PIT Project Research Report: SS1

Development of Generalised Stress Strain Relationships for the Concrete Slab in Shell Models

Martin Gillie

December 1999

The University of Edinburgh
School of Civil Engineering
Crew Building
King's Buildings
West Mains Road
University of Edinburgh
EH9 3JN

Contents

1 Introduction

The numerical modelling of reinforced concrete is challenging because the stress-strain relationship is highly non-linear and different in tension and compression. The behaviour is particularly complex when trying to model heated concrete slabs with a high degree of edge restraint. This report describes the approach used for modelling the concrete floor slabs in the Cardington tests with shell finite elements. The modelling was performed using the FEAST (**F**inite **E**lement **A**nalysis of **S**hells at High **T**emperatures) suite of programs. FEAST was developed specifically for the analysis of complex plates such as the Cardington floor slab. It was also designed to interact smoothly with the commercial finite element package, ABAQUS.

The first section of the report provides a general description of the approaches available for modelling plate structures at high temperatures and a description of the software developed during the course of this research to model such structures. The second section contains a detailed description of how FEAST was used to model the Cardington slab. A description of the assumptions made during the modelling is given.

2 Modelling of Plates

2.1 Depth Integration Approach to Modelling Plates

The most usual approach to modelling any sort of plate structure using finite element techniques is to use what can be described as the depth integration technique. This method calculates the stresses through the depth of an element by consideration of the geometry of the element, its strain state and its material properties. The accuracy of a calculation using this technique can be increased by increasing the number of points through the depth of the plate at which the stresses are calculated. The integration of the stresses to obtain forces is normally done using Simpson's rule or Gauss quadrature.[6]

After a number of attempts to model the Cardington slab using the concrete model built in to ABAQUS and the depth integration approach it was found that numerical problems were too severe for a usable model to be developed. Others who have attempted this approach to modelling complex plates, particularly at elevated temperatures, have had similar problems. This has usually meant that results have either only been available for a limited range of plate behaviour, or that the models used have been very simplistic in terms of material behaviour.

2.2 Stress-Resultant Approach to Modelling Plates

An alternative method of plate modelling is to use a stress-resultant approach. This essentially involves the combination of the material behaviour and geometry of a plate into one set of equations. The forces and moments per unit width of plate are calculated based on the strain, curvature and temperature of the plate reference surface. Temperature gradients can also be taken account of. The drawback of this method is that it does not allow stresses to be output from the analysis; only stress-resultants. This means that it is not possible to determine the variation of stresses through the depth of the plate. Despite this restriction, the method is very powerful in that it allows the analyst to define a plate with arbitrary behaviour without the need to resort to complex multi-axial material models with their associated numerical convergance problems.

For a linear-elastic unheated orthotropic plate the relationship between stress-resultants and generalised strains can be represented by:

$$\left[\begin{array}{c} P_{11} \\ P_{22} \\ P_{12} \\ \hline M_{11} \\ M_{22} \\ M_{12} \end{array}\right] = \left[\begin{array}{ccc|ccc} K_{11} & K_{12} & K_{13} & K_{14} & K_{15} & K_{16} \\ K_{21} & K_{22} & K_{23} & K_{24} & K_{25} & K_{26} \\ K_{31} & K_{32} & K_{33} & K_{34} & K_{35} & K_{36} \\ \hline K_{41} & K_{42} & K_{43} & K_{44} & K_{45} & K_{46} \\ K_{51} & K_{52} & K_{53} & K_{54} & K_{55} & K_{56} \\ K_{61} & K_{62} & K_{63} & K_{64} & K_{65} & K_{66} \end{array}\right] \left[\begin{array}{c} \epsilon_{11} \\ \epsilon_{22} \\ \epsilon_{12} \\ \hline \phi_{11} \\ \phi_{22} \\ \phi_{12} \end{array}\right] \tag{1}$$

The vector on the left of Eqn. 1 is a set of stress resultants for the plate that relate to two membrane forces, an in-plane shear force, two bending moments and a twisting moment. The vector on the right contains the corresponding generalised strains. The matrix contains a set of stiffness terms.

The degree to which the matrix is populated increases as more complex plates are modelled. For the simplest cases it is sufficient to populate only the leading diagonal of the matrix, and possibly some of diagonal terms to allow for Poisson effects. For all but these cases however, it is necessary to at least partially populate the off-diagonal quadrants of the matrix. This introduces coupling between membrane and bending forces and allows formulations for more complex plates to be produced. If coupling is introduced it is necessary to select a plate reference surface from which all quantities can be measured. Typically this might coincide with the top or bottom surface of the plate, or pass through the centroid of the cross section. Eqn. 1 can, if desired, be further generalised by the addition of two transverse (vertical) shear terms.

For a plate with non-linear material behaviour the terms in the stiffness matrix will vary with the strain vector. For such cases en. 1 must be modified to read:

$$\left[\begin{array}{c} \delta P_{11} \\ \delta P_{22} \\ \delta P_{12} \\ \hline \delta M_{11} \\ \delta M_{22} \\ \delta M_{12} \end{array}\right] = \left[\begin{array}{ccc|ccc} K_{11} & K_{12} & K_{13} & K_{14} & K_{15} & K_{16} \\ K_{21} & K_{22} & K_{23} & K_{24} & K_{25} & K_{26} \\ K_{31} & K_{32} & K_{33} & K_{34} & K_{35} & K_{36} \\ \hline K_{41} & K_{42} & K_{43} & K_{44} & K_{45} & K_{46} \\ K_{51} & K_{52} & K_{53} & K_{54} & K_{55} & K_{56} \\ K_{61} & K_{62} & K_{63} & K_{64} & K_{65} & K_{66} \end{array}\right] \left[\begin{array}{c} \delta \epsilon_{11} \\ \delta \epsilon_{22} \\ \delta \epsilon_{12} \\ \hline \delta \phi_{11} \\ \delta \phi_{22} \\ \delta \phi_{12} \end{array}\right] \tag{2}$$

Now the stiffness matrix contain instantaneous tangential values of stiffness, and both vectors represent a small change from a previous state. It is possible to generalise this equation further by taking into account the differences between total, mechanical and thermal strains. Typically in numerical work the total strains and the temperature state in plates are known. As it is the mechanical strains that give rise to stresses, and hence forces and moments, it is these that must be used in Eqn. 2. This gives rise to the following equation:

$$\left[\begin{array}{c} \delta P_{11} \\ \delta P_{22} \\ \delta P_{12} \\ \hline \delta M_{11} \\ \delta M_{22} \\ \delta M_{12} \end{array}\right] = \left[\begin{array}{ccc|ccc} K_{11} & K_{12} & K_{13} & K_{14} & K_{15} & K_{16} \\ K_{21} & K_{22} & K_{23} & K_{24} & K_{25} & K_{26} \\ K_{31} & K_{32} & K_{33} & K_{34} & K_{35} & K_{36} \\ \hline K_{41} & K_{42} & K_{43} & K_{44} & K_{45} & K_{46} \\ K_{51} & K_{52} & K_{53} & K_{54} & K_{55} & K_{56} \\ K_{61} & K_{62} & K_{63} & K_{64} & K_{65} & K_{66} \end{array}\right] \left[\begin{array}{c} \delta \epsilon_{t11} - \delta \epsilon_{th11} \\ \delta \epsilon_{t22} - \delta \epsilon_{th22} \\ \delta \epsilon_{t12} - \delta \epsilon_{th12} \\ \hline \delta \phi_{t11} - \delta \phi_{th11} \\ \delta \phi_{t22} - \delta \phi_{th22} \\ \delta \phi_{t12} - \delta \phi_{th12} \end{array}\right] \tag{3}$$

where: $\delta \epsilon_{tij}$ = an increment of total strain
$\delta \phi_{tij}$ = an increment of total curvature
$\delta \epsilon_{thij}$ = an increment of thermal strain
$\delta \phi_{thij}$ = an increment of thermal curvature

The values of thermal strains and curvatures in Eqn. 3 can easily be determined provided that the coefficient of thermal expansion for the plate is known. Thermal strain is given by:

$$\alpha_{(T)} = \frac{\epsilon_{th(T)}}{\Delta T} \tag{4}$$

where: $\epsilon_{th(T)}$ = the strain resulting from a change in temperature
ΔT = change in temperature
$\alpha_{(T)}$ = coefficient of thermal expansion
T = temperature

If it is assumed that the temperature gradient through the slab is linear then the thermal curvatures can be calculated from:

$$\phi = \alpha G \tag{5}$$

where: G = temperature gradient through slab

Eqn. 3 can be used incrementally during a finite element analysis to provide solutions to the behaviour of arbitrary plates under complex loading conditions.

2.3 The FEAST Suite of Programs

The FEAST suite was developed to provide a set of programs that enabled the modelling of arbitrary plates at high temperatures using the stress-resultant approach to be undertaken. It consists of two main programs:

- SRAS - **S**tress-**R**esultant **A**nalysis of **S**hells. This program analyses user defined plates over given range of stress-strain-temperature states
- FEAI - **F**inite **E**lement **A**nalysis **I**nterface. This program allows stress-resultant based calculations for plates to be undertaken with ABAQUS.

Although the FEAST suite was developed mainly for modelling the Cardington slab, care was taken to ensure that the programs could be easily adapted to model any plate. No limitations were placed on the number of materials that could be included in a plate section or on the geometry of the section. All input parameters were defined externally in separate files so that changing between plate sections was straightforward. By ensuring this generality it was hoped that FEAST could be used in other problems where complex plates needed to be modelled. To be consistent with this approach FEAST is described in this section in general terms and then its specific application to the Cardington slab is described in the following section.

2.4 SRAS

The aim of SRAS is to allow the analyst to determine the behaviour of arbitrary plates over a wide range of strain-curvature-temperature states and to provide output that can easily be manipulated for graphical representation. To calculate a plate's behaviour SRAS reads an input file that divides the plate's cross section into a number of layers as shown in Fig. 3. Each layer is defined by its area, the distance of its centroid from the plate reference surface and its material. Stress-strain-temperature data-files are read for each material in the cross-section. The input file contains user specified information regarding the range of reference surface strain, curvature and temperature values over which the stress-resultants are to be calculated and also the intervals between these values.

Based on the assumption that initially plane sections remain plane after bending the strain in a given layer is calculated according the equation:

$$\epsilon_l = \epsilon_r + z_l \phi \tag{6}$$

where: ϵ_l = average strain in the layer

ϵ_r = reference surface strain

z_l = distance of the centre a layer from the reference surface

ϕ = curvature of the reference surface

The temperature in each layer is a function of the reference surface temperature and the thermal gradient through the plate. The gradient in a plate may be approximately linear but, in many fire situations, is not. SRAS allows for non-linear gradients but requires that they can be approximated as a polynomial that relates the distance from the reference surface (x-variable) to temperature (y-value). Different polynomials may be specified at each of the reference surface temperature intervals specified in the input file. It is also possible to divide the layers of the plate into groups and assign different sets of polynomial temperature relationships to each group. This capability is useful when dealing with plates of varying thickness as shown in Fig. 3.

Once the strain and temperature in each layer have been established the stress can be determined from the stress-strain-temperature relationship of the layer's material. Given the stress in a layer it is possible to establish the force by multiplying the stress by the layer's area. The force in a plate section is the integral of the stresses over the area of the section:

$$F = \int_A \sigma dA \tag{7}$$

which in the SRAS layered approach reduces to:

$$F = \sum_{i=1}^{n} A_i \sigma_i \tag{8}$$

where: F = the force acting on the cross section
A = the area of the cross section
A_i = the area of layer i
n = the number of layers

Similarly the moment about the reference axis is calculated from the expression:

$$M_r = \int_A \sigma z dA \tag{9}$$

which in the SRAS layered approach reduces to:

$$M_r = \sum_{i}^{n} F_i z_i \tag{10}$$

where: M_r = moment about the reference surface
F_i = force in layer i

The moments and forces produced by this procedure can be converted to stress-resultants by dividing by the width of plate section that was used in the analysis.

A flow diagram describing the behaviour of SRAS is shown in Fig. 1 and the details of how the stress in each layer is calculated are shown in Fig. 2.

SRAS can be made to output data in a number of forms. Most simply it can produce tabulated data for plotting the force-strain and moment-curvature behaviour of plates at various sections but it can also produce output that can be plotted as force-moment interaction diagrams for plates.

