上海交通大学
SHANGHAI JIAO TONG UNIVERSITY

SJTU COMPUTATIONAL MARINE HYDRODYNAMICS LAB
上海交大船舶与海洋工程计算水动力学研究中心

Class-8

NA26018

Finite Element Analysis of Solids and Fluids

万德成

dcwan@sjtu.edu.cn , http://dcwan.sjtu.edu.cn/

上海交通大学

船舶海洋与建筑工程学院
海洋工程国家重点实验室

2020年

Contents

- **Implementation and explanation of the finite element source code for 1-D/2-D problems (FEM1D, FEM2D)**
- **Introduction of more general finite element code - OOFEM**

Background

Finite element method, like most numerical methods, converts a continuum problem to a discrete one (i.e, converting a system with an infinite number of degrees of freedom into one with a finite number of degrees of freedom)

- **The finite element model of a system ultimately represents a set of algebraic equations among the values of the dependent variables of the system at the selected nodes of the domain**
- **The coefficients of the algebraic equations are typically integrals of approximation functions multiplied by the data of the problem**
- **Exact evaluation of these integrals is not always possible because of the algebraic complexity of the data in the mathematical model**

Background

Review for

Numerical integration

Numerical evaluation of integrals, called numerical integration or numerical quadrature, involves approximation of the integrand by a polynomial of sufficient degree, because the integral of a polynomial can be evaluated exactly

$$I = \int_{x_a}^{x_b} F(x)dx$$

In general, a quadrature formula has the form

$$I = \int_{x_a}^{x_b} F(x)dx \approx \sum_{I=1}^{r} F(x_I)W_I$$

where xl are called the quadrature points and Wl, are the quadrature weights

Background

The commonly used numerical integration methods can be classified into two groups

1. The Newton-Cotes formula that employ values of the function at equally spaced points

$$\int_a^b F(x)dx = (b-a)\sum_{I=1}^{r} F(x_I)w_I$$

2. The Gauss-Legendre quadrature formula that employs unequally spaced points

Gauss-Legendre one is the most commonly used numerical integration approach

$$\int_a^b F(x)dx = \int_{-1}^{1} \widehat{F}(\xi)d\xi \approx \sum_{I=1}^{r} \widehat{F}(\xi)w_I$$

Review for

Natural coordinates

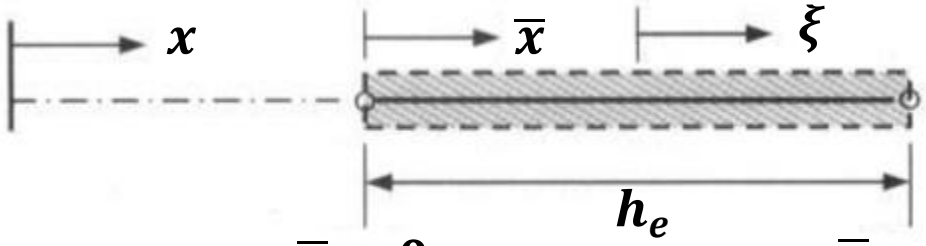
Numerical integration formula requires the integral to be cast as one to be evaluated over the interval [-1, 1]. This requires the transformation of the problem coordinate x to a local coordinate

[x] ------> [ξ]

Consider when $x = x_a$, $\xi = -1$; when $x = x_b$, $\xi = 1$

the transformation takes the form:

$$x(\xi) = x_a + \frac{1}{2} h_e (1 + \xi)$$



local coordinate ξ is called the normal coordinate or natural coordinate, and its values always lie between -1 and 1, with its origin at the center of the element

Background

The local coordinate ξ is useful in two ways: It is

- convenient in constructing the interpolation functions
- required in numerical integration using Gauss-Legendre quadrature

the transformation x=f(ξ) can be written as

$$x = \sum_{i=1}^{m} x_i^e \hat{\psi}_i^e(\xi)$$

The transformation allows us to rewrite integrals involving x as those in terms of

$$\int_{x_a}^{x_b} F(x)dx = \int_{-1}^{1} \hat{F}(\xi)d\xi\,, \qquad \hat{F}(\xi)d(\xi) = F\big(x(\xi)\big)dx$$

The differential element dx in the global coordinate system x is related to the differential element dξ in the natural coordinate system ξ by

$$dx = \frac{dx}{d\xi}d\xi = J_e d\xi$$

Computer implementation

Previous courses were devoted to the finite element formulations of two classes of problems in one dimension

1. **Second-order differential equations (e. g, heat transfer, fluid mechanics, one-dimensional elasticity, bars, and the Timoshenko beam theory)**
2. **Fourth-order differential equations governing the Euler-Bernoulli beam theory. The frame element, obtained by superposing the bar element and the beam element**

- **By now, it should be clear that the steps involved in the finite element analysis of a general class of problems are systematic, and once the finite element formulation is completed, the model can be implemented on computer**

Indeed, the success of the finite element method is largely due to the ease with which the analysis of a class of problems, without regard to the geometry and boundary conditions, can be implemented on a computer

Computer implementation

A particular class of problems can be solved by simply supplying the required input data to the program. If we develop a general computer program to solve equations of the form

$$c_1 \frac{\partial u}{\partial t} + c_2 \frac{\partial^2 u}{\partial^2 t} - \frac{\partial}{\partial x}\left(a \frac{\partial u}{\partial x}\right) + \frac{\partial^2}{\partial x^2}\left(b \frac{\partial^2 u}{\partial x^2}\right) + cu = f$$

then physical problems for any compatible boundary and initial conditions can be solved

Objectives:

- The purpose of this class is to discuss the basic steps involved in the development of a computer program for second-and fourth-order one-dimensional differential equations studied in the preceding classes
- The ideas presented here are used in the development of the model program FEM1D, and they are meant to be illustrative of the steps used in a typical finite element program development
- One can make use of the ideas presented here and implement them using FEM1D to develop a program of ones own

Computer implementation

A typical finite element program consists of three basic units:

1. **Preprocessor**
2. **Processor**
3. **Postprocessor**

In the preprocessor part of the program, the input data of the problem are read in and/or generated. This includes

- **the geometry (e. g, length of the domain and boundary conditions),**
- **the data of the problem (e. g, coefficients in the differential equation),**
- **finite element mesh information (e. g, element type, number of elements, element length, coordinates of the nodes, and connectivity matrix),**
- **indicators for various options (e.g, print, no print, type of field problem analyzed, static analysis, eigenvalue analysis, transient analysis, and degree of interpolation)**

Computer implementation

In the processor part, all the steps in the finite element method discussed in the preceding chapter, except for postprocessing, are performed. The major steps of the processor are :

- Generation of the element matrices using numerical integration
- Assembly of element equations
- Imposition of the boundary conditions
- Solution of the algebraic equations for the nodal values of the primary variables

Computer implementation

In the postprocessor part of the program, the solution is computed by interpolation at points other than nodes, secondary variables that are derivable from the solution are computed, and the output data are processed in a desired format for printout and/or plotting

