C
C *** ADD (ELEMENT'S) INTERNAL VOLUMETRIC HEAT RESOURCE TO THE FORCE VECTOR
C
C
C *** TRANSFER ELEMENT FORCES TO GLOBAL FORCE VECTOR
C
      DO 125 J=1,NNPEL(IELEM)
      NOD=NDFEL(J)
      FORCE(NOD)=FORCE(NOD)+EFORC(J)
  125 CONTINUE
C
C *** MODIFY FORCE VECTOR TO ACCOUNT FOR FIXED TEMPERATURE NODES
C
      DO 126 J=1,NNPEL(IELEM)
      IROW=NDFEL(J)
      IF (IFFIX(IROW).EQ.1) GOTO 106
      DO 127 K=1,NNPEL(IELEM)
      ICOL=NDFEL(K)
      IF (IFFIX(ICOL).EQ.1) THEN
      FORCE(IROW)=FORCE(IROW)-ASTIF(J,K)*TFIXD(ICOL)
      ENDIF
  127 CONTINUE
  106 CONTINUE
```

```
  126 CONTINUE
C
C *** ASSEMBLE INTO GLOBAL MATRIX IN VECTOR FORM
C
      DO 128 J=1,NNPEL(IELEM)
      DO 129 K=J,NNPEL(IELEM)
      IROW=NDFEL(J)
      IF (IFFIX(IROW).EQ.1) GOTO 103
      ICOL=NDFEL(K)
      IF (IFFIX(ICOL).EQ.1) GOTO 103
      IF (IROW.LE.ICOL) GOTO 104
      ITEM=IROW
      IROW=ICOL
      ICOL=ITEM
  104 IRC=NDIAG(ICOL)-ICOL+IROW
      GSTIF(IRC)=GSTIF(IRC)+ASTIF(J,K)
  103 CONTINUE
  129 CONTINUE
  128 CONTINUE
  122 CONTINUE
C
C *** SOLVE THE FINAL SYSTEM USING PROFILE SOLVER
C
C 1.  SET DIAGONAL ELEMENTS OF GSTIF CORRESPONDING TO FIXED TEMPERATURE
C     NODES EQUAL TO UNITY AND THE FORCE VECTOR TO THE FIXED VALUE
C
      DO 130 I=1,NPOIN
      IF (IFFIX(I).EQ.1) THEN
      J=NDIAG(I)
      GSTIF(J)=1.0
      FORCE(I)=TFIXD(I)
      TEMPR(I)=TFIXD(I)
      ENDIF
  130 CONTINUE
C
C 2.  SOLVE THE FINAL SYSTEM USING PROFILE SOLVER
      CALL PROFAC (MPROF,MJNT,NPOIN,GSTIF,NDIAG)
      CALL PROSOL (MPROF,MJNT,NPOIN,GSTIF,FORCE,TEMPR,NDIAG)
C       PERFORMS RELAXATION FORMULA FOR NON-LINEAR PROBLEM
      DO 118 I=1,NPOIN
      TEMPR(I)=RELAX*TEMPR(I)+(1.0-RELAX)*TLAST(I)
  118 CONTINUE
C
C *** CHECK FOR CONVERGENCE OF ITERATION
C
      IF (ITER.EQ.1) THEN
       CALL L2NORM (MJNT,NPOIN,TNRM1,TEMPR)
      ELSE
       CALL L2NORM (MJNT,NPOIN,TNRM2,TEMPR)
       CONVG=ABS(TNRM2-TNRM1)/TNRM2
       IF (CONVG.LE.TOLER) GOTO 150
       TNRM1=TNRM2
      ENDIF
  119 CONTINUE
      WRITE (*,151)
      STOP 3333
  150 CONTINUE
  151 FORMAT (//' ','HEAT FLOW SOLUTION HAS FAILED TO CONVERGE')
      RETURN
      END

      SUBROUTINE PROPTY(IELEM,MELM,MJNT,NNPEL,TEMPR,DEN,DIFS,MTYP,
     -                  NCONC,DENST,CONDT,DIFST,CAPCT,IMATR)
C     ------------------------------------------------------------
C
C     This subroutine calculates temperature-dependent properties
C
C
C     DENST : Element average density value
C     CONDT : Element average conductivity value
C     DIFST : Element average diffusivity value
```

```
C     CAPCT : Element average specific heat value
      IMPLICIT REAL(A-H,O-Z)
      INTEGER NNPEL(MELM),NCONC(MELM,4)
      REAL DEN(MTYP),DIFS(MTYP),TEMPR(MJNT)
C
      TEMPT=0.0
      DO 41 I=1, NNPEL(IELEM)
      TEMPT=TEMPT+TEMPR(NCONC(IELEM,I))/NNPEL(IELEM)
   41 CONTINUE
C
      DENST=DEN(IMATR)
      DIFST=DIFS(IMATR)
      DIFST=9.16391E-7*TEMPT**2-0.00136982*TEMPT+0.909062
      CONDT=1.36469E-6*TEMPT**2-0.00256908*TEMPT+2.24266
      DIFST=DIFST/1.0E6
      CAPCT=CONDT/DENST/DIFST
      RETURN
      END