2.5 FEAI

FEAI takes advantage of the fact that ABAQUS can be programmed to model plates using a stress-resultant approach with the addition of a user- defined subroutine. At the start of each load increment during an analysis, ABAQUS calls FEAI and passes to it information about the current state of the plate section. This information includes the current temperature of the plate reference surface, T; a vector, $\mathbf{N_0}$, containing section stress-resultants; a vector, $\mathbf{E}$, containing section total strains and a vector, $\Delta\mathbf{E}$, containing section total strain increments for the current iteration of the load increment. ABAQUS also allows the user to pass information from the problem's ABAQUS input file directly to FEAI. In FEAI this information includes the thermal expansion coefficient of the plate, the final thermal gradient through the plate and its Poisson ratio. FEAI must perform two functions with this data: it must update the stress-resultants to their values at the end of the load increment and it must provide a section stiffness matrix, $\partial \mathbf{N_0}/\partial \mathbf{E}$. The process can be represented algebraically as:

$$\mathbf{N_u} = \mathbf{N_0} + \frac{\partial \mathbf{N_0}}{\partial \mathbf{E}}(\Delta\mathbf{E} - \Delta\mathbf{E_T}) \tag{11}$$

where: $\mathbf{N_u}$ = the updated stress-resultant vector.
$\mathbf{E_T}$ = a vector containing section thermal strain and curvature increments

The quantity $\partial \mathbf{N_0}/\partial \mathbf{E}$ represents the matrix of tangent stiffness terms in Eqn. 3. To calculate these stiffnesses it is necessary to take the vector $\mathbf{N_0}$ and then calculate two other vectors $\mathbf{N}_{\delta\epsilon}$ and $\mathbf{N}_{\delta\phi}$. These two vectors represent the stress resultants in the slab after infinitesimal increases in the strain, $\delta\epsilon$, and curvature, $\delta\phi$, respectively. The vectors $\mathbf{N}_{\delta\epsilon}$ and $\mathbf{N}_{\delta\phi}$ are computed in a very similar manner to the way in which stress-resultants are calculated in SRAS. The only variation is that in FEAI the strains and curvatures are based on values obtained from the ABAQUS analysis rather than being specified by the user. By taking components from all three stress-resultant vectors it is possible to determine the terms of $\partial \mathbf{N_0}/\partial \mathbf{E}$. The terms in the one direction of the plate are shown below and terms in the perpendicular direction can be obtained by altering the subscripts. The stress-resultants in perpendicular directions in the plate are coupled by making $K_{12} = \nu K_{11}$ etc.

$$K_{11} = \frac{\partial N_{0_1}}{\partial \epsilon_1} = \frac{N_{\delta\epsilon_1} - N_{0_1}}{\delta\epsilon}, \qquad K_{14} = \frac{\partial N_{0_1}}{\partial \phi_1} = \frac{N_{\delta\phi_1} - N_{0_1}}{\delta\phi} \tag{12}$$

$$K_{44} = \frac{\partial N_{0_4}}{\partial \epsilon_4} = \frac{N_{\delta\epsilon_4} - N_{0_4}}{\delta\phi}, \qquad K_{41} = \frac{\partial N_{0_4}}{\partial \phi_1} = \frac{N_{\delta\phi_4} - N_{0_4}}{\delta\epsilon} \tag{13}$$

where: N_{0_i} = is the ith term in the vector $\mathbf{N_0}$
$N_{\delta\epsilon_i}$ = is the ith term in the vector $\mathbf{N}_{\delta\epsilon}$
$N_{\delta\phi_i}$ = is the ith term in the vector $\mathbf{N}_{\delta\phi}$

FEAI uses 1% of the total strain or curvature for values of $\delta\epsilon$ and $\delta\phi$.

The values of the terms in the vector $\Delta\mathbf{E_T}$ are obtained by subtracting the mechanical component from the total values using Eqns. 4 and 5. For the purposes of calculating thermal curvatures, the thermal gradients are assumed to be linear through the section and to increase linearly during the analysis from zero at the start of the heating to the value specified in the ABAQUS input file at the end. This is in contrast to the polynomial temperature profiles used for determining the material properties and hence the stiffness of the section.

3 Modelling the Cardington Slab

3.1 The Cardington Slab

The concrete floor slab used in the Cardington frame was typical of floor slabs used in composite construction. The slab was orthotropic due to ribs running in one direction along its bottom surface. Above the ribs a layer of anti-cracking reinforcement mesh was included. The bottom surface was covered in a layer of steel decking that was bonded to the concrete by means of small protrusions. The cross section parallel to the ribs is shown in Fig. 4.

The material properties of the concrete used in the slab were tested on each of the eight floors in the Cardington frame.[1] The results are shown in Table 1. The concrete strength on the first floor at seven days was considerably above that for all the other floors. It was suggested that this value be treated with a degree of scepticism.[1]

Floor	Crushing strength (N/mm^2)	
	7-day	28-day
1	41.3	48.0
2	25.5	45.8
3	26.6	45.5
4	24.3	46.7
5	26.5	50.7
6	28.3	48.2
7	26.2	48.7
8	25.2	43.7
all floors	28.0	47.1
all floors excluding floor 1	26.1	47.0

Table 1: Table showing the crushing strength of the concrete used in the Cardington frame[1] on various floors.

3.2 Assumptions made during the Modelling of the Cardington Slab

Before a detailed description of the model used to represent the Cardington slab is given the assumptions used in it will be explained and justified. During any analysis of structures, analytical or numerical, some assumptions will be made. The aim must be to make assumptions that do not affect the outcome of analysis significantly, or at least not the parts of the analysis that are of most interest. The goal of this project is to obtain an understanding of the overall behaviour of the Cardington structure under fire conditions. The assumptions made, therefore, are aimed at achieving this goal - local details of behaviour were sometimes ignored to ensure that the overall structural trends were captured in the model. The assumptions made are listed below and then explained more fully. Some of these assumptions are implicit in the FEAST suite and some are made externally to it. It is assumed that:

- That the stress-strain relationships for the constituent materials are unique.
- That the material behaviour was uniaxial
- The Cardington slab has uniform thickness and material properties throughout the area of the building being modelled.
- That the transverse shear stiffnesses of the slab remained constant for all strain and temperature states.

Stress-strain Relationships are Unique This assumption implies that all deformations are reversible and that during unloading materials follow the same stress-strain path as they do during loading. In the post elastic range this is manifestly untrue for all materials. It can only be justified with the argument that engineering problems are principally concerned with regimes where strains are increasing. It should be noted however that in problems where large parts of a structure start to unload any model assuming unique stress-strain relationships would be invalid. An example of such a problem is the cooling of a redundant structure after a fire. In such a case the present model of the Cardington slab would be insufficient.

This assumption also implies that creep strains are either not present or that they are accounted for in the material constitutive relationships. As will be explained, the constitutive models used are based on those in the Eurocodes and these implicitly include some allowance for creep strains.[4,5] In effect then, the models used for modelling the Cardington slab assume "long-term" loading. Although fires only last a few hours at most, creep effects at high temperatures occur relatively quickly and so the assumption of "long term" loading is not unreasonable.

Material Behaviour is Uniaxial In the Cardington slab the reinforcement is a mesh consisting of bars running in perpendicular directions, stressed only along their lengths. Similarly the decking can only be considered to act in the direction parallel to the ribs. This means that all the steel components of the slab were stressed uniaxially and so a uniaxial material model for the steel was justified. By contrast the concrete in the slab was subject to a complex stress state. Biaxial stressing of concrete can lead to strength enhancements of up to 20% and typically of 10% over uniaxial peak values. Ignoring this behaviour is, in effect, using a slightly weaker concrete in the model than was used in the Cardington slab. It was felt that this small inaccuracy was justified on pragmatic grounds. Producing a stress-resultant slab model that included bi-axial material behaviour would have been impractical.

Uniform Properties It is assumed that the geometry of the section of the Cardington slab remains constant and equal to the dimensions given in the reports of the Cardington test.[1,8,9] Also any variation in the strength of the materials in different areas of the building are ignored.

When slabs are being produced it is inevitable that some 'ponding' occurs, that is, deflections caused by the self-weight of the slab result in concrete flowing to the centre of the bays resulting in slightly thicker slabs in these regions. Also, the tolerance achieved when pouring concrete is quite poor; attempts to ensure that the design thickness of a slab is present over the entire floor often result in the average thickness being considerably greater. Some attempt to quantify these effects for the Cardington slab was reported by Rose et al.[10] and also briefly by Huang.[7] Their findings show that the Cardington slab varied in thickness between 124mm and 170mm. It was found that the average thickness was considerably more than the design thickness of 130mm but that 15% of the floor was thinner than this nominal thickness.

It would clearly be impractical to have constantly varying floor slab thickness in a stress-resultant numerical model, even if reliable data were available. Consequently it has been assumed that the dimensions of the slab are equal to those in the design drawings.

The strength of the concrete used in the floor slabs was shown in Table 1 to vary somewhat between floors. No data is available for the variation in strength between different areas of individual floors. In the absence of better information it has been assumed that the strength of the concrete is equal to $48kN/mm^2$. The reason for this value being somewhat above the 28-day average for all floors quoted in Table 1 is that the value recorded on floor eight is less than the values for all the other floors. As no fire tests were performed on floor eight, it was thought best not to let this discrepancy skew the rest of the data. Typically, the strength of concrete rises to be about 20% more than its 28-day strength after 1 year. This increase in strength was not accounted for in the analyses and so the results are more relevant to a new structure. The strength of the steel was assumed to be equal to that quoted in the design notes.

Transverse Shear Stiffnesses Remain Constant As noted in the discussion of Eqn. 1 it is possible to add two transverse shear stiffness terms to the equation thus giving an 8×8 matrix. These two terms are not passed to FEAI by ABAQUS and so can not be varied during an analysis. ABAQUS only allows the two stiffnesses to be defined as constant values in the ABAQUS input file. This limitation would not have a large effect on the analysis of the Cardington slabs. It is easiest to demonstrate that this is the case by taking an analogy with a linear elastic, transversely loaded cantilever. It can be shown[3] that the ratio of bending deflection to shear deflection is $1 : k(h/L)^2$, where k is a constant that for most materials is around unity, h is the depth of the cantilever and L its length. Clearly the shearing deflections only become important for deep cantilevers. Assuming that this argument can be extended to the Cardington slab it is clear that shear deflections are unimportant. With this in mind the transverse shear stiffnesses were given very high values in all the input files so that the predicted transverse shear deflections from finite element analyses would be effectively zero.

Related to this assumption is the assumption that plane sections remain plane during bending. In reality some distortion of initially plane sections would take place as a result of transverse shear deformations. The plane section assumption was implicitly present in early work on the elastic curvature of beams by Euler and Jacob Bernoulli[12] and was applied to plates by Kirchoff in 1850 and to curved plates by Love in 1888.[2,11] Consequently it is often referred to as the Kirchoff-Love hypothesis. It will be assumed here on the basis that because of the high transverse shear stiffnesses the distortions of initially plane sections would be very small. If it were felt necessary to include some allowance for these distortions in the model it would be possible by altering Eqn.6. The strain due to curvature could, for example, be made to vary quadratically with distance from the reference surface rather than just linearly.

3.3 The Model of Cardington Slab Used

In the direction of the ribs the slab was divided into layers, as required by FEAST, in the manner shown in Fig. 4. One complete rib was taken and divided in to three vertical sections, this was necessary to allow different temperature profiles to be represented in the rib section of the slab and the trough section. The reinforcing mesh was modelled using a smeared layer approach, that is, a layer of steel of area equal to the cross sectional area of the reinforcement was added. The strain range over which SRAS calculated values was −0.01 to 0.01 at an interval of 0.005; the curvature range was −0.00049 to 0.005 with an interval of 0.0001 and the temperature range was 0C to 300C with an interval 50C. It was felt that these values would provide data for the overwhelming majority of the strain-curvature-temperature states that the slab would be likely to adopt. The reference surface selected for the calculations was 70mm above the lower surface of the ribs, this approximately coincided with the slab geometric centroid in the direction of the ribs.

Perpendicular to the ribs the slab was modelled in a similar way although there were some noteworthy differences. The ribs were assumed to play no part in the strength in this direction because longitudinal stresses would not be able to pass from rib to rib. The decking was also assumed not to act because of its undulations over the ribs.