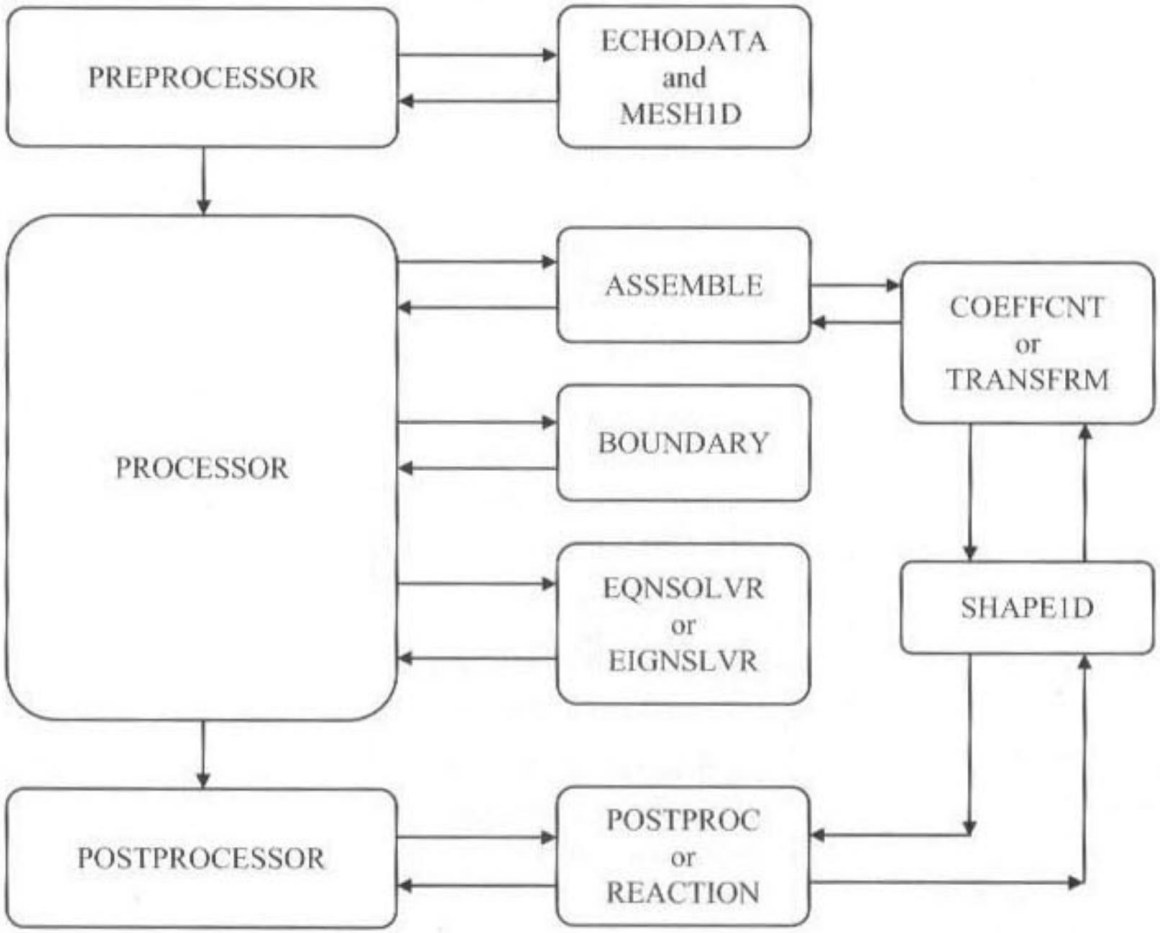
Implementation and Functions

- The preprocessor and postprocessors can be a few Fortran statements to read and print information, subroutines to generate mesh and compute the gradient of the solution

- The processor, where typically large amounts of computing time are spent can consist of several subroutines, each having a special purpose (e. g, a subroutine for the calculation of element matrices, a subroutine for the imposition of boundary conditions, and a subroutine for the solution of equations)

Computer implementation

A flow chart of the computer program FEMID is presented in Fig. The objective of each of the subroutines listed in the flow chart is described below



- **The degree of sophistication and the complexity of a finite element program depend on the general class of problems being programmed, the generality of the data in the equation, and the intended user of the program**
- **It is always desirable to describe, through comment statements, all variables used in the computer program**

Computer implementation

ASSEMBLE: Subroutine for the assembly of element equations.
BOUNDARY: Subroutine for imposition specified BC
COEFFCNT: Subroutine for computing element matrices [Ke], [Me] and [f] for all model problems
ECHODATA: Subroutine to echo the input to the program
MESHID: Subroutine to generate the mesh
POSTPROC: Subroutine to postprocess the solution for all model problems except for truss and frame elements
REACTION: Subroutine to calculate the reactions for truss and frame elements
SHAPEID: Subroutine to compute the approximation functions and their derivatives
EQNSOLVR: Subroutine for solving banded symmetric system of algebraic equations.
TRANSFRM: Subroutine to compute element stiffness matrix and force vector for truss and frame elements

Computer implementation

Preprocessor

MESH

- The preprocessor that deals with the generation of finite element mesh can be separated into a subroutine (MESH1D), depending on the convenience and complexity of the program
- The Mesh input data to a finite element program consist of element type IELEM (i. e, Lagrange element or Hermite element), number of elements used (NEM)
- Mesh generation includes computation of the global coordinates XI, and the connectivity array [B]=NOD
- If a uniform mesh is used, the length of the domain should be read in, and global coordinates of the nodes can be generated in the program

The connectivity array describes the relationship between element nodes to global nodes as

NOD(*I* , *J*)= *global node number* corresponding to the *J*th (local) node of element *I*

Computer implementation

Calculation of Element Matrices（Processor）

The most significant function of a processor is to generate element matrices. The element matrices are computed in various subroutines (COEFFCNT and TRANSFRM), depending on the type of problem being solved

- These subroutines typically involve numerical evaluations of the element matrices [Ke] and [Me] (program variables ELK and ELM) and the element vector {fe} (program variable ELF) for various field problems
- The Gauss Legendre quadrature is used to evaluate element matrices and vectors, and the arrays are assembled as soon as they are computed
- A loop on the number of elements in the mesh (NEM) is used to compute element matrices and assemble them (subroutine ASSEMBLE). It is here that the connectivity array NOD plays a crucial role. By putting one element matrix into global locations at a time, we avoid the computation of all element matrices at once

Computer implementation

Element matrices for different model equations（MODEL）and type of problem（NTYPE）are generated by assigning values

MODEL =1, NTYPE=0

heat-transfer-type problems

$$-\frac{d}{dx}\left(a\frac{du}{dx}\right)+cu-f=0;\ AX=a(x)\,,CX=c(x)\,,FX=f(x)$$

MODEL=2, NTYPE=0

reduced integration element, RIE

MODEL=2, NTYPE=2

consistent interpolation element, CIE

Bending of straight beams using Timoshenko beam theory

$$-\frac{d}{dx}\left[GAK_s\left(\psi+\frac{dw}{dx}\right)\right]+c_f w=q$$

$$-\frac{d}{dx}(EI\frac{d\psi}{dx})+GAK_s(\psi+\frac{dw}{dx})=0$$

$$AX=GAK_s\,,\ CX=c_f(x)\,,\ FX=q(x)\,,\ BX=EI$$

Computer implementation

MODEL=3, NTYPE=0

Bending of straight beams using the Euler-Bernoulli beam theory

$$\frac{d^2}{dx^2}\left(EI\frac{d^2w}{dx^2}\right)+c_f w=q(x),\ BX=EI,\ CX=c_f(x),\ FX=q(x)$$

MODEL=4, NTYPE=0

Two-node truss element

MODEL=4, NTYPE=1

Two-node Euler-Bernoulli frame element

MODEL=4, NTYPE=2

Two-node Timoshenko frame element

Computer implementation

- The time-dependent option is exercised through variable **ITEM**

ITEM=0 : static analysis
ITEM=1 : first-order time derivative（i.e, parabolic）problems
ITEM=2 : second-order time derivative（i.e, hyperbolic）problems

- The element shape functions SF and their derivatives GDSF are evaluated at the gauss points in subroutine **SHAPEID**

- The Gaussian weights and points associated with 2-, 3-, 4-, and 5-point integration are stored in arrays **GAUSWT** and **GAUSPT**, respectively

e.g. nth column of GAUSWT contains the weights corresponding to the n-point Gauss-Legendre quadrature rule

GAUSPT(*i* , *n*) = *i*th Gauss point corresponding to the *n*-point Gauss rule

Computer implementation

The variable **NGP** is used to denote the number of Gauss points,

- As noted earlier, the linear, quadratic and cubic interpolation functions require 2, 3, and 4 quadrature points, respectively, to evaluate the element coefficients exactly

IELEM is the element type

$$\text{IELEM}=\begin{cases} 1 & \text{linear} \\ 2 & \text{quadratic (Lagrange elements)} \\ 3 & \text{cubic} \end{cases}$$

Computer implementation

The Gauss-Legendre quadrature formula can be implemented in the computer as follows: Consider Keij of the form

$$K_{ij}^{e} = \int_{x_a}^{x_b} \left[a(x) \frac{d\psi_i^e}{dx} \frac{d\psi_j^e}{dx} + c(x) \psi_i^e \psi_j^e \right] dx$$

We use the following program variables for the quantities

$$\mathrm{ELK(I,J)} = K_{ij}^{e}\ ,\ \mathrm{SF(I)} = \psi_i^e\ ,\ \mathrm{GDSF(I)} = \frac{d\psi_i^e}{dx}$$

$$\mathrm{AX} = a(x)\ ,\ \ \mathrm{CX} = c(x)\ ,\ \mathrm{ELX(I)} = x_i^e$$

$$\mathrm{NPE} = n \text{ (the number of nodes in the element)}$$

transforming x to ξ $\quad x = \sum_{i=1}^{n} x_i^e \psi_i^e$

$$K_{ij}^{e} = \int_{-1}^{1} \left[a(\xi) \frac{1}{J} \frac{d\psi_i^e}{d\xi} \frac{1}{J} \frac{d\psi_j^e}{d\xi} + c(\xi) \psi_i^e \psi_j^e \right] J d\xi \equiv \int_{-1}^{1} G_{ij}^{e}(\xi) J d\xi$$

$$= \sum_{I=1}^{NGP} G_{ij}^{e}(\xi_I) J W_I$$

Computer implementation

$$K_{ij}^e = \int_{-1}^{1} \left[a(\xi) \frac{1}{J} \frac{d\psi_i^e}{d\xi} \frac{1}{J} \frac{d\psi_j^e}{d\xi} + c(\xi) \psi_i^e \psi_j^e \right] J d\xi \equiv \int_{-1}^{1} G_{ij}^e(\xi) J d\xi$$

$$= \sum_{I=1}^{NGP} G_{ij}^e(\xi_I) J W_I$$

G_{ij}^e denotes the expression in the square brackets

(ξ_I, W_I) denotes l th Gauss point and weight

3 free indices: i, j, l; where l is Gauss-point loop
We evaluate Gij at the Gauss point ξl for each i and j, multiply with the Jacobian J and the weights Wl, and sum over the range of l:

$$\mathbf{ELK}(i,j) = \mathbf{ELK}(i,j) + G_{ij}^e(\xi_I) J W_I$$

Computer implementation

Since [Ke], [Me], [fe] are evaluated for e=1, 2,⋯, NEM, where **NEM** denotes the number of elements in the mesh, we must initialize all arrays that are being evaluated using the Gauss-Legendre quadrature. The initialization must be made outside of the gauss egendre quadrature loop

$$K_{ij}^{e} = \int_{-1}^{1} \left[a(\xi) \frac{1}{J} \frac{d\psi_i^e}{d\xi} \frac{1}{J} \frac{d\psi_j^e}{d\xi} + c(\xi) \psi_i^e \psi_j^e \right] J d\xi$$