      SUBROUTINE STIFFN (MELM,IELEM,NNPEL,ITRAN,DENST,CONDT,
     -                   CAPCT,ASTIF,AMASS,XCORD,YCORD)
C     -------------------------------------------------------
C                                                                       STIFFN
C     This subroutine constructs element [K] & [C] matrices             STIFFN
C                                                                       STIFFN
C
      IMPLICIT REAL(A-H,O-Z)
      REAL ASTIF(4,4),AMASS(4,4),XCORD(4),YCORD(4)
      INTEGER NNPEL(MELM)
C
C *** CONSTRUCT STIFFNESS AND MASS MATRICES
C
      IF (NNPEL(IELEM).EQ.4) THEN
C     !RECTANGULAR SIMPLIFIED SCHEME
        CALL SSTIF4(ASTIF,XCORD,YCORD,CONDT)
        IF (ITRAN.EQ.1) THEN
        CALL SMASS4(AMASS,XCORD,YCORD,DENST,CAPCT)
        ENDIF
      ELSEIF (NNPEL(IELEM).EQ.3) THEN
        C1=YCORD(2)-YCORD(3)
        C2=YCORD(3)-YCORD(1)
        C3=YCORD(1)-YCORD(2)
        D1=XCORD(3)-XCORD(2)
        D2=XCORD(1)-XCORD(3)
        D3=XCORD(2)-XCORD(1)
        AREA=0.5*(D2*C1-D1*C2)
        FACTK=CONDT/(4*AREA)
        ASTIF(1,1)=(C1*C1+D1*D1)*FACTK
        ASTIF(1,2)=(C1*C2+D1*D2)*FACTK
        ASTIF(1,3)=(C1*C3+D1*D3)*FACTK
        ASTIF(2,2)=(C2*C2+D2*D2)*FACTK
        ASTIF(2,3)=(C2*C3+D2*D3)*FACTK
        ASTIF(3,3)=(C3*C3+D3*D3)*FACTK
        IF (ITRAN.EQ.1) THEN
        FACTC=DENST*CAPCT*AREA/12.0
        AMASS(1,1)=2*FACTC
        AMASS(1,2)=1*FACTC
        AMASS(1,3)=1*FACTC
        AMASS(2,2)=2*FACTC
        AMASS(2,3)=1*FACTC
        AMASS(3,3)=2*FACTC
        ENDIF
      ENDIF
C     FILL IN THE SYMMETRIC PART OF MATRICES
      DO 51 I=2,NNPEL(IELEM)
      DO 52 J=1,(I-1)
      ASTIF(I,J)=ASTIF(J,I)
      IF (ITRAN.EQ.1) AMASS(I,J)=AMASS(J,I)
   52 CONTINUE
   51 CONTINUE
C
```

```
      RETURN
      END
      SUBROUTINE SSTIF4(ASTIF,XCORD,YCORD,CONDT)
C     ------------------------------------------
C                                                                       SSTIF4
C     This subroutine generates the "analytical" stiffness matrix       SSTIF4
C     for a four node quadrilateral in plane strain based on NIP=4.     SSTIF4
C                                                                       SSTIF4
      IMPLICIT REAL(A-H,O-Z)
      REAL ASTIF(4,4),XCORD(4),YCORD(4)
      REAL X1,X2,X3,X4,Y1,Y2,Y3,Y4,A2,A2ST3,
     _     ALPH,BETA,F1,F2,S1,S2,S3,S4,T1,T2,T3,T4
C
      X1=XCORD(1)
      X2=XCORD(2)
      X3=XCORD(3)
      X4=XCORD(4)
      Y1=YCORD(1)
      Y2=YCORD(2)
      Y3=YCORD(3)
      Y4=YCORD(4)
C
      A2=(X4-X2)*(Y3-Y1)-(X3-X1)*(Y4-Y2)
      A2ST3=3.0*A2*A2
C
      CALL GRUPA(X1,X2,X3,X4,Y1,Y2,Y3,Y4,S1,S2,S3,S4,
     -           T1,T2,T3,T4,F1,F2)
      ALPH=CONDT*(A2*(S1+S2)+F1*(S3+S4))
      BETA=CONDT*(A2*(T1+T2)+F2*(T3+T4))
      ASTIF(1,1)=-(ALPH/(A2ST3-F1**2)+BETA/(A2ST3-F2**2))*0.5
C
      CALL GRUPA(X2,X3,X4,X1,Y2,Y3,Y4,Y1,S1,S2,S3,S4,
     -           T1,T2,T3,T4,F1,F2)
      ALPH=CONDT*(A2*(S1+S2)+F1*(S3+S4))
      BETA=CONDT*(A2*(T1+T2)+F2*(T3+T4))
      ASTIF(2,2)=-(ALPH/(A2ST3-F1**2)+BETA/(A2ST3-F2**2))*0.5
C
      CALL GRUPA(X3,X4,X1,X2,Y3,Y4,Y1,Y2,S1,S2,S3,S4,
     -           T1,T2,T3,T4,F1,F2)
      ALPH=CONDT*(A2*(S1+S2)+F1*(S3+S4))
      BETA=CONDT*(A2*(T1+T2)+F2*(T3+T4))
      ASTIF(3,3)=-(ALPH/(A2ST3-F1**2)+BETA/(A2ST3-F2**2))*0.5
C
      CALL GRUPA(X4,X1,X2,X3,Y4,Y1,Y2,Y3,S1,S2,S3,S4,
     -           T1,T2,T3,T4,F1,F2)
      ALPH=CONDT*(A2*(S1+S2)+F1*(S3+S4))
      BETA=CONDT*(A2*(T1+T2)+F2*(T3+T4))
      ASTIF(4,4)=-(ALPH/(A2ST3-F1**2)+BETA/(A2ST3-F2**2))*0.5
C
      CALL GRUPC(X1,X2,X3,X4,Y1,Y2,Y3,Y4,S1,S2,S3,S4,
     -           T1,T2,T3,T4,F1,F2)
      ALPH=CONDT*(A2*(S1+S2)+F1*(S3+S4))
      BETA=CONDT*(A2*(T1+T2)+F2*(T3+T4))
      ASTIF(1,2)=-(ALPH/(A2ST3-F1**2)+BETA/(A2ST3-F2**2))*0.5
C
      CALL GRUPC(X2,X3,X4,X1,Y2,Y3,Y4,Y1,S1,S2,S3,S4,
     -           T1,T2,T3,T4,F1,F2)
      ALPH=CONDT*(A2*(S1+S2)+F1*(S3+S4))
      BETA=CONDT*(A2*(T1+T2)+F2*(T3+T4))
      ASTIF(2,3)=-(ALPH/(A2ST3-F1**2)+BETA/(A2ST3-F2**2))*0.5
C
      CALL GRUPC(X3,X4,X1,X2,Y3,Y4,Y1,Y2,S1,S2,S3,S4,
     -           T1,T2,T3,T4,F1,F2)
      ALPH=CONDT*(A2*(S1+S2)+F1*(S3+S4))
      BETA=CONDT*(A2*(T1+T2)+F2*(T3+T4))
      ASTIF(3,4)=-(ALPH/(A2ST3-F1**2)+BETA/(A2ST3-F2**2))*0.5
C
      CALL GRUPC(X4,X1,X2,X3,Y4,Y1,Y2,Y3,S1,S2,S3,S4,
     -           T1,T2,T3,T4,F1,F2)
      ALPH=CONDT*(A2*(S1+S2)+F1*(S3+S4))
      BETA=CONDT*(A2*(T1+T2)+F2*(T3+T4))
```