The material stress-strain curves used in SRAS were obtained from the relevant Eurocodes[4,5] and are shown in figures 5 and 6. Steel was assumed to behave in the same manner in tension and compression while concrete was taken to have a tensile strength of one tenth of its compressive strength. In FEAI the same material properties were used for the ascending branches of the stress-strain curves but beyond this point the materials were assumed to be infinitely ductile. This was necessary because the Newton-Raphson type solver used in ABAQUS is unable to track the behaviour of structures with negative stiffnesses.

The temperature profiles through the slab were obtained from the experimental data produced by British Steel.[8] For a given reference surface temperature the exact vertical distribution varied slightly between tests and between areas within the same test. For example, the areas of slab directly over beams tended to have a slightly more uniform vertical temperature distribution than those in the middle of the bays. However, these variations were small and it was felt that assuming a uniform vertical temperature distribution for a given reference surface temperature for all areas of the slab was reasonable. An area of slab near the middle of a bay in the first of the Cardington experiments was selected to obtain representative temperature distributions.

The temperature data available gave temperature profiles through the slab at the centre of a trough and at the centre of a rib. Temperature measurements were recorded throughout the test at two minute intervals. To obtain the polynomial curves that were required by FEAST to represent the temperature profiles in the slab, the data was examined and the profiles extracted when the sensor closest to reference surface reached temperatures equal to each of the reference surface temperatures at which profiles were required. These reference surface temperatures were 10C (ambient temperature), 50C, 100C, 150C and 200C. Once this data had been selected fourth order polynomial curves were fitted to all the profiles using a least squares approach and these polynomials specified in FEAST. The curves are shown in Figs. 7 and 8.

The accuracy of temperature profiles obtained parallel to the ribs using this technique was very good because precise curves could be fitted to both the trough and rib parts of the cross-section. Perpendicular to the ribs the situation was more complex because the correct temperature profile depended on whether the section being considered contained a trough or a rib. Since the overall slab behaviour was likely to be governed by the weakest part of the cross section, the temperature profiles obtained for the trough sections of the slab were applied. These profiles were correct for the parts of the slab that were over a trough and also resulted in the hottest therefore weakest section.

References

[1] P.N.R. Bravery. Cardington Large Building Test Facility. Technical report, iki, c1993.

[2] C.R. Calladine. *Theory of Shell Structures.* Cambridge University Press, 1983.

[3] J. Case, Lord Chilver, and C.T.F. Ross. *Strength of Materials and Structures.* Edward Arnold, 3rd edition, 1992.

[4] ENV. *Eurocode 2, Design of concrete structures*, 1992.

[5] ENV. *Eurocode 3, Design of composite steel and concrete structures*, 1994.

[6] Hibbet, Karlson and Sorenson, Providence, Rhode Island, USA. *ABAQUS Users' Manual, Vols I to III, Ver 5.8*, 1998.

[7] Z. Huang, I.W. Burgess, and R.J. Plank. Three Dimensional Modelling of Two Full-scale Fire Tests on a Composite Building. *Proceedings of the Institute of Civil Engineers, Structures and Buildings*, (134):243–255, Aug 1999.

[8] B.R. Kirby. The behaviour of a multi-storey steel framed building subjected to fire attack, experimental data. Technical report, British Steel, 1998.

[9] D.M. Martin, B.R. Kirby, and M.A. O'Connor. Behaviour of a Multistorey, Steel Framed Building Subjected to Natural Fire Effects, Final Report. Technical report, British Steel, 1998. Confidential report.

[10] P.S. Rose, I.W. Burgess, R.J. Plank, and C.G. Bailey. The Influence of Floor Slabs on the Structural Behaviour of Composite Frames. In Lee, editor, *Structures in the New Millennium*, 1997.

[11] S.Timoshenko and S. Woinowsky-Krieger. *Theory of Plates and Shells.* Mc Graw-Hill International Editions, 1959.

[12] S. Timoshenko. *History of Strength of Materials.* Mc Graw-Hill Publishing Company, 1953.

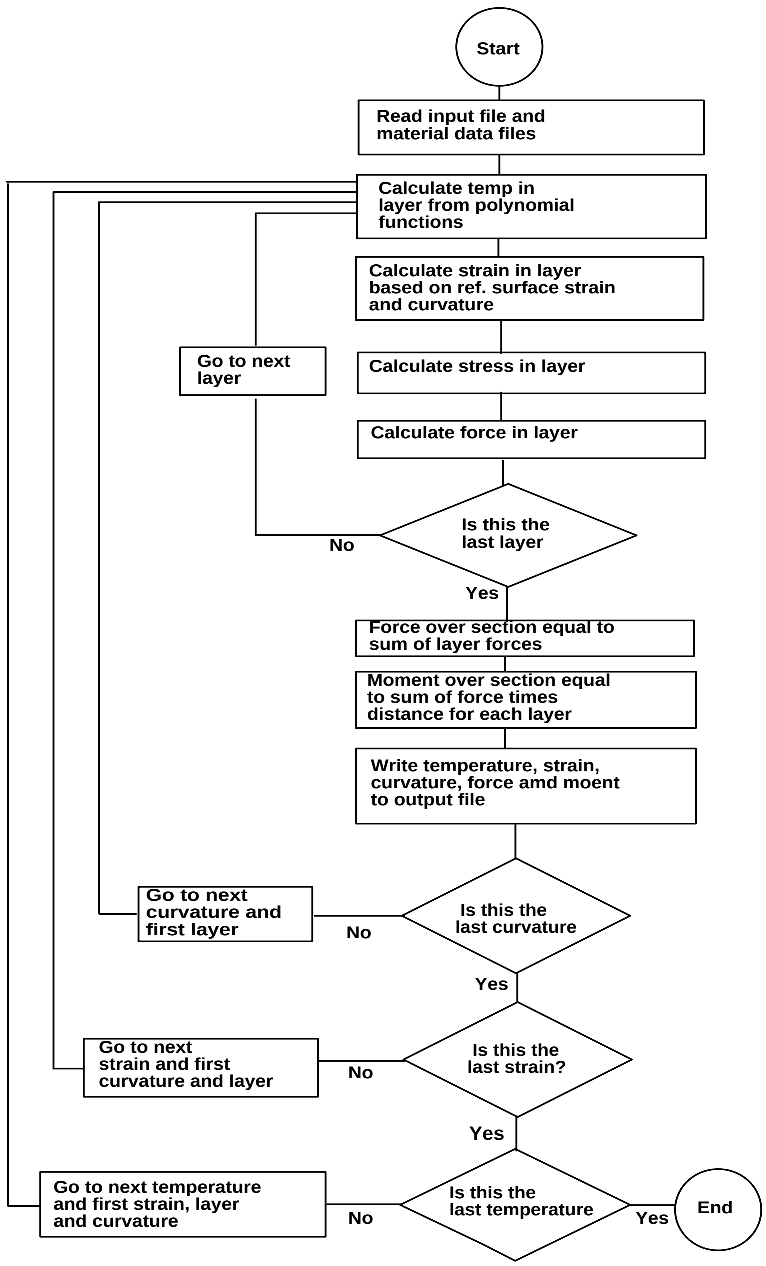

Figure 1: Flowchart describing the program SRAS

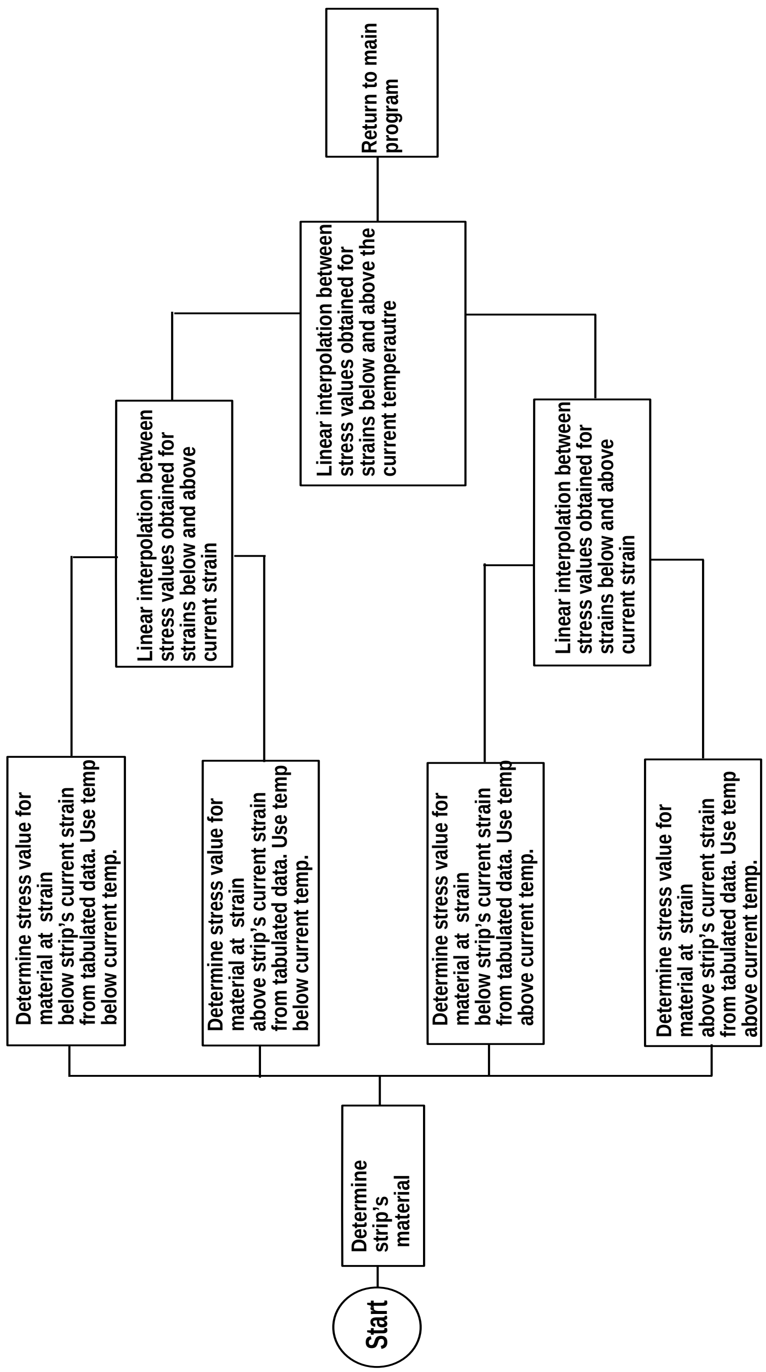

Figure 2: Flowchart describing the details of stress calculation within SRAS

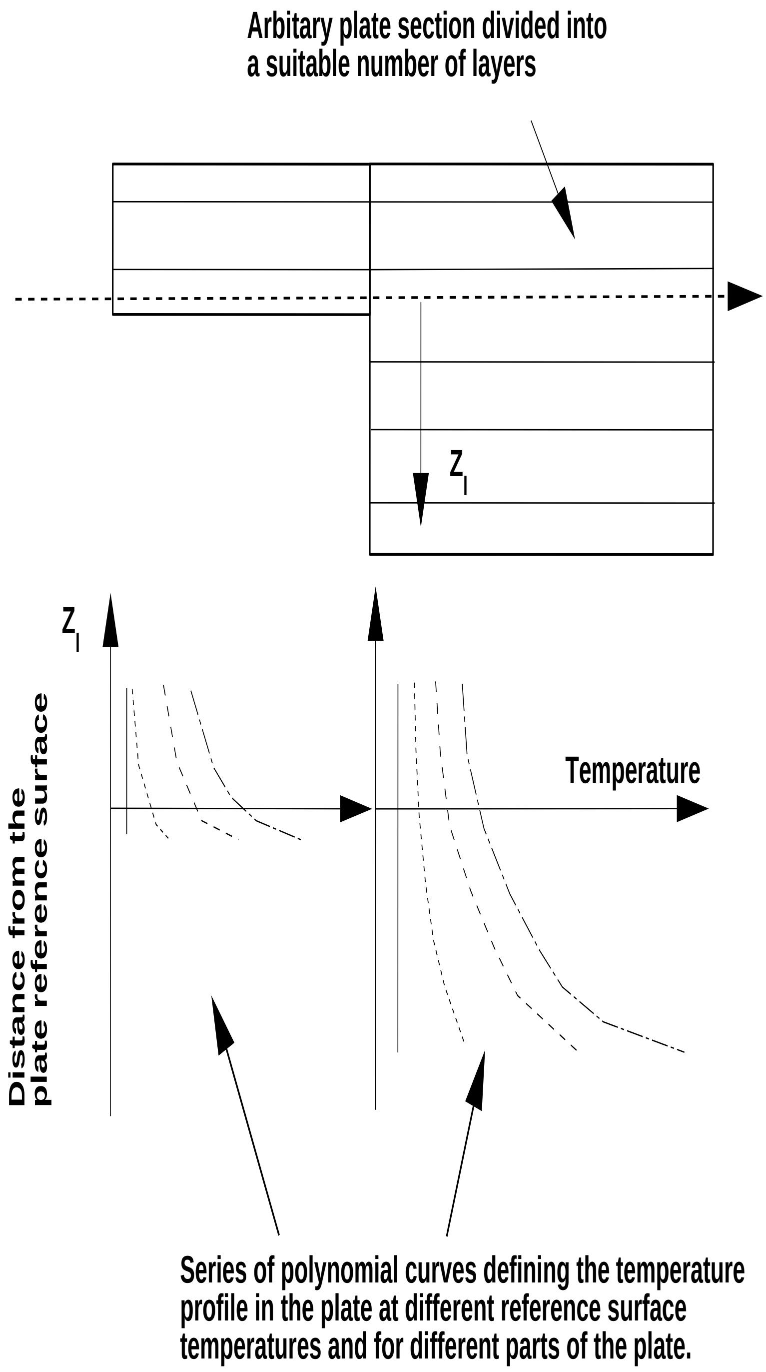

Figure 3: Diagram showing the method used by SRAS for calculating the stresses and temperatures for each layer of an arbitrary plate section