The computation of coefficients Keij requires evaluation of a(ξ), c(ξ), ψei, and dψei/dξ at the Gauss point ξl. Hence, inside the loop on l, we call subroutine SHAPEID to evaluate ψi, dψi/dx= (dψi/dξ)/J

Computer implementation

Fortran statements to evaluate [Ke] and {fe} are given below

```
      DO 100 NI = 1,NGP
      XI = GAUSPT(NI,NGP)
C     Call subroutine SHP1D to evaluate the interpolation functions
C     (SF) and their global derivatives (GDSF) at the Gauss point XI
      CALL SHP1D(XI,NPE,SF,GDSF,GJ)
      CONST = GJ*GAUSWT(NI,NGP)
      DO 20 I = 1,NPE
20    X = X + SF(I)*ELX(I)
      AX = AX0 + AX1*X
      CX = CX0 + CX1*X
      DO 30 J = 1,NPE
      ELF(J) = ELF(J) + CONST*SF(J)*FX
      DO 30 I = 1,NPE
30    ELK(I,J) = ELK(I,J) + CONST*(AX*GDSF(I)*GDSF(J)
                + CX*SF(I)*SF(J))
```

$$K_{ij}^{e} = \int_{-1}^{1} \left[a(\xi) \frac{1}{J} \frac{d\psi_i^e}{d\xi} \frac{1}{J} \frac{d\psi_j^e}{d\xi} + c(\xi) \psi_i^e \psi_j^e \right] J d\xi$$

Computer implementation

Assembly of Element Equations（Processor）

- **The assembly of element equations should be carried out as soon as the element matrices are computed, rather than waiting till the element coefficients of all the elements are computed**

The latter requires storage of the element coefficients of each element. In former case, we can perform the assembly in the same loop in which a subroutine is called to calculate element matrices

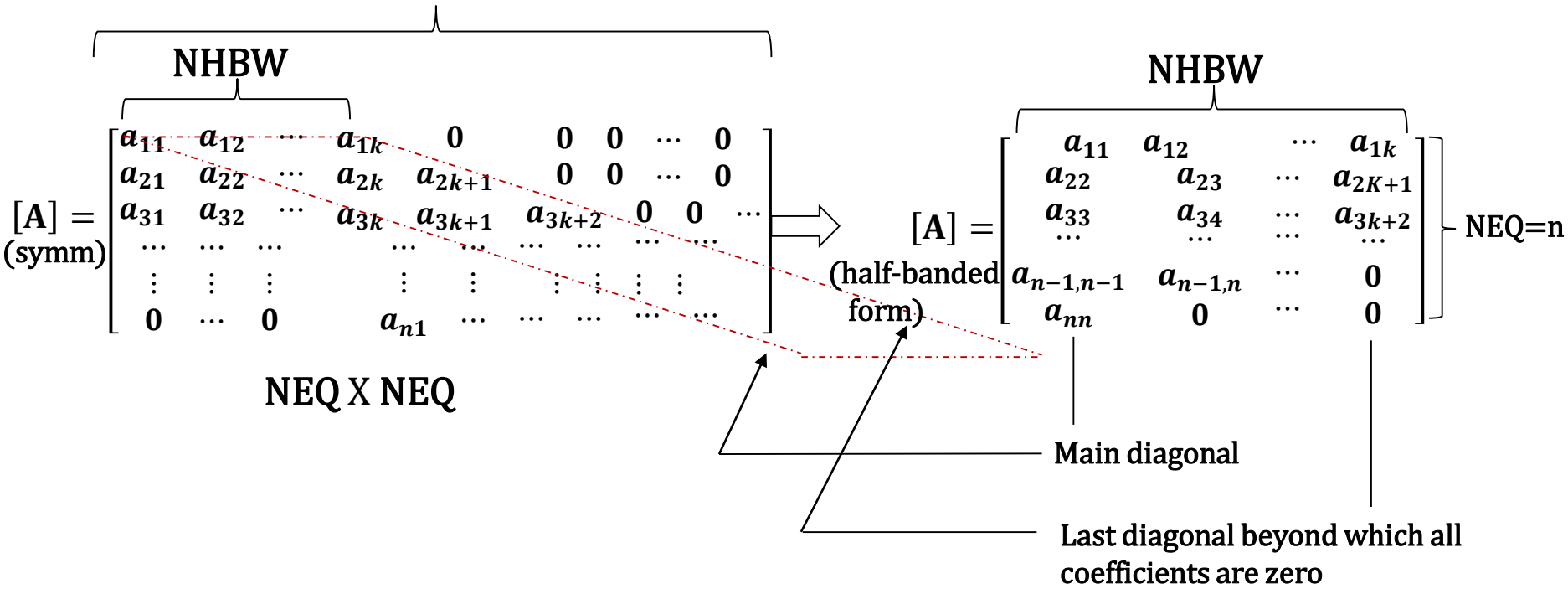
- **A feature of the finite element equations that enables us to save storage and computing time is the assembly of element matrices in upper-banded form**

When element matrices are symmetric, the resulting global （or assembled) matrix is also symmetric, with many zeros away from the main diagonal. Therefore, it is sufficient to store only the upper half-band of the assembled matrix

Computer implementation

Half bandwidth of a matrix:

- Let Ni, be the number of matrix elements between the diagonal element and the last nonzero element in the ith row after which all elements in that row are zero: the **half-bandwidth is the maximum of (Ni+1)xNDF**, where NDF is the number of degrees of freedom per node



Computer implementation

- The half-bandwidth **NHBW** of the assembled finite element matrix can be determined in the finite element program itself
- The local nature of the finite element interpolation functions is responsible for the banded character of the assembled matrix. If two global nodes do not belong to the same element, then the corresponding entries in the global matrix are zeros

$$K_{IJ} = 0$$ if global nodes I and J do not correspond to local nodes of the same element

This property enables us to determine the half-bandwidth **NHBW** of the assembled matrix

$$\mathrm{NHBW} = \max_{\substack{1 \le N \le \mathrm{NEM} \\ 1 \le I,J \le \mathrm{NPE}}} \{\mathrm{abs}[\mathrm{NOD}(N,\ I) - \mathrm{NOD}(N,\ J)] + 1\} \times \mathrm{NDF}$$

NEM = number of elements in the mesh
NPE = number of nodes per element
NDF = number of degrees of freedom per element

Computer implementation

For example, for one-dimensional problems with elements connected in series, the maximum difference between the nodes of an element is equal to NPE-1, Hence

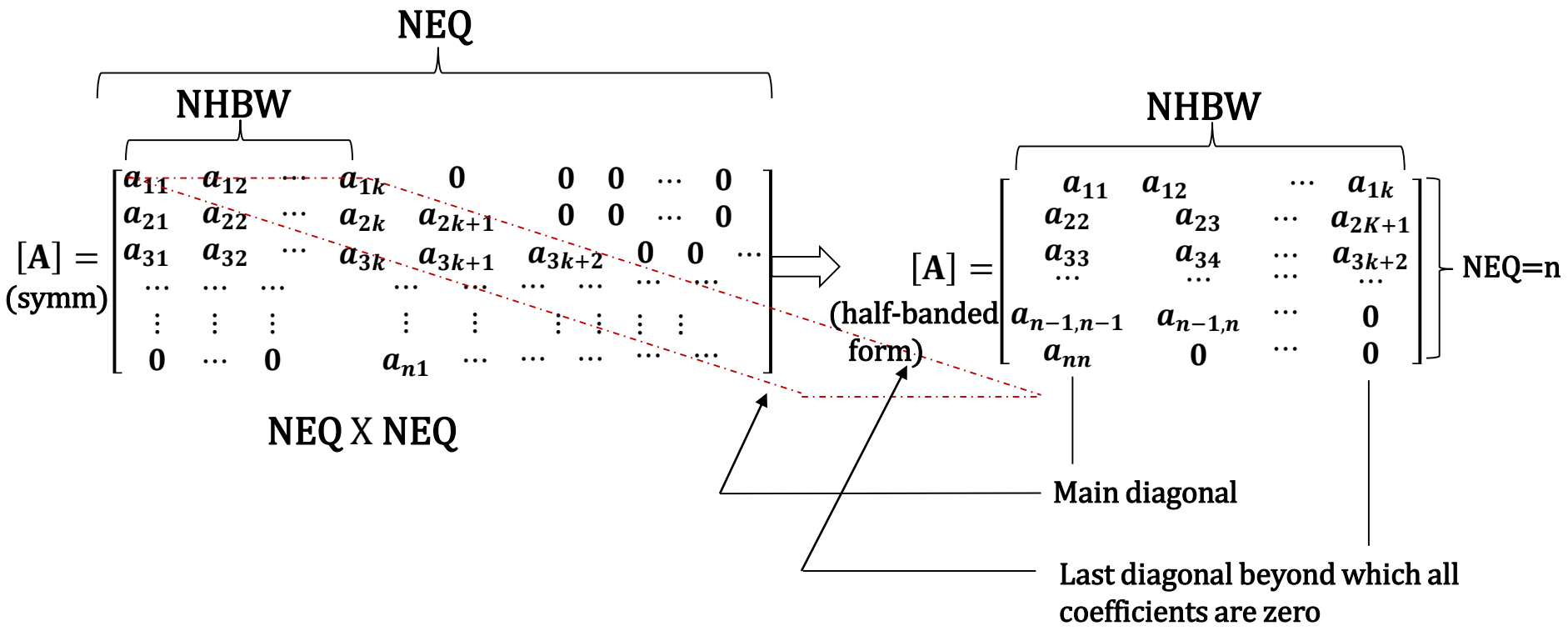
$$\mathbf{NHBW} = [(\mathbf{NPE} - \mathbf{1}) + \mathbf{1}] \times \mathbf{NDF} = \mathbf{NPE} \times \mathbf{NDF}$$