```
      ASTIF(1,4)=-(ALPH/(A2ST3-F1**2)+BETA/(A2ST3-F2**2))*0.5
C
      CALL GRUPE(X1,X2,X3,X4,Y1,Y2,Y3,Y4,S1,S2,S3,S4,
     -           T1,T2,T3,T4,F1,F2)
      ALPH=CONDT*(A2*(S1+S2)+F1*(S3+S4))
      BETA=CONDT*(A2*(T1+T2)+F2*(T3+T4))
      ASTIF(1,3)=-(ALPH/(A2ST3-F1**2)+BETA/(A2ST3-F2**2))*0.5
C
      CALL GRUPE(X2,X3,X4,X1,Y2,Y3,Y4,Y1,S1,S2,S3,S4,
     -           T1,T2,T3,T4,F1,F2)
      ALPH=CONDT*(A2*(S1+S2)+F1*(S3+S4))
      BETA=CONDT*(A2*(T1+T2)+F2*(T3+T4))
      ASTIF(2,4)=-(ALPH/(A2ST3-F1**2)+BETA/(A2ST3-F2**2))*0.5
C
      RETURN
      END
      SUBROUTINE GRUPA(X1,X2,X3,X4,Y1,Y2,Y3,Y4,S1,
     -                 S2,S3,S4,T1,T2,T3,T4,F1,F2)
      IMPLICIT REAL(A-H,O-Z)
      REAL X1,X2,X3,X4,Y1,Y2,Y3,Y4
      REAL S1,S2,S3,S4,T1,T2,T3,T4,F1,F2
      S1=2.0*(Y4-Y2)**2
      S2=2.0*(X4-X2)**2
      S3=-S1/2.0
      S4=-S2/2.0
      T1=(Y2-Y3)**2+(Y3-Y4)**2+(Y4-Y2)**2
      T2=(X2-X3)**2+(X3-X4)**2+(X4-X2)**2
      T3=(Y4-Y3)**2-(Y3-Y2)**2
      T4=(X4-X3)**2-(X3-X2)**2
      CALL F1F2(X1,X2,X3,X4,Y1,Y2,Y3,Y4,F1,F2)
      RETURN
      END
C
      SUBROUTINE GRUPC(X1,X2,X3,X4,Y1,Y2,Y3,Y4,S1,
     -                 S2,S3,S4,T1,T2,T3,T4,F1,F2)
      IMPLICIT REAL(A-H,O-Z)
      REAL X1,X2,X3,X4,Y1,Y2,Y3,Y4
      REAL S1,S2,S3,S4,T1,T2,T3,T4,F1,F2
      S1=(Y4-Y2)*(2.0*Y1-Y3-Y4)
      S2=(X4-X2)*(2.0*X1-X3-X4)
      S3=(Y4-Y2)*(Y4-Y1)
      S4=(X4-X2)*(X4-X1)
      T1=(Y3-Y1)*(2.0*Y2-Y3-Y4)
      T2=(X3-X1)*(2.0*X2-X3-X4)
      T3=(Y3-Y1)*(Y3-Y2)
      T4=(X3-X1)*(X3-X2)
      CALL F1F2(X1,X2,X3,X4,Y1,Y2,Y3,Y4,F1,F2)
      RETURN
      END
C
      SUBROUTINE GRUPE(X1,X2,X3,X4,Y1,Y2,Y3,Y4,S1,
     -                 S2,S3,S4,T1,T2,T3,T4,F1,F2)
      IMPLICIT REAL(A-H,O-Z)
      REAL X1,X2,X3,X4,Y1,Y2,Y3,Y4
      REAL S1,S2,S3,S4,T1,T2,T3,T4,F1,F2
      S1=-(Y4-Y2)**2
      S2=-(X4-X2)**2
      S3=0.0
      S4=0.0
      T1=(Y3+Y1)*(Y4+Y2)-2.0*(Y4-Y2)**2-2.0*(Y1*Y3+Y2*Y4)
      T2=(X3+X1)*(X4+X2)-2.0*(X4-X2)**2-2.0*(X1*X3+X2*X4)
      T3=(Y4-Y2)*(Y1-Y2+Y3-Y4)
      T4=(X4-X2)*(X1-X2+X3-X4)
      CALL F1F2(X1,X2,X3,X4,Y1,Y2,Y3,Y4,F1,F2)
      RETURN
      END
C
      SUBROUTINE F1F2(X1,X2,X3,X4,Y1,Y2,Y3,Y4,F1,F2)
      IMPLICIT REAL(A-H,O-Z)
      REAL X1,X2,X3,X4,Y1,Y2,Y3,Y4
      REAL F1,F2
```