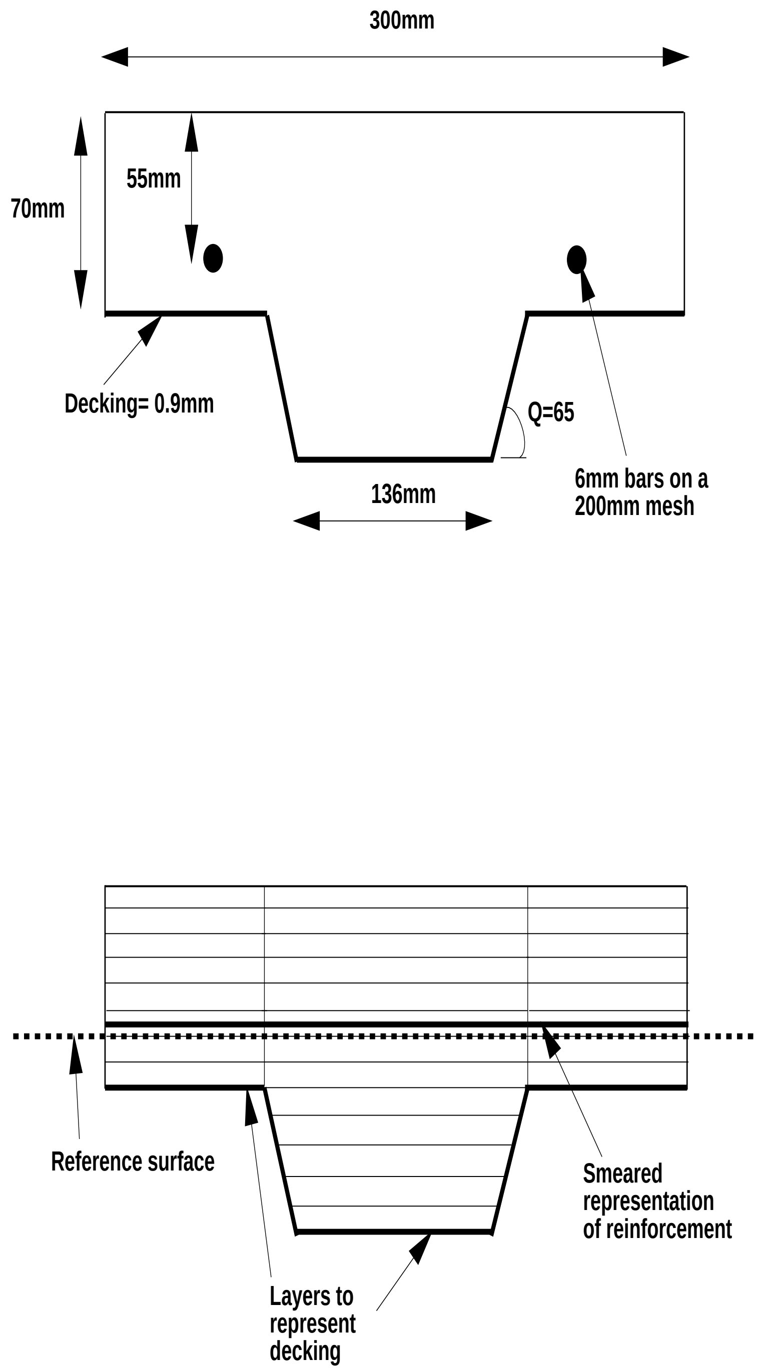

Figure 4: Diagrams showing the dimensions of one rib of the cross section of the Cardington slab and the division of the rib into layers

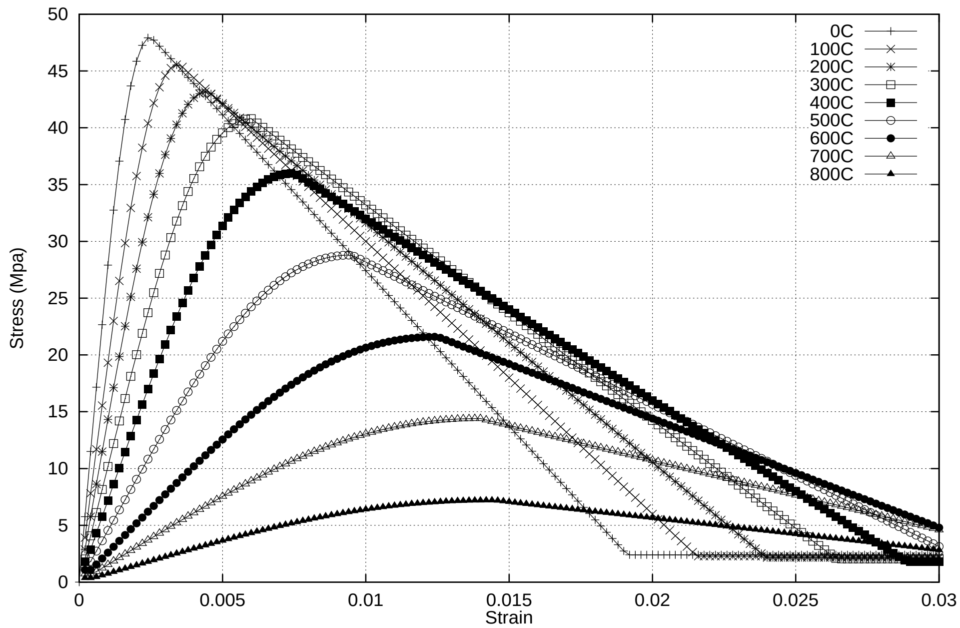

Figure 5: Compressive concrete material behaviour used by SRAS for modelling the Cardington slab

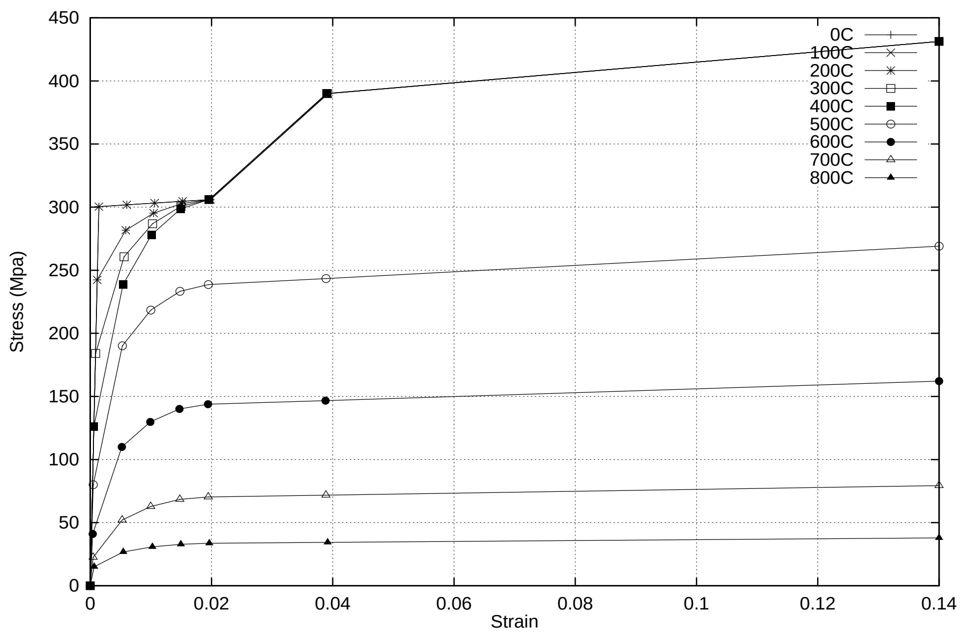

Figure 6: Steel material behaviour used by SRAS for modelling the Cardington slab

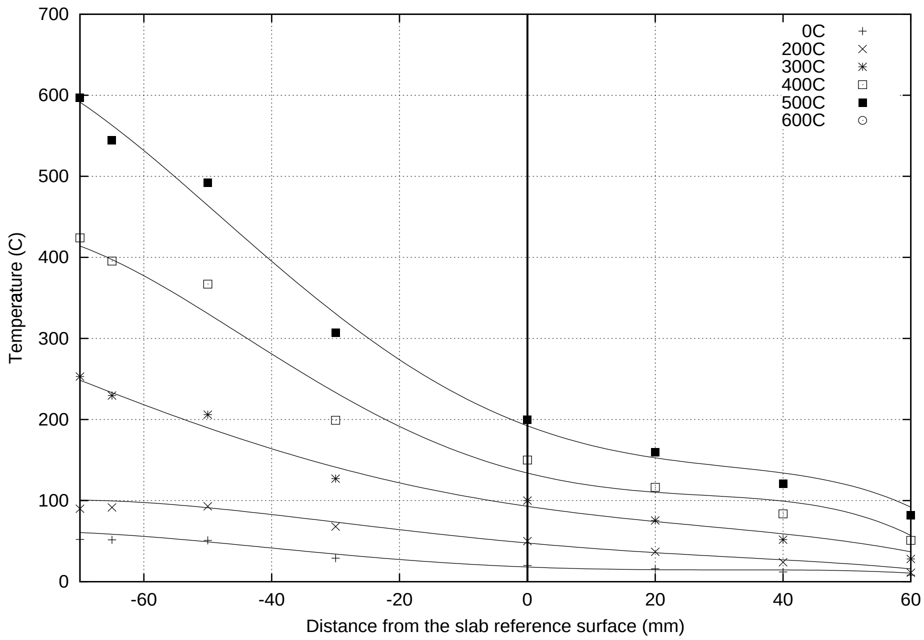

Figure 7: Temperature profiles through a rib of the Cardington Slab with polynomial fits.