NHBW is always less than or equal to the number of primary degrees of freedom in the mesh, i.e., the number of equations, **NEQ**

Logic for assembling the element:

- Matrices Keij will be assembled into the upper-banded form of the global coefficients Kij. The assembly can be skipped whenever J<I and J>NHBW.
- The main diagonal, I=J, of the assembled square matrix, becomes the first column of the assembled banded matrix

Computer implementation



- The upper diagonals (parallel to the main diagonal) take the position of respective columns in the banded matrix. Thus, the banded matrix has dimension NEQ X NHBW, where NEQ denotes the total number of equations in the problem

Computer implementation

- The element coefficients Knij and fni of a typical element Ωn are to be assembled into the global coefficients matrix [K] and source vector {F}, respectively. If the ith node of the element is equal to the Ith global node and the jth node of the element is equal to the Jth global node, we have

$$K_{IJ} = K^n_{ij}, \qquad F_I = F^n_i \qquad \text{(for NDF = 1)}$$

The values of I and J can be obtained with the help of array NOD

$$I = \mathrm{NOD}(n, i), \qquad J = \mathrm{NOD}(n, j)$$

- Recall that it is possible that the same I and J may correspond to a pair of i and j of some other element Ωm.In that case, Kmij will be added to existing coefficients KIJ, during the assembly

Computer implementation

Imposition of Boundary Conditions (Processor)

Imposition of boundary conditions on the primary and secondary global degrees of freedom can be carried out through subroutine BOUNDARY. There are 3 types of boundary conditions for any problem

- **Essential boundary conditions**

i. e, boundary conditions on primary variables（Dirichlet boundary conditions）

- **Natural boundary conditions**

i. e, boundary conditions on secondary variables（Neumann boundary conditions）

- **Mixed boundary conditions**

Newton boundary conditions

The procedure for implementing the boundary conditions on the primary variables involves modifying the assembled coefficient matrix（GLK） and right-hand column vector

Computer implementation

Procedure for implementing the boundary conditions

Step 1: Moving the known products to the right-hand column of the matrix equation

Step 2: Replacing the columns and rows of GLK corresponding to the known primary variable by zeros, and setting the coefficient on the main diagonal to unity

Step 3: Replacing the corresponding component of the right-hand column by the specified value of the variable

Computer implementation

Consider the following N algebraic equations in full matrix form

$$\begin{bmatrix} K_{11} & K_{12} & K_{13} & \cdots \\ K_{21} & K_{22} & K_{23} & \cdots \\ K_{31} & K_{32} & K_{33} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \begin{Bmatrix} U_1 \\ U_2 \\ U_3 \\ \vdots \end{Bmatrix} = \begin{Bmatrix} F_1 \\ F_2 \\ F_3 \\ \vdots \end{Bmatrix}$$

here UI, and FI are the global primary and secondary degrees of freedom, respectively and KIJ are the assembled coefficients. Suppose that Us=U_s is specified

1. Set Kss=1 and Fs=U_s
2. Set Ksi=Kis=0 for/=1, 2,⋯, N and I/=S

Computer implementation

For S=2, the modified equations are

$$\begin{bmatrix} K_{11} & 0 & K_{13} & K_{14} & \cdots & K_{1n} \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ K_{31} & 0 & K_{33} & K_{34} & \cdots & K_{3n} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ K_{n1} & 0 & K_{n3} & K_{n4} & \cdots & K_{nn} \end{bmatrix} \begin{Bmatrix} U_1 \\ U_2 \\ U_3 \\ \vdots \\ U_n \end{Bmatrix} = \begin{Bmatrix} \widehat{F_1} \\ \widehat{U_2} \\ \widehat{F_3} \\ \vdots \\ \widehat{F_n} \end{Bmatrix}$$

where $\widehat{F_i} = F_i - K_{i2}\widehat{U_2} \qquad (i = 1, 3, 4, 5, \ldots\ldots n;\ \ i \neq 2)$

In general, if Us=U_s is known, we have

$$K_{SS} = 1,\ \ F_S = \widehat{U_S};\qquad \widehat{F_i} = F_i - K_{iS}\widehat{U_S};\qquad K_{Si} = K_{iS} = 0$$

where $i = 1, 2, \ldots\ldots, S-1, S+1, \ldots., n;\ \ i \neq S$

- It enables us to retain the original order of the matrix
- The specified boundary conditions on the primary degrees of freedom are printed as part of the solution
- The logic should be implemented for a banded system of equations

Computer implementation

The specified secondary degrees of freedom (Qi) are implemented directly by adding their specified values to the computed values

- Suppose that the point source corresponding to the Rth secondary degree of freedom is specified to be F_R. Then

$$F_R = f_R + \widehat{F}_R$$

where fR is the contribution due to the distributed source f(x); fR will be computed as a part of the element computations and assembled

Mixed-type boundary conditions are of the form

$$a\frac{du}{dx} = k(u - \overline{u}) = 0 \quad (\overline{u} \text{ and } k \text{ are known constants})$$

which contains both the primary variable u and the secondary variable adu/dx. Thus, adu/dx at the node p is replaced by –kp(up-u_p)

$$Q_p = -k_p(U_p - \overline{U}_p)$$

Computer implementation

This amounts to modifying Kpp by adding kp to its existing value

$$K_{pp} \leftarrow K_{pp} + k_p$$

and adding kpU_p to Fp

$$F_p \leftarrow F_p + k_p \bar{U}_p$$

Following variable names are used in the subroutine BOUNDARY

NSPV	Number of specified primary variables
NSSV	Number of specified secondary variables
NNBC	Number of Newton boundary conditions
VSPV	Column of the specified values $\bar{U}_S$ of primary variables
VSSV	Column of the specified values $\bar{F}_R$ of secondary variables
VNBC	Column of the specified values k_p
UREF	Column of the specified values $\bar{U}_p$
ISPV	Array of the global node and degree of freedom at the node that is specified [ISPV(I,1)=global node of the Ith boundary condition, ISPV(I,2)=degree of freedom specified at the global node, ISPV(I,1)]

Computer implementation

Solving Equations and Postprocessing

Subroutine **EQNSOLVR** is used to solve a banded system of equations, and the solution is returned in array **GLF**

- The program performs the **Gaussian elimination** and back-substitution to compute the solution

Postprocessing involves computation of the solution and its gradient at **preselected points** of the domain

- The preselected points are the end points of the element, the midpoint, and three evenly spaced points between the end points and the midpoint
- Subroutine **POSTPROC** is used to evaluate the solution and its derivatives at the preselected points of an element

Computer implementation

NOTE :

The values computed using the derivatives of the solution are often inaccurate because the derivatives of the approximate solution become increasingly inaccurate with increasing order of differentiation.

e.g., the shear force computed in the Euler-Bernoulli beam theory (EBT)

$$V = -\frac{d}{dx}\left(b\frac{d^2w}{dx^2}\right) = \sum_{j=1}^{n} u_j^e \frac{d}{dx}\left(b\frac{d^2\phi_j^e}{dx^2}\right)$$

will be in considerable error in addition to being discontinuous across the elements. The accuracy increases, rather slowly, with mesh refinement and higher-order elements. The derivatives computed using are more accurate if they are computed at the Gauss points

Computer implementation

Applications of program FEMID

The computer program **FEM1D**, which embodies the ideas presented in the previous section is intended to illustrate the use of the finite element models to a variety of one-dimensional field problems.The program FEM1D is developed as a **learning computational tool** for a first course on the finite element method

- In the interest of simplicity and ease of understanding, only the model equations discussed and their immediate extensions are included in the program

Computer implementation

A summary of the definitions of coefficients of various model problems and their corresponding program variables