```
      F1=(X1+X3)*(Y4-Y2)-(Y1+Y3)*(X4-X2)-2.0*(X2*Y4-X4*Y2)
      F2=(Y2+Y4)*(X3-X1)-(X2+X4)*(Y3-Y1)-2.0*(X3*Y1-X1*Y3)
      RETURN
      END
C
      SUBROUTINE SMASS4(AMASS,XCORD,YCORD,DENST,CAPCT)
C     ------------------------------------------------
C                                                                       SMASS4
C     This subroutine generates the analytical capacitance matrix       SMASS4
C     for a four node quadrilateral                                     SMASS4
C                                                                       SMASS4
      IMPLICIT REAL(A-H,O-Z)
      REAL AMASS(4,4),XCORD(4),YCORD(4)
      REAL   X1,X2,X3,X4,Y1,Y2,Y3,Y4
      INTEGER I,J
C
      X1=XCORD(1)
      X2=XCORD(2)
      X3=XCORD(3)
      X4=XCORD(4)
      Y1=YCORD(1)
      Y2=YCORD(2)
      Y3=YCORD(3)
      Y4=YCORD(4)
C
      AMASS(1,1)=X1*(Y2-Y4)/12        + X2*(Y3-3*Y1+2*Y4)/36
     -          +X3*(Y4-Y2)/36        + X4*(3*Y1-2*Y2-Y3)/36
      AMASS(1,2)=X1*(3*Y2-Y3-2*Y4)/72 + X2*(2*Y3-3*Y1+Y4)/72
     -          +X3*(Y1-2*Y2+Y4)/72   + X4*(2*Y1-Y2-Y3)/72
      AMASS(1,3)=X1*(Y2-Y4)/72        + X2*(Y3-Y1)/72
     -          +X3*(Y4-Y2)/72        + X4*(Y1-Y3)/72
      AMASS(1,4)=X1*(2*Y2+Y3-3*Y4)/72 + X2*(Y3-2*Y1+Y4)/72
     -          +X3*(2*Y4-Y1-Y2)/72   + X4*(3*Y1-Y2-2*Y3)/72
      AMASS(2,2)=X1*(3*Y2-2*Y3-Y4)/36 + X2*(Y3-Y1)/12
     -          +X3*(2*Y1+Y4-3*Y2)/36 + X4*(Y1-Y3)/36
      AMASS(2,3)=X1*(2*Y2-Y3-Y4)/72   + X2*(3*Y3-2*Y1-Y4)/72
     -          +X3*(2*Y4+Y1-3*Y2)/72 + X4*(Y1+Y2-2*Y3)/72
      AMASS(2,4)=X1*(Y2-Y4)/72        + X2*(Y3-Y1)/72
     -          +X3*(Y4-Y2)/72        + X4*(Y1-Y3)/72
      AMASS(3,3)=X1*(Y2-Y4)/36        + X2*(3*Y3-Y1-2*Y4)/36
     -          +X3*(Y4-Y2)/12        + X4*(Y1+2*Y2-3*Y3)/36
      AMASS(3,4)=X1*(Y2+Y3-2*Y4)/72   + X2*(2*Y3-Y1-Y4)/72
     -          +X3*(3*Y4-Y1-2*Y2)/72 + X4*(2*Y1+Y2-3*Y3)/72
      AMASS(4,4)=X1*(Y2+2*Y3-3*Y4)/36 + X2*(Y3-Y1)/36
     -          +X3*(3*Y4-2*Y1-Y2)/36 + X4*(Y1-Y3)/12
      DO I=1,4
      DO J=I,4
      AMASS(I,J)=DENST*CAPCT*AMASS(I,J)
      END DO
      END DO
      RETURN
C
      END