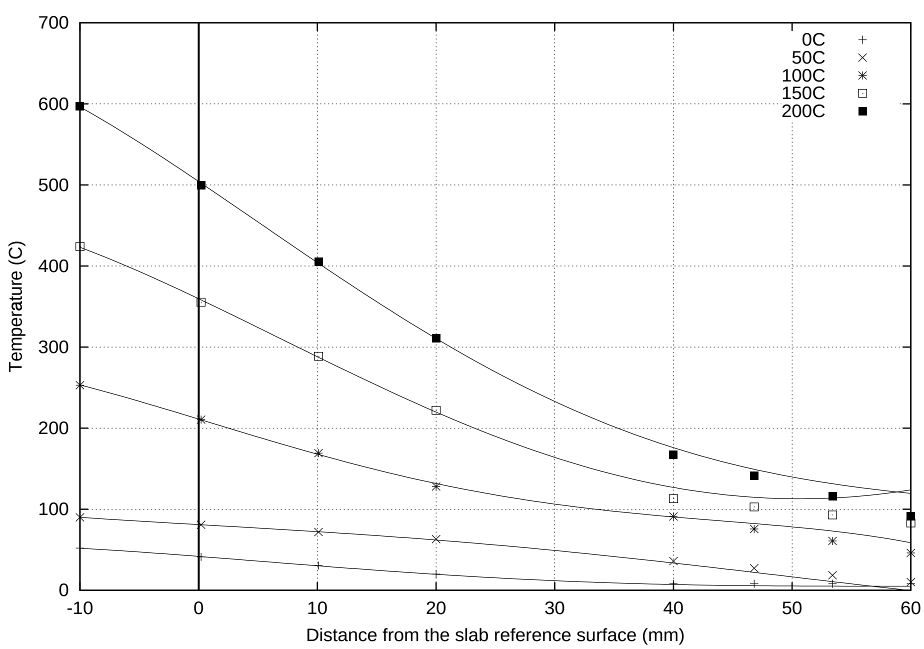

Figure 8: Temperature profiles through a trough of the Cardington Slab with polynomial fits.

A SRAS code listing

```
      PROGRAM SRAS
       IMPLICIT NONE
       INTEGER NOSTRIP,STRIP         !number of strips,current strip
       REAL STRIPWIDTH            !width of section being analysed
       REAL TEMPMIN,TEMPMAX,TEMPINC  !temp range and inc
       REAL STMIN,STMAX,STINC        !strain range and inc
       REAL CVMIN,CVMAX,CVINC        !curve range and inc
       REAL REFDIS                   !dist of ref axis from lower surface
       REAL TEMP,STRAIN,CURVE        !current temp,strain,curve and strip
       REAL STRIPTEMP                !temp of current strip
       REAL MOMENT,FORCE             !
       REAL SDAT(100,4)              !array containing data about shell
       REAL STEEL(89,4),CONC(1360,4),CONCT(46,4) !material data

       CALL OC(1,STEEL,CONCT,CONC)
       CALL RANGES(NOSTRIP,STRIPWIDTH,TEMPMIN,TEMPMAX,TEMPINC,STMIN,STMAX
     +       ,STINC,CVMIN,CVMAX,CVINC,REFDIS,SDAT)
         DO TEMP=TEMPMIN,TEMPMAX,TEMPINC
          DO STRAIN=STMIN,STMAX,STINC
c           DO CURVE=CVMIN,CVMAX,CVINC
             DO STRIP=1,NOSTRIP
                CALL TEMPCALC(TEMP,STRIP,REFDIS,STRIPTEMP,SDAT)
                CALL FORCECALC(STRIPTEMP,CURVE,STRAIN
     +            ,FORCE,MOMENT,REFDIS,STRIP,SDAT,STRIPWIDTH,STEEL,CONCT,
     +             CONC)
             END DO
             CALL OUTPUT(FORCE,MOMENT,TEMP
     +                 ,CURVE,STRAIN)
           END DO
c          END DO
        END DO
       CALL OC(2,STEEL,CONCT,CONC)

       RETURN
       END
C*******************************************************************
       SUBROUTINE OC(IDENT,STEEL,CONCT,CONC)
C      subroutine opens and closes files
       REAL STEEL(89,4),CONC(1360,4),CONCT(46,4)
       INTEGER IDENT

       IF (IDENT.EQ.1) THEN
        OPEN(15,FILE='/disk/u5/mgillie/ugens_store/SRAS/test',
     +              STATUS='OLD')
        OPEN(16,FILE='/disk/u5/mgillie/ugens_store/SRAS/concrete',
     +              STATUS='OLD')
        OPEN(17,FILE='/disk/u5/mgillie/ugens_store/SRAS/concretet',
     +              STATUS='OLD' )
        OPEN(18,FILE='/disk/u5/mgillie/ugens_store/SRAS/steel',
     +              STATUS='OLD' )
        OPEN(19,FILE='/disk/u5/mgillie/ugens_store/SRAS/output',
     +              STATUS='unknown')
```

```
        READ(18,*)STEEL(1,1),STEEL(1,2),STEEL(1,3),STEEL(1,4)
          DO I=2,(STEEL(1,1)*STEEL(1,2))+1
            READ(18,*)STEEL(I,1),STEEL(I,2),STEEL(I,3)
          END DO
        READ(17,*)CONCT(1,1),CONCT(1,2),CONCT(1,3),CONCT(1,4)
          DO I=2,(CONCT(1,1)*CONCT(1,2))+1
            READ(17,*)CONCT(I,1),CONCT(I,2),CONCT(I,3)
          END DO
        READ(16,*)CONC(1,1),CONC(1,2),CONC(1,3),CONC(1,4)
          DO I=2,(CONC(1,1)*CONC(1,2))+1
            READ(16,*)CONC(I,1),CONC(I,2),CONC(I,3)
          END DO
        ELSE
         CLOSE(15)
         CLOSE(16)
         CLOSE(17)
         CLOSE(18)
         CLOSE(19)
       END IF

       RETURN
       END
c*********************************************************************
      SUBROUTINE RANGES(NOSTRIP,STRIPWIDTH,TEMPMIN,TEMPMAX,TEMPINC,STMIN
     +       ,STMAX,STINC,CVMIN,CVMAX,CVINC,REFDIS,SDAT)
C     subroutine reads number of strips,range data, and
C     distance of ref axis from shell lower surface

      INTEGER NOSTRIP,I             !number of strips
      REAL    STRIPWIDTH            !width of the section being considered
      REAL TEMPMIN,TEMPMAX,TEMPINC  !temp range and inc
      REAL STMIN,STMAX,STINC        !strain range and inc
      REAL CVMIN,CVMAX,CVINC        !curve range and inc
      REAL REFDIS                   !dist of ref axis from lower surface
      REAL SDAT(100,4)              !shell data

      READ(15,*)NOSTRIP,STRIPWIDTH
      READ(15,*)TEMPMIN,TEMPMAX,TEMPINC
      READ(15,*)STMIN,STMAX,STINC
      READ(15,*)CVMIN,CVMAX,CVINC
      READ(15,*)REFDIS
      DO I=1,NOSTRIP
        READ(15,*)SDAT(I,1),SDAT(I,2),SDAT(I,3),SDAT(I,4)
      END DO

      RETURN
      END
c*********************************************************************
      SUBROUTINE TEMPCALC(TEMP,STRIP,REFDIS,STRIPTEMP,SDAT)
C     subroutine calculates temp of current strip

      INTEGER STRIP         !current strip
      REAL    TEMP,TEMP1,TEMP2,TEMP3
```

```
      REAL    Y              !dist from ref surface to layer
      REAL    DIST           !dist lower surface of shell to layer
      REAL    REFDIS,AREA    !dist from ref axis to lower surface of shell
      REAL    SDAT(100,4)

      DIST=SDAT(STRIP,1)
      Y=DIST-REFDIS

c      IF (SDAT(STRIP,3).EQ.2)THEN  !ensure decking temp =atmos temp
c         Y=REFDIS*-1
c       END IF

      IF (SDAT(STRIP,4).EQ.2)THEN   !thick sections
       IF (TEMP .LT. 50) THEN
        TEMP1 = -0.000001*Y**4-0.00002*Y**3+0.0078*Y**2-0.3057*Y+18.037
        TEMP2 = -0.000001*Y**4+0.000008*Y**3+0.0061*Y**2-0.7113*Y+47.633
        TEMP3 = 0
       ELSEIF ((TEMP.GE.50).AND.(TEMP.LT.100)) THEN
        TEMP1=-0.000001*Y**4+0.000008*Y**3+0.0061*Y**2-0.7113*Y+47.633
        TEMP2=-0.000001*Y**4-0.0001*Y**3+0.0132*Y**2-1.1518 * Y + 92.883
        TEMP3=50
       ELSEIF ((TEMP.GE.100).AND.(TEMP.LT.150)) THEN
        TEMP1=-0.000001*Y**4-0.0001*Y**3+ 0.0132*Y**2-1.1518*Y+92.883
        TEMP2=-0.000006*Y**4-0.0002*Y**3+0.0447*Y**2-1.9493*Y+133.95
        TEMP3= 100
       ELSEIF ((TEMP.GE.150).AND.(TEMP.LT. 200)) THEN
        TEMP1=-0.000006*Y**4-0.0002*Y**3+0.0447*Y**2-1.9493*Y+133.95
        TEMP2=-0.000006*Y**4-0.0002*Y**3+0.0548*Y**2-2.9453*Y+192.27
        TEMP3=150
       ELSE
        TEMP1=-0.000006*Y**4-0.0002*Y**3+0.0548*Y**2-2.9453*Y+192.27
        TEMP2=TEMP1
        TEMP3=200
       END IF
      ELSE
       IF (TEMP .LT. 50) THEN    !thin sections
        TEMP1=-0.000009*Y**4+0.0011*Y**3-0.0228*Y**2-1.1111*Y+44.311
        TEMP2=-0.000003*Y**4+0.0005*Y**3-0.0258*Y**2-0.7692*Y+85.395
        TEMP3 =0
       ELSEIF ((TEMP.GE.50).AND.(TEMP.LT.100)) THEN
        TEMP1=-0.000003*Y**4+0.0005*Y**3-0.0258*Y**2-0.7692*Y+85.395
        TEMP2= 0.00002*Y**4-0.0028*Y**3+0.1438*Y**2-4.8598*Y+187.06
        TEMP3=50
       ELSEIF ((TEMP.GE.100).AND.(TEMP.LT.150)) THEN
        TEMP1=0.00002*Y**4-0.0028*Y**3+0.1438*Y**2-4.8598*Y+187.06
        TEMP2=-0.00007*Y**4+0.0084*Y**3-0.2375*Y**2-6.5335*Y+391.44
        TEMP3=100
       ELSEIF ((TEMP.GE.150).AND.(TEMP.LT.200)) THEN
        TEMP1=-0.00007*Y**4+0.0084*Y**3-0.2375*Y**2-6.5335*Y+391.44
        TEMP2=-0.00005*Y**4+0.0062*Y**3-0.1443*Y**2-9.6993*Y+521.18