	Field problem	MODEL	NTYPE	ITEM†	AX0	AX1	BX0	BX1	CX0	CX1	FX0	FX1	FX2	CT0‡	CT1‡
1.	Plane wall	1	0	1	k_0	k_1	0.0	0.0	0.0	0.0	f_0	f_1	f_2	ρ_0	ρ_1
2.	Heat Exchanger fin	1	0	1	$(kA)_0$	$(kA)_1$	0.0	0.0	c_0	c_1	f_0	f_1	f_2	ρ_0	ρ_1
3.	Radially symmetric heat transfer	1	0	1	0.0	k_1	0.0	0.0	0.0	0.0	0.0	f_1	f_2	0.0	ρ_1
4.	Viscous flow through channels	1	0	1	μ_0	μ_1	0.0	0.0	0.0	0.0	f_0	f_1	f_2	ρ_0	ρ_1
5.	Viscous flow through pipes	1	0	1	0.0	μ_1	0.0	0.0	0.0	0.0	0.0	f_1	f_2	0.0	ρ_1
6.	Unidirectional seepage	1	0	1	μ_0	μ_1	0.0	0.0	0.0	0.0	f_0	f_1	f_2	ρ_0	ρ_1
7.	Radially symmetric seepage (ground water flow)	1	0	1	0.0	μ_1	0.0	0.0	0.0	0.0	0.0	f_1	f_2	0.0	ρ_1
8.	Axial deformation of a bar	1	0	2	$(AE)_0$	$(AE)_1$	0.0	0.0	c_0	c_1	f_0	f_1	f_2	$(\rho A)_0$	$(\rho A)_1$
9.	Radially symmetric deformation of a disk (plane stress)	1	1	2	E_1	E_2	v_{12}	H	c_0	c_1	f_0	f_1	f_2	$(\rho A)_0$	$(\rho A)_1$
10.	Radially symmetric deformation of a cylinder (plane strain)	1	2	2	E_1	E_2	v_{12}	H	c_0	c_2	f_0	f_1	f_2	$(\rho A)_0$	$(\rho A)_1$
11.	Euler–Bernoulli beam theory	3	0	2	0.0	0.0	$(EI)_0$	$(EI)_1$	c_0	c_1	f_0	f_1	f_2	ρA	ρ_I

Computer implementation

	Field problem	MODEL	NTYPE	ITEM†	AX0	AX1	BX0	BX1	CX0	CX1	FX0	FX1	FX2	CT0†	CT1†
12.	Euler–Bernoulli theory for circular plates	3	1	2	E_1	E_2	v_{12}	H	c_0	c_2	f_0	f_1	f_2	ρ_A	ρ_I
13.	Timoshenko beam theory (RIE)*	2	0	2	$(SK)_0$	$(SK)_1$	$(EI)_0$	$(EI)_1$	c_0	c_1	f_0	f_1	f_2	ρ_A	ρ_I
14.	Timoshenko beam theory (CIE)*	2	2	2	$(SK)_0$	$(SK)_1$	$(EI)_0$	$(EI)_1$	c_0	c_1	f_0	f_1	f_2	ρ_A	ρ_I
15.	Timoshenko theory of circular plates (RIE)	2	1	2	E_1	E_2	v_{12}	H	c_0	c_1	f_0	f_1	KG_{13}	ρ_A	ρ_I
16.	Timoshenko theory of circular plates (CIE)	2	3	2	E_1	E_2	v_{12}	H	c_0	c_1	f_0	f_1	KG_{13}	ρ_A	ρ_I
17.	Plane truss	4	0	0											
18.	The Euler–Bernoulli frame element	4	1	0	For field problems 17–19, these parameters are not read; instead, E =SE, A =SA, L =SL, and so on are read for each member of the structure, where SE=modulus E, SA=cross-sectional area A, SI=moment of inertia I, SL=length L of the member, CN= $\cos\alpha$, SN= $\sin\alpha$, etc. (see Table 7.3.2).										
19.	The Timoshenko frame element	4	2	0											

Computer implementation

Illustrative Examples

A description of the input variables to program FEM1D

• Data Card 1	
TITLE	Title of the problem being solved (80 characters)
• Data Card 2	
MODEL	Model equation being solved (see below)
NTYPE	Type of problem solved (see below)
	MODEL = 1, NTYPE = 0: A problem of MODEL EQUATION (3.2.1)
	MODEL = 1, NTYPE = 1: A circular DISK (PLANE STRESS)
	MODEL = 1, NTYPE > 1: A circular DISK (PLANE STRAIN)
	MODEL = 2, NTYPE = 0: A Timoshenko BEAM (RIE) problem
	MODEL = 2, NTYPE = 1: A Timoshenko PLATE (RIE) problem
	MODEL = 2, NTYPE = 2: A Timoshenko BEAM (CIE†) problem
	MODEL = 2, NTYPE > 2: A Timoshenko PLATE (CIE) problem
	MODEL = 3, NTYPE = 0: A Euler–Bernoulli BEAM problem
	MODEL = 3, NTYPE > 0: A Euler–Bernoulli circular plate
	MODEL = 4, NTYPE = 0: A plane TRUSS problem
	MODEL = 4, NTYPE = 1: A Euler–Bernoulli FRAME problem
	MODEL = 4, NTYPE = 2: A Timoshenko (CIE) FRAME problem
ITEM	Indicator for transient analysis
	ITEM = 0, Steady-state solution
	ITEM = 1, Transient analysis of PARABOLIC equations
	ITEM = 2, Transient analysis of HYPERBOLIC equations
	ITEM = 3, Eigenvalue analysis

Computer implementation

• **Data Card 3**

IELEM	Type of finite element IELEM = 0, Hermite cubic finite element IELEM = 1, Linear Lagrange finite element IELEM = 2, Quadratic Lagrange finite element
NEM	Number of elements in the mesh

• **Data Card 4**

ICONT	Indicator for continuity of data for the problem ICONT = 1, Data (AX,BX,CX,FX and mesh) is continuous ICONT = 0, Data is element dependent
NPRNT	Indicator for printing of element/global matrices NPRNT = 0, Not to print element or global matrices but postprocess the solution and print NPRNT = 1, Print Element 1 coefficient matrices only but postprocess the solution and print NPRNT = 2, Print Element 1 and global matrices but NOT postprocess the solution NPRNT > 2, Not to print element or global matrices and NOT postprocess the solution

Skip Cards 5–15 for TRUSS/FRAME problems (MODEL=4), and read Cards 5–15 only if MODEL $\neq$ 4. SKIP cards 5–9 if data is discontinuous (ICONT = 0).

• **Data Card 5**

DX(I)	Array of element lengths. DX(1) denotes the global coordinate of Node 1 of the mesh; DX(I) (I = 2, NEM1) denotes the length of the (I - 1)st element, where NEM1 = NEM + 1, and NEM denotes the number of elements in the mesh.

Cards 6–9 define the coefficients in the model equations. All coefficients are expressed in terms of GLOBAL coordinate x. See Table 7.2.1 for the meaning of the coefficients.

Computer implementation

- **Data Card 6**

AX0	Constant term of the coefficient [$a(x) =$] AX
AX1	Linear term of AX

- **Data Card 7**

BX0	Constant term of the coefficient [$b(x) =$] BX
BX1	Linear term of the coefficient BX

- **Data Card 8**

CX0	Constant term of the coefficient [$c(x) =$] CX
CX1	Linear term of the coefficient CX

SKIP Card 9 for eigenvalue problems (i.e., when ITEM = 3)

- **Data Card 9**

FX0	Constant term of the source [$f(x) =$] FX
FX1	Linear term of FX
FX2	Quadratic term of FX

SKIP Cards 10–15 if data is continuous (ICONT $\neq 0$). Cards 10–15 are read for each element (i.e., NEM times). All coefficients are with respect to the LOCAL coordinate $\bar{x}$.

- **Data Card 10**

NNM	Number of global nodes in the mesh

- **Data Card 11**

NOD	Connectivity of the element: NOD(N,I) = Global node number corresponding to the Ith node of Element N (I=1, NPE) where NPE denotes the Number of nodes Per Element
GLX(I)	Length of the Ith element

- **Data Card 12**

DCAX	Constant and linear terms of the coefficient AX

- **Data Card 13**

DCBX	Constant and linear terms of the coefficient BX

- **Data Card 14**

DCCX	Constant and linear terms of the coefficient CX

- **Data Card 15**

DCFX	Constant, linear and quadratic terms of FX

Computer implementation

READ Cards 16–23 only for TRUSS/FRAME problems (MODEL = 4); otherwise SKIP.

- **Data Card 16**

NNM	Number of nodes in the finite element mesh

SKIP Cards 17–19 for TRUSS problems (NTYPE = 0)

- **Data Card 17 (Read for each element)**

PR	Poisson's ratio of the material (not used in EBT)
SE	Young's modulus of the material
SL	Length of the element
SA	Cross-sectional area of the element
SI	Moment of inertia of the element
CS	Cosine of the angle of orientation of the element
SN	Sine of the angle of orientation of the element; the angle is measured clockwise from the global x axis

- **Data Card 18 (Read for each element)**

HF	Intensity of the horizontal distributed force
VF	Intensity of the transversely distributed force
PF	Point load on the element
XB	Distance from node 1, along the length of the element to the point of load application, PF
CNT	Cosine of the angle of orientation of the load PF
SNT	Sine of the angle of orientation of the load PF; the angle is measured clockwise from the element x axis.