      SUBROUTINE PROFAC (MPROF,MJNT,N,A,ND)
C     -------------------------------------
C                                                                       PROFAC
C     This subroutine factorizes the global stiff matrix                PROFAC
C                                                                       PROFAC
C
      IMPLICIT REAL(A-H,O-Z)
      REAL A(MPROF)
      INTEGER N,ND(MJNT)
      IF (A(1).GT.0.0) GOTO 78
   79 WRITE(*,*) 'GSTIF IS NOT POSITIVE DEFINITE!'
      STOP 2222
   78 CONTINUE
      DO 77 J=2,N
       J1=J-1
       NJ=ND(J1)
       JJ=ND(J)
```

```
       NCJ=JJ-NJ
       IF (NCJ.EQ.1) GOTO 77
       IF (J1.EQ.1) GOTO 75
       DO 74 I=2,J1
        JMI=J-I
        IF (NCJ.LE.(JMI+1)) GOTO 74
        I1=I-1
        NI=ND(I1)
        NJ=ND(J1)
        II=ND(I)
        IJ=JJ-JMI
        NCI=II-NI
        NCJI=NCJ-JMI
        NCD=NCJI-NCI
        IF (NCD.LT.0) GOTO 71
        K1=NCI-1
        NJ=NJ+NCD
        GOTO 72
   71   K1=NCJI-1
        NI=NI-NCD
   72   SUM=A(IJ)
        IF (K1.EQ.0) GOTO 74
        DO 73 K=1,K1
         KI=NI+K
         KJ=NJ+K
         SUM=SUM-A(KI)*A(KJ)
   73   CONTINUE
        A(IJ)=SUM
   74  CONTINUE
   75  SUM=A(JJ)
       NCJ1=NCJ-1
       DO 76 K=1,NCJ1
        KJ=ND(J1)+K
        KK=ND(J-NCJ+K)
        TEMP=A(KJ)/A(KK)
        SUM=SUM-TEMP*A(KJ)
        A(KJ)=TEMP
   76  CONTINUE
       IF (SUM.LE.0.0) GOTO 79
       A(JJ)=SUM
   77 CONTINUE
      RETURN
      END

      SUBROUTINE PROSOL (MPROF,MJNT,N,U,B,X,ND)
C     ------------------------------------------
C                                                                       PROSOL
C     This subroutine solves the final system                           PROSOL
C                                                                       PROSOL
C
      IMPLICIT REAL(A-H,O-Z)
      REAL U(MPROF),B(MJNT),X(MJNT)
      INTEGER N,ND(MJNT)
      DO 82 I=1,N
      SUM=B(I)
      IF (I.EQ.1) GOTO 86
      I1=I-1
      NI=ND(I1)
      II=ND(I)
      NCI=II-NI
      K1=NCI-1
      KR=I-NCI
      IF (K1.EQ.0) GOTO 86
      DO 81 K=1,K1
      KI=NI+K
      KR=KR+1
      SUM=SUM-U(KI)*X(KR)
   81 CONTINUE
   86 X(I)=SUM
   82 CONTINUE
      DO 83 I=1,N
```

```
      II=ND(I)
      X(I)=X(I)/U(II)
   83 CONTINUE
      DO 85 I1=1,N
      I=N-I1+1
      K2=I+1
      SUM=X(I)
      IF (I.EQ.N) GOTO 87
      DO 84 K=K2,N
      K1=K-1
      KMI=K-I
      NK=ND(K1)
      NCK=ND(K)-NK
      NCKI=NCK-KMI
      IF (NCKI.LE.0) GOTO 84
      IK=NK+NCKI
      SUM=SUM-U(IK)*X(K)
   84 CONTINUE
   87 X(I)=SUM
   85 CONTINUE
      RETURN
      END

      SUBROUTINE L2NORM (MJNT,NPOIN,TNORM,TEMPR)
C     ------------------------------------------
C                                                                       L2NORM
C     This subroutine calculates the L2 norm for a given vector         L2NORM
C                                                                       L2NORM
C
      IMPLICIT REAL(A-H,O-Z)
      REAL TEMPR(MJNT)
      TMAXM=0.0
      TSUM2=0.0
      DO 98 I=1,NPOIN
      IF (TMAXM.LT.ABS(TEMPR(I))) TMAXM=ABS(TEMPR(I))
      TSUM2=TSUM2+TEMPR(I)**2.0
   98 CONTINUE
      IF (TMAXM.EQ.0.0) TMAXM=1.0
      TNORM=SQRT((TSUM2/TMAXM**2.0)/NPOIN)
      RETURN
      END