        TEMP3=150
       ELSE
        TEMP1=-0.00005*Y**4+0.0062*Y**3-0.1443*Y**2-9.6993*Y+521.18
        TEMP2=TEMP1
```

```
        TEMP3=200
        END IF
      END IF

      STRIPTEMP=TEMP1+((TEMP2-TEMP1)/(50))*(TEMP-TEMP3)
      IF(TEMP.GT.200) THEN   !special case of ref axis temp above 200C
         STRIPTEMP=TEMP1+(STRIPTEMP-TEMP3)
      END IF

      IF (TEMP.EQ.0) THEN   !SPECIAL CASE AT START OF HEATING
       STRIPTEMP = 0
      END IF

      RETURN
      END
C***************************************************************************
      SUBROUTINE FORCECALC(STRIPTEMP,CURVE,STRAIN
     +                     ,FORCE,MOMENT,REFDIS,STRIP,SDAT,STRIPWIDTH,
     +                      STEEL,CONCT,CONC)
C     subroutine calculates force and moment in the strip due to curvature
C     and strain.  The total is gradually summed over the section as each
C     strip is calculated.  The force and moments are reset in OUTPUT after
C     the end of each strain state.

      REAL STRIPTEMP,STRIPWIDTH  !current strip temp,width of strip
      REAL STRAIN,CURVE          !ref axis strain and curve
      REAL STRIPSTRAIN           !strain in strip
      REAL FORCE, MOMENT         !total force and moment in section
      REAL SDAT(100,4)           !shelldat
      REAL STEEL(89,4),CONC(1360,4),CONCT(46,4) !material data
      REAL STRESS                !stress in strip
      REAL LEVER                 !lever arm of section
      REAL REFDIS                !distance from reference axis to lower surf
      INTEGER STRIP             !No of strip

      STRESS=0
      LEVER=SDAT(STRIP,1)-REFDIS    !calc lever arm of strip

      IF (ABS(CURVE).GT.1E-7)THEN
       STRIPSTRAIN=STRAIN+(-CURVE*LEVER)  !calc strain in strip +ve
      ELSE
       STRIPSTRAIN=STRAIN
      END IF
      CALL STRESSCALC(STRIPSTRAIN,STRESS,STRIPTEMP,SDAT
     +     ,STRIP,STEEL,CONCT,CONC)
      FORCE=FORCE+(SDAT(STRIP,2)*STRESS)/STRIPWIDTH
      MOMENT=MOMENT-(SDAT(STRIP,2)*STRESS*LEVER)/STRIPWIDTH  !-ve for sign

      RETURN
      END
C**************************************************************************
      SUBROUTINE STRESSCALC(STRAIN,STRESS,STRIPTEMP,SDAT,STRIP
     +        ,STEEL,CONCT,CONC)
C     subroutine calculates stress in a layer based on layer's strain and
```

```
c     material properties

      REAL STRAIN,STRESS,STRIPTEMP
      REAL STRESS1,STRESS2
      REAL STRESSA,STRESSB
      REAL MAXSTRAIN       !max tabulated strain
      REAL MATDAT(1359,3)     !material data
      REAL SDAT(100,4)        !shell data
      REAL STEEL(89,4),CONC(1360,4),CONCT(46,4) !material data
      REAL TEMP1,TEMP2        !temps above and below strip temp
      REAL LINTERP            !interpolation function
      INTEGER TEMPLINES       !recs per t emp
      INTEGER NOTEMP,TEMPINC  !no tempincs,tempincs
      INTEGER MATERIAL,ROW,CHECK,STRIP

      CHECK=0
      ROW=0
      STRESS=0

      IF (SDAT(STRIP,3).EQ.1) THEN   !identify material
       IF (STRAIN.LE.0) THEN         !concrete
          TEMPLINES=CONC(1,1)
          NOTEMP=CONC(1,2)
          TEMPINC=CONC(1,3)
          MAXSTRAIN=CONC(1,4)
          DO I=1,NOTEMP*TEMPLINES
            MATDAT(I,1)=CONC(I+1,1)
            MATDAT(I,2)=CONC(I+1,2)
            MATDAT(I,3)=CONC(I+1,3)
          END DO
       ELSE                          ! concrete in tension
          TEMPLINES=CONCT(1,1)
          NOTEMP=CONCT(1,2)
          TEMPINC=CONCT(1,3)
          MAXSTRAIN=CONCT(1,4)
          DO I=1,NOTEMP*TEMPLINES
            MATDAT(I,1)=CONCT(I+1,1)
            MATDAT(I,2)=CONCT(I+1,2)
            MATDAT(I,3)=CONCT(I+1,3)
          END DO
       END IF
      ELSE  IF ((SDAT(STRIP,3).EQ.2).OR.(SDAT(STRIP,3).EQ.3))THEN !steel
          TEMPLINES=STEEL(1,1)
          NOTEMP=STEEL(1,2)
          TEMPINC=STEEL(1,3)
          MAXSTRAIN=STEEL(1,4)
          DO I=1,NOTEMP*TEMPLINES
            MATDAT(I,1)=STEEL(I+1,1)
            MATDAT(I,2)=STEEL(I+1,2)
            MATDAT(I,3)=STEEL(I+1,3)
          END DO
      END IF

      TEMP1=INT(STRIPTEMP/TEMPINC)*TEMPINC   !calc temp above and below
```

```
 TEMP2=TEMP1+TEMPINC
 IF ((TEMP2.GT.TEMPINC*(NOTEMP-1)).OR.(TEMP1.GT.TEMPINC*(NOTEMP-1))
+) THEN
    TEMP1=TEMPINC*(NOTEMP-2)   !prevents going beyond highest temp
    TEMP2=TEMP1+TEMPINC
 END IF
 IF (TEMP2.GT.(NOTEMP-1)*TEMPINC) THEN
  TEMP1=(NOTEMP-1*TEMPINC)
  TEMP2=TEMP1+TEMPINC
 END IF

 ROW=TEMPLINES*INT((TEMP1/TEMPINC))+1 !get to right temp
 DO WHILE ((MATDAT(ROW,2).LT.ABS(STRAIN)).AND.(MATDAT(ROW,1)
+      .LE.TEMP1+1)) ! To get to right strain
   ROW=ROW+1
   CHECK=1
 END DO
 IF (CHECK.EQ.1)THEN   !add one to row if do-loop used
    ROW=ROW-1
 END IF
 IF (MATDAT(ROW,1).GT.TEMP1)THEN  !avoids going beyond end of temp
    ROW=ROW-2
 ELSE IF (MATDAT(ROW+1,1).GT.TEMP1)THEN
    ROW=ROW-1
 END IF

 STRESS1=MATDAT(ROW,3)      !calc stress below strain
 STRESS2=MATDAT(ROW+1,3)    !calc stress above strain
 STRESSA=LINTERP(STRESS1,STRESS2,MATDAT(ROW,2)  !calc stress at low temp
+        ,MATDAT(ROW+1,2),ABS(STRAIN))
 IF (ABS(STRESSA).GT.ABS(STRESS2))THEN
   STRESSA=STRESS2
 END IF
 STRESS1=MATDAT(ROW+TEMPLINES,3)
 STRESS2=MATDAT(ROW+TEMPLINES+1,3)
 STRESSB=LINTERP(ABS(STRESS1),ABS(STRESS2),MATDAT(ROW,2)
+        ,MATDAT(ROW+1,2),ABS(STRAIN))
 IF (STRESSB.GT.STRESS2)THEN
   STRESSB=STRESS2
 END IF
 STRESS=LINTERP(STRESSA,STRESSB,TEMP1,TEMP2,STRIPTEMP)  !calc final stress
 IF (STRIPTEMP.GT.MATDAT(ROW,1))THEN
    STRESS=STRESSB
 END IF

 IF (STRAIN.GE.0) THEN     !correct sign for compression
    STRESS=STRESS !does not effect concrete in tension
   ELSE
    STRESS=STRESS*-1
 END IF
 IF (SDAT(STRIP,3).EQ.2)THEN
  STRESS=0
 END IF
```

```
      RETURN
      END

      FUNCTION LINTERP (Y1,Y2,X1,X2,X)    !linear interpolation function
      REAL Y1,Y2,X1,X2,X,LINTERP
      LINTERP=Y1+((Y2-Y1)/(X2-X1))*(X-X1)
      RETURN
      END

C**************************************************************************
      SUBROUTINE OUTPUT(FORCE,MOMENT,TEMP,CURVE
     +                     ,STRAIN)
C     subroutine outputs forces and moment to output file and sets values
C     to zero

      REAL FPHI,FEPS,MPHI,MEPS   !moment and forces
      REAL TEMP,CURVE,STRAIN     !current temp curve and strain (ref axis)
      REAL FORCE,MOMENT
      REAL DDNDDE11,DDNDDE44

      WRITE(19,10)TEMP,STRAIN,CURVE,FORCE,MOMENT
 10   FORMAT(F4,TR1,F8.5,TR1,F8.5,TR1,F8,TR1,F8)

      FORCE=0
      MOMENT=0

      RETURN
      END
c***************************************************************************
```