Computer implementation

- **Data Card 19**

NOD	Connectivity of the element: NOD(N,I) = global node number corresponding to the Ith node of element N (I = 1,NPE)

READ Cards 20 and 21 only for TRUSS problems (NTYPE = 0).

- **Data Card 20 (Read for each element)**

SE	Young's modulus of the material
SL	Length of the element
SA	Cross-sectional area of the element
CS	Cosine of the angle of orientation of the element
SN	Sine of the angle of orientation of the element Angle is measured counterclockwise from x axis
HF	Intensity of the horizontal distributed force

- **Data Card 21**

NOD(N,I)	Connectivity of the element: NOD(N,I) = global node number corresponding to the Ith node of element N (I = 1,NPE)

- **Data Card 22**

NCON	Number of inclined support conditions

SKIP Card 23 if no inclined support conditions are specified (NCON=0).

- **Data Card 23 (I = 1 to NCON)**

ICON(I)	Global node number of the support
VCON(I)	Angle (in degrees) between the normal and the global x-axis

- **Data Card 24**

NSPV	Number of specified PRIMARY degrees of freedom

SKIP Card 25 if no primary variables is specified (NSPV=0).

- **Data Card 25 (I = 1 to NSPV)**

ISPV(I,1)	Node number at which the PV is specified
ISPV(I,2)	Specified local primary degree of freedom (DOF) at the node
VSPV(I)	Specified value of the primary variable (PV) (will not read for eigenvalue problems)

Computer implementation

SKIP Card 26 for eigenvalue problems (i.e., when ITEM = 3).

- **Data Card 26**

NSSV	Number of specified (nonzero) SECONDARY variables

SKIP Card 27 if no secondary variables is specified (NSSV=0); repeat Card 27 NSSV times.

- **Data Card 27 (I = 1 to NSSV)**

ISSV(I,1)	Node number at which the SV is specified
ISSV(I,2)	Specified local secondary DOF at the node
VSSV(I)	Specified value of the secondary variable (SV)

- **Data Card 28**

NNBC	Number of the Newton (mixed) boundary conditions

SKIP Card 29 if no mixed boundary condition is specified (NNBC = 0). The mixed boundary condition is assumed to be of the form:

SV+VNBC *(PV − UREF) = 0. Repeat Card 29 NNBC times.

- **Data Card 29 (I = 1 to NNBC)**

INBC(I,1)	Node number at which the mixed B.C. is specified
INBC(I,2)	Local DOF of the PV and SV at the node
VNBC(I)	Value of the coefficient of the PV in the B.C.
UREF(I)	Reference value of the PV

Computer implementation

- **Data Card 30**

NMPC	Number of multipoint constraints (solid mechanics)

SKIP Card 31 if no multipoint conditions are specified (NMPC = 0). The multipoint condition is assumed to be of the form:
VMPC(.,1)*PV1+VMPC(.,2)*PV2=VMPC(.,3). Repeat Card 31 NMPC times.

- **Data Card 31 (I = 1 to NMPC)**

IMC1(I,1)	Node number associated with PV1
IMC1(I,2)	Local DOF of PV1
IMC2(I,1)	Node number associated with PV2
IMC2(I,2)	Local DOF of PV2
VMPC(I)	Values of the coefficients of the constraint equation
VMPC(4)	Value of the force applied at the node of PV1 or PV2

Skip Card 32 if ITEM = 0 (read only for time-dependent or eigenvalue problems).

- **Data Card 32**

CT0	Constant part of CT = CT0 + CT1*X
CT1	Linear part of CT = CT0 + CT1*X

Skip remaining cards if steady-state or eigenvalue analysis is to be performed (ITEM = 0 or ITEM = 3).

Computer implementation

- **Data Card 33**

DT	Time increment (uniform)
ALFA	Parameter in the time approximation scheme
GAMA	Parameter in the time approximation scheme*
	GAMA (not used when ITEM = 1: parabolic equation).

Give GAMA= 10^{-6} when centered difference is used (formulation in Problem 6.23 is the correct way to implement the centered difference scheme).

- **Data Card 34**

INCOND	Indicator for initial conditions
	INCOND = 0, Homogeneous (zero) initial conditions
	INCOND > 0, Nonhomogeneous initial conditions
NTIME	Number of time steps for which solution is sought
INTVL	Time step intervals at which solution is to be printed

Skip Cards 35 and 36 if initial conditions are zero (INCOND = 0).

- **Data Card 35**

GUO	Array of initial values of the primary variables

Skip Card 36 for parabolic equations (ITEM = 1).

- **Data Card 36**

GUI	Array of initial values of the first time derivatives of the primary variables.

Computer implementation

- Variables of each "data card" are read from the same line

if the values are not found on the same line, the computer will look for them on the next line

- Data required by different data cards cannot be put on single line

each data card must start with a new line

- The space available after typing required data on a given line may be used to include any comments

e.g., we may list the variable names on that line for read reference but only after all of the required data are listed. The text included thereafter is not read by the computer (except to echo the input file)

Example 1-Steady Heat Transfer in a Rod

Consider the heat transfer problem. The problem is governed by

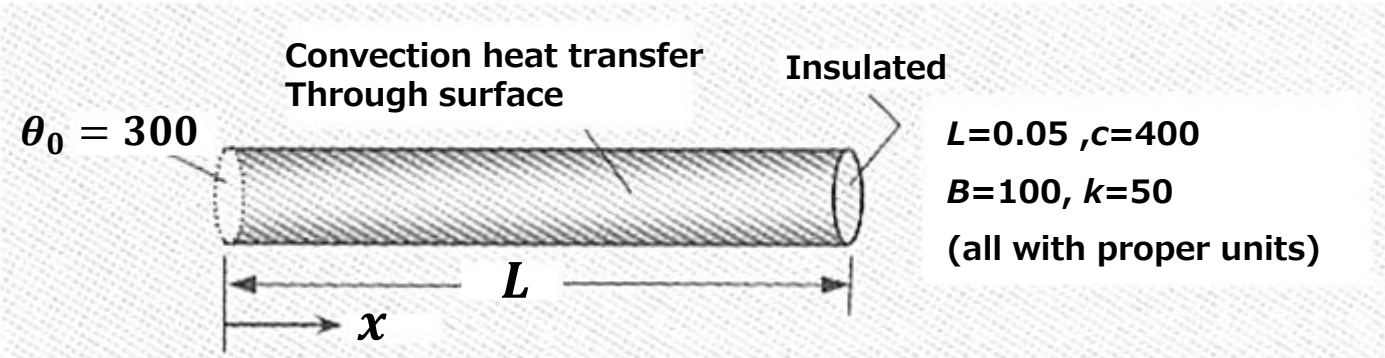
$$-\frac{d^2\theta}{dx^2} + m^2\theta = 0 \quad for \quad 0 < x < L$$

$$\theta(0) = \theta_0 , \quad (\frac{d\theta}{dx})|_{x=L} = 0$$

where θ is the nondimensional temperature

$$L = 0.05\text{m}, \; m^2 = 400/\text{m}^2, \; \theta_0 = 300°\text{C}$$

$$\beta = 100\text{W}/(\text{m}^2 \cdot °\text{C}), \quad k = 50\text{W}/(\text{m} \cdot °\text{C})$$

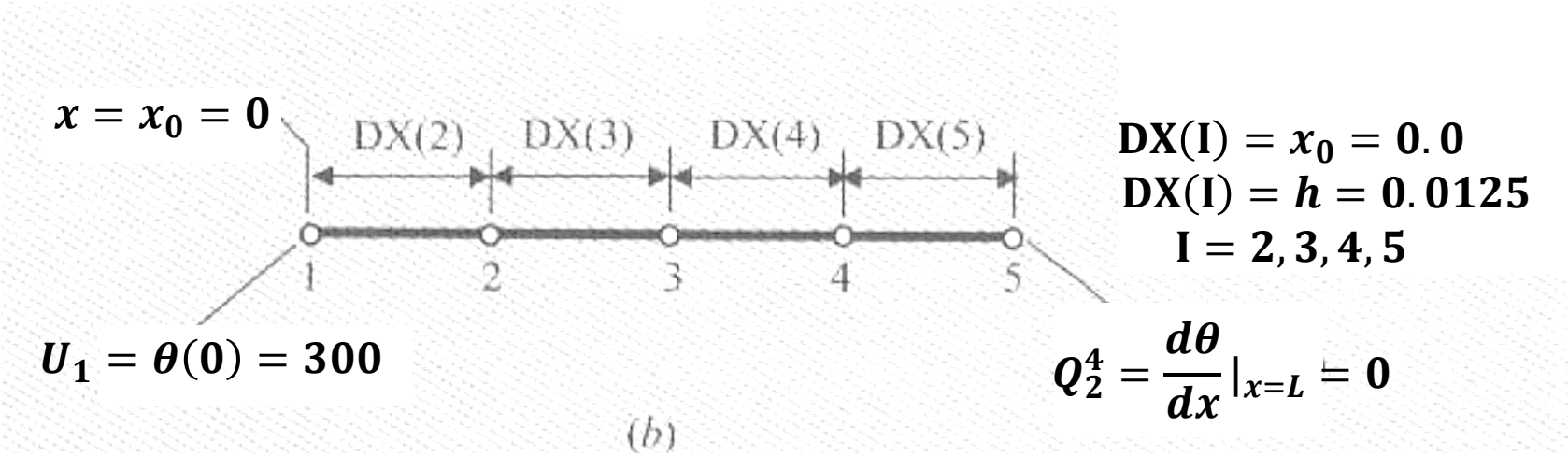


For this problem, we have

- MODEL=1, NTYPE=0 and ITEM=0 (for a steady-state solution). Since a=1.0 and c=400 are the same for all elements, we set ICONT=1, AX0=1.0, and CX0=400. All other coefficients are zero [b=0 and f=0] for this problem