      SUBROUTINE TRANSI (MELM,MJNT,MPROF,TOLER,ITRAN,NITER,
     -                   NELEM,NPOIN,NNPEL,NPROF,NDFEL,NCONC,
     -                   NDIAG,IFFIX,TFIXD,ASTIF,AMASS,GSTIF,
     -                   EFORC,FORCE,TLAST,TEMPR,COORD,DTIME,
     -                   ALPHA,TEMP1,RVECT,MTYP,MTYPE,DEN,DIFS)
C     ------------------------------------------------------------
C                                                                       TRANSI
C     This subroutine performs transient analysis                       TRANSI
C                                                                       TRANSI
      IMPLICIT REAL(A-H,O-Z)
      REAL ASTIF(4,4),GSTIF(MPROF),AMASS(4,4)
      REAL EFORC(4),XCORD(4),YCORD(4),FORCE(MJNT),TLAST(MJNT)
      REAL TEMPR(MJNT),COORD(MJNT,2),TFIXD(MJNT),TEMP1(4,4),RVECT(4)
      REAL DEN(MTYP),DIFS(MTYP),TOLER,DTIME,ALPHA
      INTEGER ITRAN,NITER,NELEM,NPOIN,NNPEL(MELM),NPROF,MTYPE(MELM)
      INTEGER NDFEL(4),NCONC(MELM,4),NDIAG(MJNT),IFFIX(MJNT)
C
C ***  PERFORM  TRANSIENT  ANALYSIS
C
      DO 135 I=1,NPOIN
      TLAST(I)=TEMPR(I)
  135 CONTINUE
      DO 136 ITER=1,NITER
         DO 137 I=1,NPOIN
         FORCE(I)=0.0
  137    CONTINUE
         DO 138 I=1,NPROF
         GSTIF(I)=0.0
  138    CONTINUE
```

```
         DO 139 IELEM=1,NELEM
           IMATR=MTYPE(IELEM)
           CALL PROPTY(IELEM,MELM,MJNT,NNPEL,TEMPR,DEN,DIFS,MTYP,
     -               NCONC,DENST,CONDT,DIFST,CAPCT,IMATR)
            DO 140 I=1,NNPEL(IELEM)
               NDFEL(I)=NCONC(IELEM,I)
               ID=NDFEL(I)
               EFORC(I)=0.0
               XCORD(I)=COORD(ID,1)
               YCORD(I)=COORD(ID,2)
               DO 141 J=1,NNPEL(IELEM)
                  AMASS(I,J)=0.0
                  ASTIF(I,J)=0.0
  141          CONTINUE
  140       CONTINUE
C
C *** CONSTRUCT ELEMENT STIFFNESS AND MASS MATRICES
C
      CALL STIFFN (MELM,IELEM,NNPEL,ITRAN,DENST,CONDT,
     -             CAPCT,ASTIF,AMASS,XCORD,YCORD)
C
C *** CALCULATE NEUMANN BOUNDARY CONDITION EFFECTS HERE
C *** ADD (ELEMENT'S) INTERNAL VOLUMETRIC HEAT RESOURCE TO THE FORCE VECTOR
C *** TRANSFER ELEMENT FORCES TO GLOBAL FORCE VECTOR
C
      DO 142 J=1,NNPEL(IELEM)
      NOD=NDFEL(J)
      FORCE(NOD)=FORCE(NOD)+EFORC(J)
  142 CONTINUE
C
C *** CALCULATE ELEMENT CONTRIBUTION TO FORCE VECTOR (RHS)
C
      DO 116 I=1,NNPEL(IELEM)
      RVECT(I)=0.0
      DO 111 J=1,NNPEL(IELEM)
      TEMP1(I,J)=AMASS(I,J)/DTIME-ASTIF(I,J)*(1.0-ALPHA)
  111 CONTINUE
  116 CONTINUE
      DO 114 I=1,NNPEL(IELEM)
      DO 115 J=1,NNPEL(IELEM)
      RVECT(I)=RVECT(I)+TEMP1(I,J)*TLAST(NDFEL(J))
  115 CONTINUE
  114 CONTINUE
C
C *** ASSEMBLE ELEMENT RHS VECTOR INTO GLOBAL FORCE VECTOR
C
      DO 143 I=1,NNPEL(IELEM)
      NOD=NDFEL(I)
      FORCE(NOD)=FORCE(NOD)+RVECT(I)
  143 CONTINUE
C
C *** CALCULATE ELEMENT CONTRIBUTION TO SYSTEM MATRIX (LHS)
C
      DO 91 I=1,NNPEL(IELEM)
      DO 105 J=1,NNPEL(IELEM)
      TEMP1(I,J)=AMASS(I,J)/DTIME+ASTIF(I,J)*ALPHA
  105 CONTINUE
   91 CONTINUE
C
C *** ASSEMBLE INTO GLOBAL MATRIX IN VECTOR FORM
C
      DO 144 J=1,NNPEL(IELEM)
      DO 145 K=J,NNPEL(IELEM)
      IROW=NDFEL(J)
      IF (IFFIX(IROW).EQ.1) GOTO 109
      ICOL=NDFEL(K)
      IF (IFFIX(ICOL).EQ.1) GOTO 109
      IF (IROW.LE.ICOL) GOTO 110
      ITEM=IROW
      IROW=ICOL
      ICOL=ITEM
```