B FEAI code listing

```
      SUBROUTINE UGENS(DDNDDE,FORCE,STATEV,SSE,SPD,PNEWDT,STRAN,
     +DSTRAN,TSS,TIME,DTIME,TEMP,DTEMP,PREDEF,DPRED,CENAME,NDI,
     +NSHR,NSECV,NSTATV,PROPS,JPROPS,NPROPS,NJPROP,COORDS,CELENT,
     +THICK,DFGRD,CURV,BASIS,NOEL,NPT,KSTEP,KINC,NIT,LINPER)
C
      INCLUDE 'ABA_PARAM.INC'
C
      CHARACTER*8 CENAME
      DIMENSION  DDNDDE(NSECV,NSECV),FORCE(NSECV),STATEV(NSTATV),
     +STRAN(NSECV),DSTRAN(NSECV),TSS(2),TIME(2),PREDEF(*),
     +DPRED(*),PROPS(*),JPROPS(*),COORDS(3),DFGRD(3,3),
     +CURV(2,2),BASIS(3,3)
C
C     SHELL PARAMETERS
C
      DOUBLE PRECISION V                  !POISSON RATIO FOR SHELL
      DOUBLE PRECISION FORCE1,FORCE2,FORCE3    !FORCES
      DOUBLE PRECISION FORCE4,FORCE5,FORCE6
      DOUBLE PRECISION MOMENT1,MOMENT2,MOMENT3 !MOMENTS
      DOUBLE PRECISION MOMENT4,MOMENT5,MOMENT6
      DOUBLE PRECISION MSTRAN(6)          !MECHANICAL STRAINS
      DOUBLE PRECISION TSTRAN(6)          !TOTAL STRAINS
      DOUBLE PRECISION DTSTRAN(6)         !INCREMENTS OF TOTAL STRAINS
      DOUBLE PRECISION STRAININC(6)       !ARRAY OF STRAIN INCS
      DOUBLE PRECISION ALPHA              !THERMAL EXPANSION COEFFICIENT
      DOUBLE PRECISION GRAD               !THERMAL GRADIENT
      DOUBLE PRECISION STEEL(89,4)               !STEEL DATA
      DOUBLE PRECISION CONC(1360,4)              !CONCRETE COMP DATA
      DOUBLE PRECISION CONCT(46,4)               !CONCRETE TENSION DATA
      DOUBLE PRECISION SDAT(100,4),STNDAT(100,4) !SHELL INPUT FILES
      DOUBLE PRECISION CVMIN,CVMAX,CVINC         !CURV DATA
      DOUBLE PRECISION NCVMIN,NCVMAX,NCVINC
      DOUBLE PRECISION STMAX,STMIN,STINC         !STRAIN DATA
      DOUBLE PRECISION NSTMAX,NSTMIN,NSTINC
      DOUBLE PRECISION TEMPMIN,TEMPMAX,TEMPINC   !TEMP DATA
      DOUBLE PRECISION NTEMPMIN,NTEMPMAX,NTEMPINC
      DOUBLE PRECISION STRIPWIDTH,NSTRIPWIDTH    !WIDTH OF STRIPS
      DOUBLE PRECISION REFDIS,NREFDIS            !REF AXIS DIST
      DOUBLE PRECISION OUTPUTX(6,10000),OUTPUTY(6,10000)
      INTEGER          NOSTRIP,NNOSTRIP,I!NUMBER OF STRIPS
      INTEGER          ITNUM,OUTPUTNUMX,OUTPUTNUMY
      DOUBLE PRECISION KLINTERP,KBLINTERP,KINCREMENT

      COMMON /KSDAT/STEEL,CONC,CONCT
     +,SDAT,STNDAT,CVMIN,CVMAX,CVINC,NCVMIN
     +,NCVMAX,NCVINC,STMAX,STMIN,STINC,NSTMAX,NSTMIN,NSTINC
     +,TEMPMAX,TEMPMIN,TEMPINC,NTEMPMIN,NTEMPMAX,NTEMPINC
     +,STRIPWIDTH,NSTRIPWIDTH,NOSTRIP,NNOSTRIP,REFDIS,NREFDIS
      COMMON /REBAR/OUTPUTX,OUTPUTY,OUTPUTNUMX,OUTPUTNUMY,ITNUM

      ALPHA=PROPS(1)           ! coeff of thermal expansion from input file
      V=0.2                      ! poisson ratio from input file
```

```
      IF (NIT.EQ.0)THEN
       ITNUM=0
      END IF

      IF (KSTEP.EQ.1)THEN
         GRAD=0                          !temp grad from input file
         DGRAD=0
       ELSE IF (KSTEP.EQ.2)THEN          !still 0 if PROPS2=0
         GRAD=(TIME(2)/PROPS(3))*PROPS(2)
         DGRAD=(DTIME/PROPS(3))*PROPS(2)
      END IF

      DO I=1,6                           !initialise stiffness matrix
       DO J=1,6
        DDNDDE(I,J)=0
       END DO
      END DO

      DO I=1,2                          !determine thermal strains
        TSTRAN(I)=ALPHA*(TEMP+DTEMP)    !and curvatures
        DTSTRAN(I)=ALPHA*DTEMP
      END DO
      DO I=4,5
        TSTRAN(I)=ALPHA*(GRAD+DGRAD)
        DTSTRAN(I)=ALPHA*DGRAD
      END DO
      TSTRAN(3)=0
      DTSTRAN(3)=0
      TSTRAN(6)=0
      DTSTRAN(6)=0
      DO I=1,6
       MSTRAN(I)=STRAN(I)-TSTRAN(I)     !determine mech strains and curves
      END DO

      STRAININC(1)=KINCREMENT(MSTRAN(1),1) !use function to calc increments
      STRAININC(2)=KINCREMENT(MSTRAN(2),1) !of strains and curves
      STRAININC(3)=0
      STRAININC(4)=KINCREMENT(MSTRAN(4),2)
      STRAININC(5)=KINCREMENT(MSTRAN(5),2)
      STRAININC(6)=0

c     Calculate force,moment and stiffnesses perpendicular to ribs

      CALL KFORCE(STEEL,CONC,CONCT,SDAT,
     + TEMP,MSTRAN(1),MSTRAN(4),REFDIS,STRIPWIDTH,NOSTRIP,
     + MOMENT1,FORCE1,COORDS,KINC,TIME,NIT,1,OUTPUTX,OUTPUTY,OUTPUTNUMX
     +,OUTPUTNUMY,ITNUM)
      CALL KFORCE(STEEL,CONC,CONCT,SDAT,
     + TEMP,MSTRAN(1)+STRAININC(1),MSTRAN(4),REFDIS,STRIPWIDTH,NOSTRIP,
     + MOMENT2,FORCE2,COORDS,KINC,TIME,NIT,0,OUTPUTX,OUTPUTY,OUTPUTNUMX
     + OUTPUTNUMY,ITNUM)
      CALL KFORCE(STEEL,CONC,CONCT,SDAT,
     + TEMP,MSTRAN(1),MSTRAN(4)+STRAININC(4),REFDIS,STRIPWIDTH,NOSTRIP,
```

```
     + MOMENT3,FORCE3,COORDS,KINC,TIME,NIT,0,OUTPUTX,OUTPUTY,
     + OUTPUTNUMX,OUTPUTNUMY,ITNUM)
      CALL KSTIFFNESS(FORCE1,FORCE2,FORCE3,MOMENT1,MOMENT2,MOMENT3,
     +      DDNDDE(1,1),DDNDDE(1,4),DDNDDE(4,1),DDNDDE(4,4),
     +      STRAININC(1),STRAININC(4))

c   Calculate forces,moments and stiffnesses parallel to ribs

      CALL KFORCE(STEEL,CONC,CONCT,STNDAT,
     + TEMP,MSTRAN(2),MSTRAN(5),NREFDIS,NSTRIPWIDTH,NNOSTRIP,MOMENT4,
     +FORCE4,COORDS,KINC,TIME,NIT,1,OUTPUTX,OUTPUTY,OUTPUTNUMX,OUTPUTNUMY,
     +ITNUM)
      CALL KFORCE(STEEL,CONC,CONCT,STNDAT,
     + TEMP,MSTRAN(2)+STRAININC(2),MSTRAN(5),NREFDIS,NSTRIPWIDTH,
     + NNOSTRIP,MOMENT5,FORCE5,COORDS,KINC,TIME,NIT,0,OUTPUTX,OUTPUTY
     +,OUTPUTNUMX,OUTPUTNUMY,ITNUM)
      CALL KFORCE(STEEL,CONC,CONCT,STNDAT,
     + TEMP,MSTRAN(2),MSTRAN(5)+STRAININC(5),NREFDIS,NSTRIPWIDTH
     +,NNOSTRIP,MOMENT6,FORCE6,COORDS,KINC,TIME,NIT,0,OUTPUTX,OUTPUTY
     +,OUTPUTNUMX,OUTPUTNUMY,ITNUM)
      CALL KSTIFFNESS(FORCE4,FORCE5,FORCE6,MOMENT4,MOMENT5,MOMENT6,
     +      DDNDDE(2,2),DDNDDE(2,5),DDNDDE(5,2),DDNDDE(5,5),
     +      STRAININC(2),STRAININC(5))

      DDNDDE(1,2)=V*DDNDDE(1,1)             !allow for Poisson effects
      DDNDDE(2,1)=V*DDNDDE(2,2)
      DDNDDE(4,5)=V*DDNDDE(4,4)
      DDNDDE(5,4)=V*DDNDDE(5,5)

      IF (ABS(DDNDDE(1,1)).LE.ABS(DDNDDE(2,2))) THEN !determine in-plane shear
        DDNDDE(3,3)=DDNDDE(1,1)                          !stiffness
       ELSE
        DDNDDE(3,3)=DDNDDE(2,2)
      END IF

      IF (ABS(DDNDDE(4,4)).LE.ABS(DDNDDE(5,5))) THEN   !determine twisting
        DDNDDE(6,6)=DDNDDE(4,4)                      !stiffness
       ELSE
        DDNDDE(6,6)=DDNDDE(5,5)
      END IF

      DO I=1,6             !loop calculates increments of forces and moment
        DO J=1,6           !due to the increment of strain
          FORCE(I)=FORCE(I)+DDNDDE(I,J)*(DSTRAN(J)-DTSTRAN(J))
        END DO
      END DO

      IF (ITNUM.LT.NIT)THEN
        ITNUM=NIT
      END IF

      RETURN
      END
```

```
c*********************************************************************
      SUBROUTINE KFORCE(STEEL,CONC,CONCT,SDAT,
     + TEMP,STRAIN,CURVE,REFDIS,STRIPWIDTH,NOSTRIP,MOMENT,FORCE,
     + COORDS,NINC,TIME,NIT,IDENT,OUTPUTX,OUTPUTY,OUTPUTNUMX,OUTPUTNUMY,
     + ITNUM)
      IMPLICIT NONE
      DOUBLE PRECISION  STRIPWIDTH        !width of section being analysed
      DOUBLE PRECISION TEMPMIN,TEMPMAX,TEMPINC  !temp range and inc
      DOUBLE PRECISION STMIN,STMAX,STINC  !strain range and inc
      DOUBLE PRECISION CVMIN,CVMAX,CVINC  !curve range and inc
      DOUBLE PRECISION REFDIS             !dist of ref axis from lower surface
      DOUBLE PRECISION TEMP,STRAIN,CURVE  !current temp,strain,curve and strip
      DOUBLE PRECISION STRIPTEMP          !temp of current strip
      DOUBLE PRECISION MOMENT,FORCE       !
      DOUBLE PRECISION STRESS             !stress in strip
      DOUBLE PRECISION LEVER              !lever arm of strip
      DOUBLE PRECISION STRIPSTRAIN        !strain in a strip
      DOUBLE PRECISION SDAT(100,4)        !array containing data about shell
      DOUBLE PRECISION STEEL(89,4),CONC(1360,4),CONCT(46,4) !material data
      DOUBLE PRECISION COORDS(3)          !coordinates of int point
      DOUBLE PRECISION OUTPUTX(6,10000),OUTPUTY(6,10000)
      DOUBLE PRECISION TIME
      INTEGER NINC,IDENT,NIT          !iteration and integration pt
      INTEGER NOSTRIP,STRIP               !number of strips,current strip
      INTEGER OUTPUTNUMX,OUTPUTNUMY,ITNUM

      FORCE=0
      MOMENT=0
      STRIPTEMP=0

      DO STRIP=1,NOSTRIP
        CALL KTEMPC(TEMP,STRIP,REFDIS,STRIPTEMP,SDAT)
        STRESS=0                    !initialise stress
        LEVER=SDAT(STRIP,1)-REFDIS    !calc lever arm of strip
        IF (ABS(CURVE).GT.1E-7)THEN
         STRIPSTRAIN=STRAIN+(-CURVE*LEVER)  !calc mech  strain in strip +ve
        ELSE
         STRIPSTRAIN=STRAIN
        END IF
        CALL KSTRESSC(STRIPSTRAIN,STRESS,STRIPTEMP,SDAT
     +    ,STRIP,STEEL,CONCT,CONC)
        FORCE=FORCE+(SDAT(STRIP,2)*STRESS)/STRIPWIDTH
        MOMENT=MOMENT-(SDAT(STRIP,2)*STRESS*LEVER)/STRIPWIDTH  !-ve for sign

c Fill the ouput arrays with the current strain data for integration points.
c if ITNUM is less than NIT then a new increment has started and so the array
c line number (OUTPUTNUM) must be reset

        IF ((STRIP.EQ.31).AND.(NOSTRIP.EQ.33).AND.(IDENT.EQ.1))THEN
           IF (ITNUM.LT.NIT) THEN
              OUTPUTNUMY=1
           END IF
          OUTPUTY(1,OUTPUTNUMY)=COORDS(1)
          OUTPUTY(2,OUTPUTNUMY)=COORDS(3)
```

```
        OUTPUTY(3,OUTPUTNUMY)=STRIPTEMP
        OUTPUTY(4,OUTPUTNUMY)=STRIPSTRAIN
        OUTPUTY(5,OUTPUTNUMY)=TIME
        OUTPUTY(6,OUTPUTNUMY)=NINC
        OUTPUTNUMY=OUTPUTNUMY+1 !increase array line by one,stored in common)
      END IF

      IF ((STRIP.EQ.8).AND.(NOSTRIP.EQ.8).AND.(IDENT.EQ.1))THEN
         IF (ITNUM.LT.NIT) THEN
            OUTPUTNUMX=1
         END IF
        OUTPUTX(1,OUTPUTNUMX)=COORDS(1)
        OUTPUTX(2,OUTPUTNUMX)=COORDS(3)
        OUTPUTX(3,OUTPUTNUMX)=STRIPTEMP
        OUTPUTX(4,OUTPUTNUMX)=STRIPSTRAIN
        OUTPUTX(5,OUTPUTNUMX)=TIME
        OUTPUTX(6,OUTPUTNUMX)=NINC
        OUTPUTNUMX=OUTPUTNUMX+1 !increase array line byone,stored in common)
      END IF
     END DO

      RETURN
      END
c**********************************************************************
      SUBROUTINE KTEMPC(TEMP,STRIP,REFDIS,STRIPTEMP,SDAT)
C     subroutine calculates temp of current strip
      IMPLICIT NONE
      INTEGER             STRIP        !current strip
      DOUBLE PRECISION    TEMP,TEMP1,TEMP2,TEMP3
      DOUBLE PRECISION    Y            !dist from ref surface to layer
      DOUBLE PRECISION    DIST         !dist lower surface of shell to layer
      DOUBLE PRECISION    REFDIS,AREA  !dist ref axis-lower surface of shell