For a uniform mesh of 4 linear elements (NEM=4, IELEM=1),

- The increments of the array DX(I) [where DX(I) is always the x coordinate of node I and h=L/4 =0.05/4 =0.0125] are: DX={0.0, 0.0125, 0.0125, 0.0125, 0.0125}



- The boundary conditions of the problem reduce to U1=0 and Q42=0
- There is one specified boundary condition (BC) on the primary variable (NSPV=1), and degree of free (DOF) at node 1 is: ISPV(1,1)=1 and ISPV(1,2)=1
- The specified value is VSPV(1)=300. Since the natural boundary condition（Q42=0） is homogenous, there is no need to add a zero to the corresponding entry of the source vector （i.e. NSSV=0）
- There are no mixed （i.e, convection）boundary conditions in this problem（NNBC=0）

The complete input data required to analyze the problem using FEM1D are presented

```
Example 7.3.1a: Heat transfer in a rod (4 linear elements)
  1 0 0                                          MODEL, NTYPE, ITEM
  1 4                                            IELEM, NEM
  1 1                                            ICONT, NPRNT
  0.0  0.0125   0.0125   0.0125   0.0125         DX(1)=X0; DX(2), etc. Element lengths
  1.0  0.0                                       AX0, AX1
  0.0  0.0                                       BX0, BX1
400.0  0.0                                       CX0, CX1
  0.0  0.0  0.0                                  FX0, FX1,  FX2
  1                                              NSPV
  1  1   300.0                                   ISPV(1, 1), ISPV(1,2), VSPV(1)
  0                                              NSSV
  0                                              NNBC
  0                                              NMPC
```

OUTPUT from program FEM1D by J. N. REDDY

Example 7.3.1a: Heat transfer in a rod (4 linear elements)

*** ANALYSIS OF MODEL 1, AND TYPE 0 PROBLEM ***
(see the code below)
MODEL=1, NTYPE=0: A problem described by MODEL EQ. 1
MODEL=1, NTYPE=1: A circular DISK (PLANE STRESS)
MODEL=1, NTYPE>1: A circular DISK (PLANE STRAIN)
MODEL=2, NTYPE=0: A Timoshenko BEAM (RIE) problem
MODEL=2, NTYPE=1: A Timoshenko PLATE (RIE) problem
MODEL=2, NTYPE=2: A Timoshenko BEAM (CIE) problem
MODEL=2, NTYPE>2: A Timoshenko PLATE (CIE) problem
MODEL=3, NTYPE=0: A Euler-Bernoulli BEAM problem
MODEL=3, NTYPE>0: A Euler-Bernoulli Circular plate
MODEL=4, NTYPE=0: A plane TRUSS problem
MODEL=4, NTYPE=1: A Euler-Bernoulli FRAME problem
MODEL=4, NTYPE=2: A Timoshenko (CIE) FRAME problem

Element type (0, Hermite,>0, Lagrange) ..= 1
No. of deg. of freedom per node, NDF ..= 1
No. of elements in the mesh, NEM .....= 4
No. of total DOF in the model, NEQ ..... = 5
No. of specified primary DOF, NSPV .....= 1
No. of specified secondary DOF, NSSV....= 0
No. of specified Newton B. C.: NNBC .....= 0

Boundary information on primary variables:

1 1 0.30000E+03

Global coordinates of the nodes, {GLX}:

0.00000E+00 0.12500E-01 0.25000E-01 0.37500E-01 0.50000E-01

Coefficients of the differential equation:

AX0 = 0.1000E+01 AX1 = 0.0000E+00
BX0 = 0.0000E+00 BX1 = 0.0000E+00
CX0 = 0.4000E+03 CX1 = 0.0000E+00
FX0 = 0.6000E+01 FX1 = 0.0000E+00 FX2 = 0.0000E+00

Element coefficient matrix, [ELK]:

0.81667E+02 –0.79167E+02
–0.79167E+02 0.81667E+02

Element source vector, {ELF}:

0.00000E+00 0.00000E+00

SOLUTION (values of PVs) at the NODES:

0.30000E+03 0.25152E+03 0.21892E+03 0.20016E+03 0.19403E+03

X	P. Variable	S. Variable
0.00000E+00	0.30000E+03	-0.38785E+04
0.12500E-01	0.25152E+03	-0.26076E+04
0.25000E-01	0.21893E+03	-0.15015E+04
0.37500E-01	0.20016E+03	-0.49018E+03

Analytical solution:

Output for the same problem when a mesh of two quadratic elements are used

OUTPUT from program FEM1D by J. N. REDDY

Example 7.3.1: Heat transfer in a rod (2 quadratic elements)

Element type (0, Hermite, > 0, Lagrange) ..= 2
No. of deg. of freedom per node, NDF ....= 1
No. of elements in the mesh, NEM .....= 2
No. of total DOF in the model, NEQ .....= 5
No. of specified primary DOF, NSPV .....= 1
No. of specified secondary DOF, NSSV....= 0
No. of specified Newton B. C.: NNBC .....= 0

Global coordinates of the nodes, {GLX}:
0.00000 E+00 0.12500E-01 0.25000E-01 0.37500E-01 0.50000E-01

Element coefficient matrix, [ELK]:

0.94667E+02	-0.10600E+03	0.13000E + 02
-0.10600E+03	0.21867E+03	-0.10600E + 03
0.13000E+02	-0.10600E+03	0.94667E + 02

Element source vector, {ELF}:
0.25000E-01 0.10000E+00 0.25000E-01

SOLUTION (values of PVs) at the NODES:
0.30000E+03 0.25170E+03 0.21923E+03 0.20052E+03 0.19442E+03

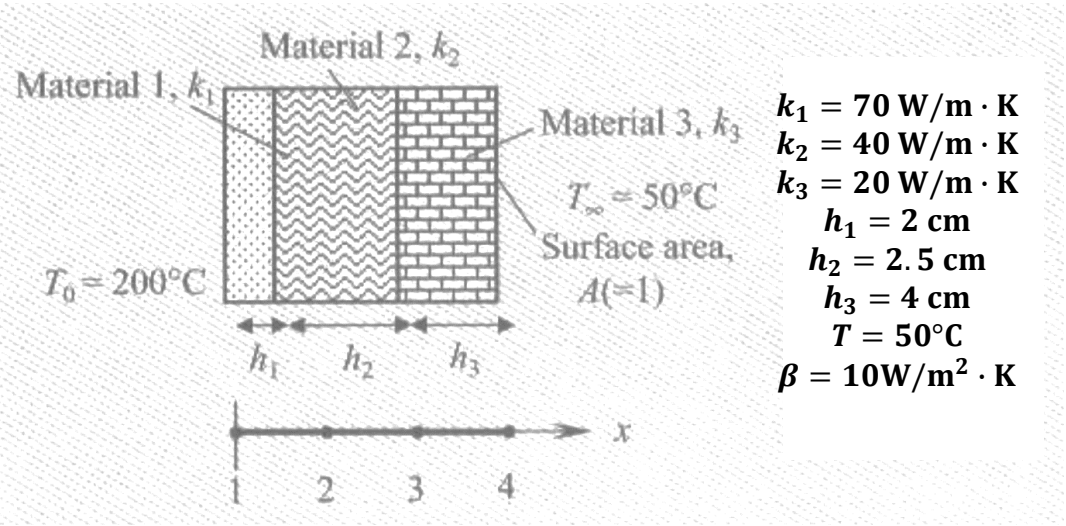
X	P.Variable	S. Variable
0.00000E+00	0.30000E+03	-0.44971E+04
0.25000E-01	0.21924E+03	-0.19642E+04
0.25000E-01	0.21924E+03	-0.20014E+04
0.50000E-01	0.19443E+03	0.16471E+02

Example 2(Steady Heat Transfer in a Composite Wall)

Here we consider the composite wall problem. The governing equations of the problem are

$$-\frac{d}{dx}\left(kA\frac{dT}{dx}\right)=0, \qquad 0<x<L$$

$$T(0)=T_0 \qquad \left[kA\frac{dT}{dx}+\beta A(T-T_\infty)\right]_{x=L}=0$$



$k_1 = 70\ \mathrm{W/m \cdot K}$
$k_2 = 40\ \mathrm{W/m \cdot K}$
$k_3 = 20\ \mathrm{W/m \cdot K}$
$h_1 = 2\ \mathrm{cm}$
$h_2 = 2.5\ \mathrm{cm}$
$h_3 = 4\ \mathrm{cm}$
$T = 50°\mathrm{C}$
$\beta = 10\mathrm{W/m^2 \cdot K}$