```
  110 IRC=NDIAG(ICOL)-ICOL+IROW
      GSTIF(IRC)=GSTIF(IRC)+TEMP1(J,K)
  109 CONTINUE
  145 CONTINUE
  144 CONTINUE
C
C *** MODIFY LOADS TO ACCOUNT FOR FIXED TEMPERATURE NODES
C
      DO 146 J=1,NNPEL(IELEM)
      IROW=NDFEL(J)
      IF (IFFIX(IROW).EQ.1) GOTO 108
      DO 147 K=1,NNPEL(IELEM)
      ICOL=NDFEL(K)
      IF (IFFIX(ICOL).EQ.1) THEN
      FORCE(IROW)=FORCE(IROW)-TEMP1(J,K)*TFIXD(ICOL)
      ENDIF
  147 CONTINUE
  108 CONTINUE
  146 CONTINUE
  139 CONTINUE
C
C *** SOLVE THE FINAL SYSTEM USING PROFILE SOLVER
C
C 1.  SET DIAGONAL ELEMENTS OF GSTIF CORRESPONDING TO FIXED TEMPERATURE
C     NODES EQUAL TO UNITY AND THE FORCE VECTOR TO THE FIXED VALUE
C
      DO 148 I=1,NPOIN
      IF (IFFIX(I).EQ.1) THEN
      J=NDIAG(I)
      GSTIF(J)=1.0
      FORCE(I)=TFIXD(I)
      TEMPR(I)=TFIXD(I)
      ENDIF
  148 CONTINUE
C
C 2.  SOLVE THE FINAL SYSTEM USING PROFILE SOLVER
C
      CALL PROFAC (MPROF,MJNT,NPOIN,GSTIF,NDIAG)
      CALL PROSOL (MPROF,MJNT,NPOIN,GSTIF,FORCE,TEMPR,NDIAG)
C
C *** CHECK FOR CONVERGENCE OF ITERATION
C
      IF (ITER.EQ.1) THEN
       CALL L2NORM (MJNT,NPOIN,TNRM1,TEMPR)
      ELSE
       CALL L2NORM (MJNT,NPOIN,TNRM2,TEMPR)
       CONVG=ABS(TNRM2-TNRM1)/TNRM2
       IF (CONVG.LE.TOLER) GOTO 250
       TNRM1=TNRM2
      ENDIF
  136 CONTINUE
      WRITE (*,251)
      STOP 3333
  250 CONTINUE
  251 FORMAT (//' ','HEAT FLOW SOLUTION HAS FAILED TO CONVERGE')
      RETURN
      END
```

Appendix D V2TRED Source Code

```
      SUBROUTINE TRED(ME,TEMPL,TREF,NAGG)
C     **********************************
C
C     This subroutine calculates temperature-related reduction
C     coefficients for various concrete and steel material properties.
C
      PARAMETER (MELM=5000)

      COMMON /REDUCTION/ TRC,TRS

      REAL TRC(MELM,4),TRS(MELM,4)
      REAL TEMPL,TREF,TA
      REAL KC1,KC2,KC3,KC4
      REAL KS1,KS2,KS3,KS4
      INTEGER ME,NAGG
      REAL TIER(13),VALUEC11(13),VALUEC12(13),VALUEC3(13),
     -     VALUES1(13),VALUES2(13),VALUES3(13)
      REAL INTPLAT,BASE,BASE1,BASE2
C
C     Note: NAGG=1 for carbonate aggregates
C           NAGG=2 for silicious aggregates
C
      TA=TEMPL+TREF
      DATA TIER/20,100,200,300,400,500,600,
     -          700,800,900,1000,1100,1200/
C
C     Concrete Compressive Strength
C
      DATA VALUEC11/1.0,1.0,0.97,0.91,0.85,0.74,0.60,
     -            0.43,0.27,0.15,0.06,0.02,0.0/
      DATA VALUEC12/1.0,1.0,0.95,0.85,0.75,0.60,0.45,
     -            0.3,0.15,0.08,0.04,0.01,0.0/
      IF (NAGG.EQ.1) THEN
        KC1=INTPLAT(TA,TIER,VALUEC11)
      ELSE
        KC1=INTPLAT(TA,TIER,VALUEC12)
      ENDIF
      IF (KC1.LT.1.E-6) KC1=1.E-6
C
C     Concrete Tensile Strength
C
      IF (TA.LE.100) KC2=1.0
      IF (TA.GT.100) KC2=1.0-(TA-100)/500.
C     NO TENSILE STRENGTH WHEN TEMP. ABOVE 500
      IF (KC2.LT.1.E-6) KC2=1.E-6
C
C     Concrete's Strain(ec') at which Stress reaches fc'
C
      DATA VALUEC3/.0025,.0040,.0055,.0070,.010,.015,.025,
     -            .025,.025,.025,.025,.025,1.E6/
      IF (NAGG.EQ.1) THEN
        IF (TA.LE.20) THEN
         BASE=0.0
        ELSEIF (TA.LE.805) THEN
         BASE=-1.2E-4+6.E-6*TA+1.4E-11*TA*TA*TA
        ELSE
         BASE=12E-3
        END IF
        BASE=INTPLAT(TA,TIER,VALUEC3)-BASE
        KC3=BASE/VALUEC3(1)
C        IF THERMAL STAIN NOT TO BE DEDUCTED THEN:
```