      DOUBLE PRECISION    SDAT(100,4)
      DOUBLE PRECISION    STRIPTEMP    !temp of strip

      DIST=SDAT(STRIP,1)
      Y=DIST-REFDIS

      IF (SDAT(STRIP,4).EQ.2)THEN   !thick sections
       IF (TEMP .LT. 50) THEN
        TEMP1 = -0.000001*Y**4-0.00002*Y**3+0.0078*Y**2-0.3057*Y+18.037
        TEMP2 = -0.000001*Y**4+0.000008*Y**3+0.0061*Y**2-0.7113*Y+47.633
        TEMP3 = 0
       ELSEIF ((TEMP.GE.50).AND.(TEMP.LT.100)) THEN
        TEMP1=-0.000001*Y**4+0.000008*Y**3+0.0061*Y**2-0.7113*Y+47.633
        TEMP2=-0.000001*Y**4-0.0001*Y**3+0.0132*Y**2-1.1518 * Y + 92.883
        TEMP3=50
       ELSEIF ((TEMP.GE.100).AND.(TEMP.LT.150)) THEN
        TEMP1=-0.000001*Y**4-0.0001*Y**3+ 0.0132*Y**2-1.1518*Y+92.883
        TEMP2=-0.000006*Y**4-0.0002*Y**3+0.0447*Y**2-1.9493*Y+133.95
        TEMP3= 100
       ELSEIF ((TEMP.GE.150).AND.(TEMP.LT. 200)) THEN
        TEMP1=-0.000006*Y**4-0.0002*Y**3+0.0447*Y**2-1.9493*Y+133.95
        TEMP2=-0.000006*Y**4-0.0002*Y**3+0.0548*Y**2-2.9453*Y+192.27
```

```
        TEMP3=150
       ELSE
        TEMP1=-0.000006*Y**4-0.0002*Y**3+0.0548*Y**2-2.9453*Y+192.27
        TEMP2=TEMP1
        TEMP3=200
       END IF
      ELSE
       IF (TEMP .LT. 50) THEN    !thin sections
        TEMP1=-0.000009*Y**4+0.0011*Y**3-0.0228*Y**2-1.1111*Y+44.311
        TEMP2=-0.000003*Y**4+0.0005*Y**3-0.0258*Y**2-0.7692*Y+85.395
        TEMP3 =0
       ELSEIF ((TEMP.GE.50).AND.(TEMP.LT.100)) THEN
        TEMP1=-0.000003*Y**4+0.0005*Y**3-0.0258*Y**2-0.7692*Y+85.395
        TEMP2= 0.00002*Y**4-0.0028*Y**3+0.1438*Y**2-4.8598*Y+187.06
        TEMP3=50
       ELSEIF ((TEMP.GE.100).AND.(TEMP.LT.150)) THEN
        TEMP1=0.00002*Y**4-0.0028*Y**3+0.1438*Y**2-4.8598*Y+187.06
        TEMP2=-0.00007*Y**4+0.0084*Y**3-0.2375*Y**2-6.5335*Y+391.44
        TEMP3=100
       ELSEIF ((TEMP.GE.150).AND.(TEMP.LT.200)) THEN
        TEMP1=-0.00007*Y**4+0.0084*Y**3-0.2375*Y**2-6.5335*Y+391.44
        TEMP2=-0.00005*Y**4+0.0062*Y**3-0.1443*Y**2-9.6993*Y+521.18
        TEMP3=150
       ELSE
        TEMP1=-0.00005*Y**4+0.0062*Y**3-0.1443*Y**2-9.6993*Y+521.18
        TEMP2=TEMP1
        TEMP3=200
        END IF
      END IF

      STRIPTEMP=TEMP1+((TEMP2-TEMP1)/50)*(TEMP-TEMP3)  !
      IF(TEMP.GT.200) THEN   !special case of ref axis temp above 200C
         STRIPTEMP=TEMP1+(STRIPTEMP-TEMP3)
      END IF

      IF (ABS(TEMP).LE.9) THEN   !special case at the start of heating
       STRIPTEMP =8
      END IF

      RETURN
      END

C************************************************************************
      SUBROUTINE KSTRESSC(STRAIN,STRESS,STRIPTEMP,SDAT,STRIP
     +        ,STEEL,CONCT,CONC)
C************************************************************************
C     subroutine calculates stress in a layer based on layer's strain and
c     material properties
      IMPLICIT NONE

      DOUBLE PRECISION STRAIN,STRESS,STRIPTEMP
      DOUBLE PRECISION STRESS1,STRESS2
      DOUBLE PRECISION STRESSA,STRESSB
      DOUBLE PRECISION MAXSTRAIN   !max tabulated strain
```

```
      DOUBLE PRECISION MATDAT(1359,3)     !material data
      DOUBLE PRECISION SDAT(100,4)        !shell data
      DOUBLE PRECISION STEEL(89,4),CONC(1360,4),CONCT(46,4) !material data
      DOUBLE PRECISION TEMP1,TEMP2        !temps above and below strip temp
      DOUBLE PRECISION KLINTERP           !interpolation function
      DOUBLE PRECISION TEMPLINES       !recs per t emp
      DOUBLE PRECISION NOTEMP,TEMPINC  !no tempincs,tempincs
      INTEGER MATERIAL,ROW,CHECK,STRIP,I

      CHECK=0
      ROW=0

      IF (SDAT(STRIP,3).EQ.1) THEN   !identify material
       IF (STRAIN.LE.0) THEN         !concrete
          TEMPLINES=CONC(1,1)
          NOTEMP=CONC(1,2)
          TEMPINC=CONC(1,3)
          MAXSTRAIN=CONC(1,4)
          DO I=1,NOTEMP*TEMPLINES
            MATDAT(I,1)=CONC(I+1,1)
            MATDAT(I,2)=CONC(I+1,2)
            MATDAT(I,3)=CONC(I+1,3)
          END DO
       ELSE                          ! concrete in tension
          TEMPLINES=CONC(1,1)
          NOTEMP=CONC(1,2)
          TEMPINC=CONC(1,3)
          MAXSTRAIN=CONC(1,4)
          DO I=1,NOTEMP*TEMPLINES
            MATDAT(I,1)=CONC(I+1,1)
            MATDAT(I,2)=CONC(I+1,2)*0.01
            MATDAT(I,3)=CONC(I+1,3)*0.01
          END DO
c          TEMPLINES=CONCT(1,1)
c          NOTEMP=CONCT(1,2)
c          TEMPINC=CONCT(1,3)
c          MAXSTRAIN=CONCT(1,4)
c          DO I=1,NOTEMP*TEMPLINES
c            MATDAT(I,1)=CONCT(I+1,1)
c            MATDAT(I,2)=CONCT(I+1,2)
c            MATDAT(I,3)=CONCT(I+1,3)
c          END DO
       END IF
      ELSE  IF ((SDAT(STRIP,3).EQ.2).OR.(SDAT(STRIP,3).EQ.3))THEN !steel
          TEMPLINES=STEEL(1,1)
          NOTEMP=STEEL(1,2)
          TEMPINC=STEEL(1,3)
          MAXSTRAIN=STEEL(1,4)
          DO I=1,NOTEMP*TEMPLINES
            MATDAT(I,1)=STEEL(I+1,1)
            MATDAT(I,2)=STEEL(I+1,2)
            MATDAT(I,3)=STEEL(I+1,3)
          END DO
      END IF
```

```
 TEMP1=INT(STRIPTEMP/TEMPINC)*TEMPINC   !calc temp above and below
 TEMP2=TEMP1+TEMPINC
 IF ((TEMP2.GT.TEMPINC*(NOTEMP-1)).OR.(TEMP1.GT.TEMPINC*(NOTEMP-1))
+) THEN
    TEMP1=TEMPINC*(NOTEMP-2)   !prevents going beyond highest temp
    TEMP2=TEMP1+TEMPINC
 END IF
 IF (TEMP2.GT.(NOTEMP-1)*TEMPINC) THEN
  TEMP1=(NOTEMP-1*TEMPINC)
  TEMP2=TEMP1+TEMPINC
 END IF

 ROW=TEMPLINES*INT((TEMP1/TEMPINC))+1 !get to right temp
 DO WHILE ((MATDAT(ROW,2).LT.ABS(STRAIN)).AND.(MATDAT(ROW,1)
+      .LE.TEMP1+1)) ! To get to right strain
   ROW=ROW+1
   CHECK=1
 END DO
 IF (CHECK.EQ.1)THEN   !take one from row if do-loop used
    ROW=ROW-1
 END IF
 IF (MATDAT(ROW,1).GT.TEMP1+1)THEN  !avoids going beyond end of temp
    ROW=ROW-2
  ELSE IF (MATDAT(ROW+1,1).GT.TEMP1+1)THEN
    ROW=ROW-1
 END IF

 STRESS1=MATDAT(ROW,3)      !calc stress below strain
 STRESS2=MATDAT(ROW+1,3)    !calc stress above strain
 IF (ABS(STRAIN).GT.MATDAT(ROW+1,2)) THEN
    STRESSA=KLINTERP(STRESS1,STRESS2,MATDAT(ROW,2)  !stress below temp
+        ,MATDAT(ROW+1,2),ABS(STRAIN))
 ELSE
    STRESSA=KLINTERP(STRESS1,STRESS2,MATDAT(ROW,2)
+        ,MATDAT(ROW+1,2),ABS(STRAIN))
 END IF
 STRESS1=MATDAT(ROW+TEMPLINES,3)
 STRESS2=MATDAT(ROW+TEMPLINES+1,3)
 IF (ABS(STRAIN).GT.MATDAT(ROW+1,2)) THEN
    STRESSB=KLINTERP(STRESS1,STRESS2,MATDAT(ROW,2)  !stress above temp
+        ,MATDAT(ROW+1,2),ABS(STRAIN))
 ELSE
    STRESSB=KLINTERP(STRESS1,STRESS2,MATDAT(ROW,2)
+        ,MATDAT(ROW+1,2),ABS(STRAIN))
 END IF
 STRESS=KLINTERP(STRESSA,STRESSB,TEMP1,TEMP2,STRIPTEMP) !calc final stress
 IF (STRIPTEMP.GT.MATDAT(ROW,1))THEN
    STRESS=STRESSB
 END IF

 IF (STRAIN.GE.0) THEN     !correct sign for compression
    STRESS=STRESS
  ELSE
```

```
          STRESS=STRESS*-1
        END IF

        RETURN
        END
C***********************************************************************
C  FUNCTION FOR LINEAR INTERPOLATION
C***********************************************************************
      DOUBLE PRECISION FUNCTION KLINTERP (Y1,Y2,X1,X2,X)
      IMPLICIT NONE
      DOUBLE PRECISION Y1,Y2,X1,X2,X

      KLINTERP=Y1+((Y2-Y1)/(X2-X1))*(X-X1)

      RETURN
      END
C************************************************************************
C  SUBROUTINE FOR CALCULATING STIFFNESSES
C************************************************************************
      SUBROUTINE KSTIFFNESS(FORCE1,FORCE2,FORCE3,MOMENT1,MOMENT2,MOMENT3
     +      ,DDNDDE11,DDNDDE14,DDNDDE41,DDNDDE44,
     +      DEPSILON,DPHI)
      IMPLICIT NONE
      DOUBLE PRECISION  FORCE1,FORCE2,FORCE3       !forces
      DOUBLE PRECISION  MOMENT1,MOMENT2,MOMENT3    !moments
      DOUBLE PRECISION  DDNDDE11,DDNDDE14,DDNDDE41,DDNDDE44 !stiffnesses
      DOUBLE PRECISION  DEPSILON,DPHI         !increments of strain and curve

      DDNDDE11=(FORCE2-FORCE1)/DEPSILON
      DDNDDE14=(FORCE3-FORCE1)/DPHI
      DDNDDE41=(MOMENT2-MOMENT1)/DEPSILON
      DDNDDE44=(MOMENT3-MOMENT1)/DPHI

      RETURN
      END

C***********************************************************************
C  FUNCTION FOR DETERMINING INCREMENTS OF STRAIN
C***********************************************************************
      DOUBLE PRECISION FUNCTION KINCREMENT (STRAIN,TYPE)
      IMPLICIT NONE
      DOUBLE PRECISION STRAIN
      INTEGER          TYPE

      IF (TYPE.EQ.1)THEN                 !strain
        IF (ABS(STRAIN).GT.1E-6) THEN     !check for very small strains
            KINCREMENT=STRAIN*0.01
        ELSE
          KINCREMENT=1E-6
          IF (STRAIN.GT.0)THEN
            KINCREMENT=KINCREMENT
          ELSE
             KINCREMENT=KINCREMENT*-1
         END IF
```

```
       END IF
      ELSE                              !curvature
         IF (ABS(STRAIN).GT.1E-6) THEN   !check for very small curvatures
           KINCREMENT=STRAIN*0.01
         ELSE
            KINCREMENT=1E-6
             IF (STRAIN.GT.0)THEN
               KINCREMENT=KINCREMENT
             ELSE
               KINCREMENT=KINCREMENT*-1
             END IF
         END IF
      END IF

      RETURN
      END
C*************************************************************************
```

C FEAST input files for the Cardington Slab

C.1 Parallel to the Ribs

```
33      300
0       300    50
-0.01        0.01       0.0005
-0.00049     0.0005     0.00001
70
125     90      1    1
115     690     1    1
105     690     1    1
95      690     1    1
85      690     1    1
75      690     1    1
65      690     1    1
55      345     1    1
125     1620    1    2
115     1620    1    2
105     1620    1    2
95      1620    1    2
85      1620    1    2
75      1620    1    2
65      1620    1    2
55      810     1    2
45      1600    1    2
35      1525    1    2
25      1450    1    2
15      1400    1    2
5       1360    1    2
125     690     1    1
115     690     1    1
105     690     1    1
95      690     1    1
85      690     1    1
75      690     1    1
65      690     1    1
55      345     1    1
65      200     3    2   !larger area to account for larger yield strength
27.5    60.8    2    2
0       122.4   2    2
55      100     2    2
```

C.2 Perpendicular to the Ribs

```
8    300
00   300   50
-0.01    0.01     0.0005
-0.00049 0.0005  0.00001
10
65   3000  1  1
55   3000  1  1
45   3000  1  1
35   3000  1  1
25   3000  1  1
15   3000  1  1
05   3000  1  1
15   200   3  1
```