This is a problem with discontinuous data (ICONT=0) because a=kA is discontinuous (as well as h1/=h2/=h3). Consider a 3-element (nonuniform) mesh of linear elements

The input data and edited output are given:

```
1  0  0                    MODEL, NTYPE, ITEM
1  3                      IELEM, NEM
0  0                      ICONT, NPRNT
4                         NNM
1     2     0.02          NOD(1,J), GLX(1)
70.0  0.0                 AX0, AX1
0.0   0.0                 BX0, BX1
0.0   0.0                 CX0, CX1
0.0   0.0   0.0           FX0, FX1, FX2

 2     3     0.025        NOD(2,J) , GLX(2)
40.0  0.0                 AX0, AX1
0.0   0.0                 BX0, BX1
0.0   0.0                 CX0, CX1
0.0   0.0   0.0           FX0, FX1, FX2

3     4     0.04         NOD(2,J), G  LX(3)
20.0  0.0                 AX 0, AX1
0.0   0.0                 BX0, BX1
0.0   0.0                 CX0, CX1
0.0   0.0   0.0           FX0, FX1, FX2
1                         NSPV
1  1  200.0               ISPV(1,1), ISPV(1,2), VSPV(1)
0                         NSSV
1                         NNBC
4  1  10.0  50.0          INBC(1,1), INBC(1, 2), VNBC(1), UREF(1)
0                         NMPC
```

Data for Element 1

Data for Element 2

Data for Element 3

```
OUTPUT from program   FEM1D   by J. N. REDDY

SOLUTION (values of PVs) at the NODES:
0.20000E+03   0.19958E+03   0.19867E+03   0.19576E+03
```

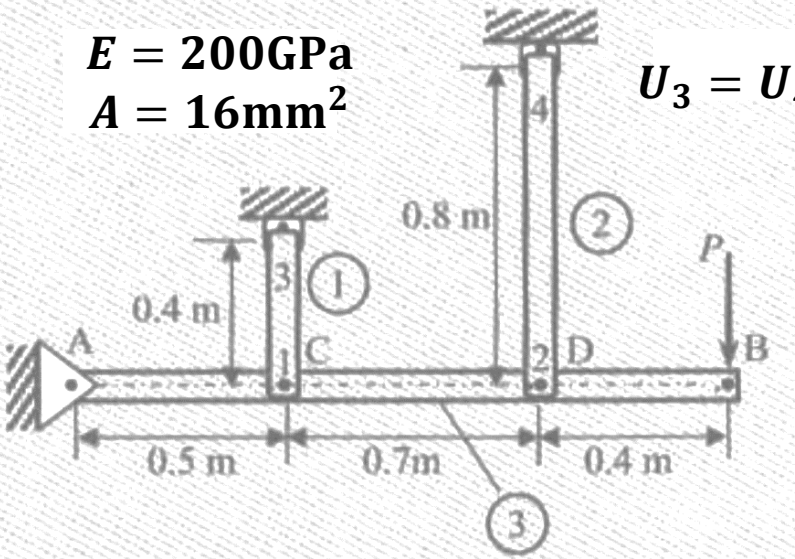
Analytical solution:

Example 3（Multipoint Constraint Problem）

A rigid bar AB of length L=1.6 m is hinged to a support at A and supported by two vertical bars attached at points C and D. Both bars have the same cross-sectional area (A=16mm^2) and are made of the same material with modulus E=200GPa. We wish to determine the elongations as well as tensile stresses in the bars

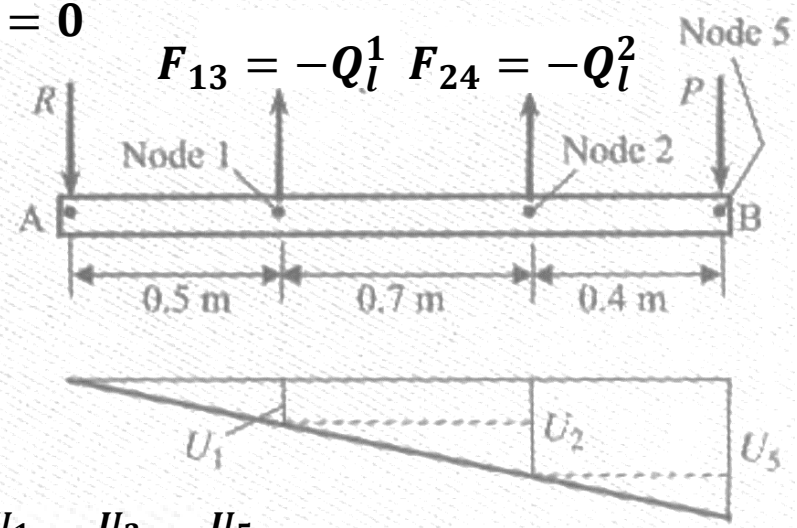
$E = 200\text{GPa}$

$A = 16\text{mm}^2$



$U_3 = U_4 = 0$

$F_{13} = -Q_l^1 \quad F_{24} = -Q_l^2$



Geometric constrains: $\frac{U_1}{0.5} = \frac{U_2}{1.2} = \frac{U_5}{1.6}$

The two geometric constraints are :

$$\frac{U_1}{0.5} = \frac{U_5}{1.6} \rightarrow 3.2U_1 - U_5 = 0, \quad \frac{U_2}{1.2} = \frac{U_5}{1.6} \rightarrow 1.333U_2 - U_5 = 0$$

Thus, we have

$$\beta_1^1 = 3.2, \beta_2^1 = -1.0, \beta_{12}^1 = 0.0; \quad \beta_2^1 = 1.333, \beta_2^2 = -1.0, \beta_{12}^2 = 0.0$$

- NEM=3 and NNM=5 with a=EA=3.2X10^6m^2 for the bar elements and a=0 for the third rigid element
- It is sufficient to use linear finite elements to represent the bars. Thus, the data for the problem are MODEL=1, NTYPE=1, ITEM=0, ICONT=0
- The boundary and constraint information are: NSPV=2, ISPV(1,1)=1, ISPV(1,2)=1, VSPV(1)=0, ISPV(2, 1))=0, ISPV(2, 2)=1, VSPV(2)=0, NSSV=0, NNBC=0, NMPC=2, VMPC(1,1)=β11=3.2, VMPC(2,1)=-1.0, VMPC(1,3)=0, VMPC(1,4)=0, VMPC(2,1)=1.33, …

```
1   0   0                   MODEL, NTYPE, ITEM
1   3                       IELEM, NEM
0   1                       ICONT, NPRNT
5                           NNM

1           3       0.4     NOD(1,J),GLX(1)
3.2E6       0.0             AX0, AX1            Data for
0.0         0.0             BX0, BX1            Element 1
0.0         0.0             CX0, CX1
0.0         0.0     0.0     FX0, FX1, FX2

2           4       0.8     NOD(2,J),GLX(2)
3.2E6       0.0             AX0, AX1            Data for
0.0         0.0             BX0, BX1            Element 2
0.0         0.0             CX0, CX1
0.0         0.0     0.0     FX0, FX1, FX2

1           5       1.6     NOD(3,J),GLX(3)
0.0         0.0             AX0, AX1            Data for
0.0         0.0             BX0, BX1            Element 3
0.0         0.0             CX0, CX1
0.0         0.0     0.0     FX0, FX1, FX2
```

```
2                           NS PV
3 1         0.0             ISPV(1,1), ISPV(1,2), VSPV(1)
4 1         0.0             ISpV(2,1), ISpV(2,2), VSpV(2)
0                           NS SV
0                           NNBC
2                           NMPC
1 1         5 1     3.2     -1.0 0.0 0.0
2 1         5 1     1.33333 -1.0 0.0 970.0
                    IMC1(1,1),IMC1(1,2),IMC2(1,1),IMC2(1,2),
                                        (VMPC(1,I),I=1 to 4)
```

Solutions

Displacements:

U1=0.1 mm, U2=0.24 mm, and U5=0.32 mm

Stresses:

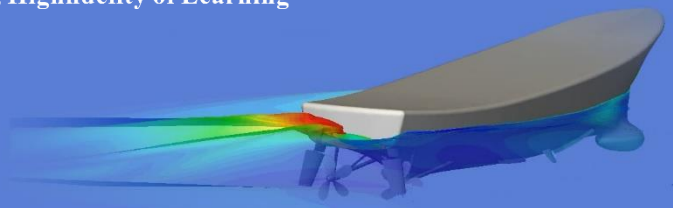
σ1=10^6(800/16)=50MPa

σ2=10^6(960/16)=60MPa

谢 谢！

CMHL SJTU COMPUTATIONAL MARINE HYDRODYNAMICS LAB
上海交大船舶与海洋工程计算水动力学研究中心

创新创智·求真求实
Creation of Mind, Highfidelity of Learning



http://dcwan.sjtu.edu.cn