```
C        KC3=INTPLAT(TA,TIER,VALUEC3)/VALUEC3(1)
      ELSE
        IF (TA.LE.20) THEN
         BASE=0.0
        ELSEIF (TA.LE.700) THEN
         BASE=-1.8E-4+9.E-6*TA+2.3E-11*TA*TA*TA
        ELSE
         BASE=14E-3
        END IF
        BASE=INTPLAT(TA,TIER,VALUEC3)-BASE
        KC3=BASE/VALUEC3(1)
C        IF THERMAL STAIN NOT TO BE DEDUCTED THEN:
C        KC3=INTPLAT(TA,TIER,VALUEC3)/VALUEC3(1)
      ENDIF
      IF (KC3.LT.1.E-6) KC3=1.E-6
C
C     Concrete Coefficient of Thermal Expansion
C     NOTE: FACTORS GIVEN HERE MAKE WORKS THE FORMULA
C           e(T)=Delta_T(T)*Alpha(T)
C               =(T-20)*Alpha(20)*KS4,
C           WHICH ORIGINALLY SUITS CONSTANT Alpha ONLY
c
C     BASE1:Alpha at 20'C for carbonate Aggregates
C     BASE2:Alpha at 20'C for Silicious Aggregates
C     BASE:Thermal Strain at TA'C
C
      BASE1=6.E-6+4.2E-11*20*20
      BASE2=9.E-6+6.9E-11*20*20
      IF (NAGG.EQ.1) THEN
       IF (TA.LE.20) THEN
        KC4=1.0
       ELSEIF (TA.LE.805) THEN
        BASE=-1.2E-4+6.E-6*TA+1.4E-11*TA*TA*TA
        KC4=BASE/(TA-20)/BASE1
       ELSE
        BASE=12E-3
        KC4=BASE/(TA-20)/BASE1
       END IF
      ELSE
       IF (TA.LE.20) THEN
        KC4=1.0
       ELSEIF (TA.LE.700) THEN
        BASE=-1.8E-4+9.E-6*TA+2.3E-11*TA*TA*TA
        KC4=BASE/(TA-20)/BASE2
       ELSE
        BASE=14E-3
        KC4=BASE/(TA-20)/BASE2
       END IF
      ENDIF
C
C     Reinforcement Yield Stress
C
      DATA VALUES1/1.0,1.0,1.0,1.0,1.0,0.78,0.47,
     -            0.23,0.11,0.06,0.04,0.02,0.0/
      KS1=INTPLAT(TA,TIER,VALUES1)
      IF (KS1.LT.1.E-6) KS1=1.E-6
C
C     Reinforcement Ultimate Strength
C
      DATA VALUES2/1.0,1.0,0.81,0.61,0.42,0.36,0.18,
     -            0.07,0.05,0.04,0.02,0.01,0.0/
      KS2=INTPLAT(TA,TIER,VALUES2)
      IF (KS2.LT.1.E-6) KS2=1.E-6
C
C     Reinforcement Modulus of Elasticity
C
      DATA VALUES3/1.0,1.0,0.90,0.80,0.70,0.60,0.31,
     -            0.13,0.09,0.07,0.04,0.02,0.0/
      KS3=INTPLAT(TA,TIER,VALUES3)
      IF (KS3.LT.1.E-6) KS3=1.E-6
C
```

```
C     Reinforcement Coefficient of Thermal Expansion (Alpha)
C     NOTE: FACTORS GIVEN HERE MAKE WORKS THE FORMULA
C           e(T)=Delta_T(T)*Alpha(T)
C               =(T-20)*Alpha(20)*KS4
C           WHICH ORIGINALLY SUITS CONSTANT Alpha ONLY
C     BASE1:Alpha at 20'C
C     BASE:Thermal Strain at TA'C
C
      BASE1=1.2E-5+0.8E-8*20
      IF (TA.LE.20) THEN
        KS4=1.0
      ELSEIF (TA.LE.750) THEN
        BASE=-2.146E-4+1.2E-5*TA+0.4E-8*TA*TA
        KS4=BASE/(TA-20)/BASE1
      ELSEIF (TA.LE.860) THEN
        BASE=11.E-3
        KS4=BASE/(TA-20)/BASE1
      ELSE
        BASE=-6.2E-3+2.E-5*TA
        KS4=BASE/(TA-20)/BASE1
      END IF
C
      TRC(ME,1)=KC1
      TRC(ME,2)=KC2
      TRC(ME,3)=KC3
      TRC(ME,4)=KC4
      TRS(ME,1)=KS1
      TRS(ME,2)=KS2
      TRS(ME,3)=KS3
      TRS(ME,4)=KS4
C
      RETURN
      END

C     THIS FUNCTION WORKS EVEN WHEN TIER'S NO LESS THAN 13
      REAL FUNCTION INTPLAT(TEM,TIER,VALUE)
      REAL TEM,TIER(13),VALUE(13)
      INTEGER INDEX
      REAL DIF
      DO 1 INDEX=1,13
        IF (TEM.LE.TIER(1)) THEN
           INTPLAT=VALUE(1)
           RETURN
        END IF
        IF (TEM.LE.TIER(INDEX)) THEN
         DIF=(VALUE(INDEX-1)-VALUE(INDEX))/(TIER(INDEX)-TIER(INDEX-1))
         INTPLAT=VALUE(INDEX-1)-DIF*(TEM-TIER(INDEX-1))
         RETURN
        END IF
        IF (TEM.GE.TIER(13)) THEN
           INTPLAT=VALUE(13)
           RETURN
        END IF
  1   CONTINUE
      END
```