University Course

MAE 207
Methods of Computer Modeling in Engineering and the Sciences

University of California, Irvine (UCI)

Spring 2006

My Class Notes

Nasser M. Abbasi

Spring 2006

Contents

Chapter 1

Introduction

Local contents

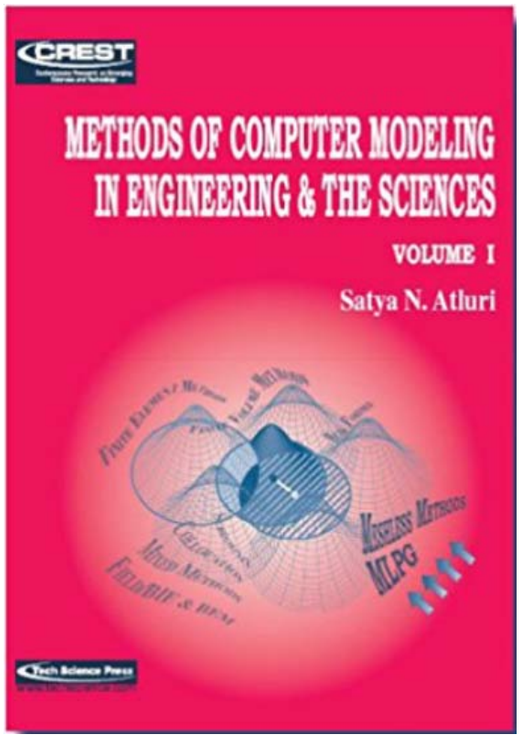
This is a course given by Professor S.N. Alturi, UCI. Part of MSc. in Mechanical engineering.

1.1 Schedule

EngrMAE 207 COMPUTATIONAL MTHDS

Code	Typ	Sec	Unt	Instructor	Time	Place	Max	Enr	WL	Req	Nor	Rstr	Fad	Web	Status
16445	Lec	A	3	ATLURI, S.N.	MW 4:00- 5:50p	CS 259	20	9	n/a	11	0	K			OPEN

1.2 Text book



http://www.techscience.com/books/mcmes_vol1.html

1.3 Links

http://care.eng.uci.edu/index.htm Professor Atluri CARE web site at UCI.

http://ocw.mit.edu/NR/rdonlyres/Aeronautics-and-Astronautics/16-20Structural-MechanicsFall2002/088DEC8D-4ACB-4E3E-9911-F3BB5EB6AF89/0/unit10.pdf This PDF file contains solution to TORSION problem in axial beam generating POISSON PDE.

http://opensees.berkeley.edu/ web site for OPENSEES FEM program.

http://opensees.berkeley.edu/OpenSees/manuals/usermanual/index.html user manual for opensees from UC Berkeley web site.

http://www.bem.uni-stuttgart.de/home.htm good notes on BEM method.

http://ocw.mit.edu/OcwWeb/Aeronautics-and-Astronautics/16-901Spring-2005/CourseHome/index.htm MIT open course for computational methods.

http://ohio.ikp.liu.se/fe/appl.html List of FEM books.

http://www.colorado.edu/engineering/CAS/courses.d/IFEM.d/Home.html on-line course material on FEM.

http://www.ann.jussieu.fr/free.htm links to free numerical software.

1.4 Other support material

Few pages scanned from book on FEM about pressure smoothing. Book is `The FEM` by Thomas Hughes.

Some Historical Remarks on Mixed and Reduced and Selective Integration Methods

Mixed finite element formulations were first discussed by Fraeijs de Veubeke [42] and Herrmann [43]. Herrmann developed a reduced form of Reissner's variational principle particularly suited to problems of incompressible and nearly incompressible elasticity and, based upon this principle, established the first effective finite elements for such cases. This is the formulation given in Sec. 4.3. Prior to this development many displacement models were applied to these problems, and poor behavior was typically observed. The reasons for this were not understood at the time. Certain elements derived from Herrmann's formulation also failed. Hughes and Allik [39] traced this failure to a correspondence between mixed and displacement models, contained within Fraeijs de Veubeke's ***limitation principle*** [42].

The first example of a uniform reduced integration element was apparently the plate-shell element presented by Zienkiewicz et al. [44]. This element, among others, is discussed in Chapter 5. The same concept was employed in other areas by Zienkiewicz and colleagues. In particular, Naylor [45] and Zienkiewicz and Godbole [46] advocated the use of the eight-node serendipity element in problems involving incompressibility. The procedure, however, was viewed by many as more a "trick" than a method and some bad experiences were subsequently noted for the serendipity element.

The concept of selective integration was first employed by Doherty et al. [47] to obtain improved bending behavior in simple four-node elasticity elements. One-point Gauss quadrature was used on the shear-strain term, and 2×2 Gauss quadrature was used to integrate the remaining terms. Although improved behavior was noted in some configurations, lack of invariance opened the approach to criticism.

Studies performed by Fried [38], Nagtegaal et al. [40], and Argyris et al. [48] provided fresh insights into why the displacement approach failed in constrained problems. Malkus [49, 50] proved the equivalence of a class of mixed models with reduced selective integration single-field elements in linear elasticity theory. The equivalence results of Malkus and Hughes [36] elevated the reduced and selective integration approaches from the realm of tricks to a legitimate methodology. Considerable research on the behavior of mixed and reduced and selective integration elements has taken place in recent years. A summary of more recent developments is contained in the following sections.

4.4.1 Pressure Smoothing

The pressure field in the reduced and selective integration penalty function formulation is to be viewed as discontinuous from element to element. In fact, all displacement derivatives for C^0 isoparametric elements are, in general, discontinuous across element boundaries. Thus, for plotting purposes, it is desirable to employ a smoothing procedure, which redefines the field under consideration in terms of the displacement shape functions N_A.

With specific reference to the pressure, there is at least one other reason for employing a smoothing procedure. It was mentioned earlier that, in certain situations,

discontinuous-pressure, mixed-method finite elements exhibit a rank-deficiency in the assembled pressure equations. By the equivalence theorem, "problems" are also to be expected with the pressure field of the penalty function formulation. These problems typically manifest themselves as pressure oscillations. For example, if four-node, quadrilateral elements are employed in a square mesh, with an even number of square elements in each direction, subjected to all velocity boundary conditions, then a checkerboard pressure oscillation is produced. Despite the pressure oscillations, the velocity field remains good.

Fortunately, smoothing procedures of a ***least squares*** type [51] seem to perform the necessary filtering as a byproduct. A comprehensive study of such techniques has been performed by Lee et al. [52]. The methods we prefer for constant-pressure elements [53], which involve slight modifications of schemes proposed in [52], are described next.

Let the discontinuous pressure field be written as

$$p^h = \sum_{e=1}^{n_{el}} \psi^e p^e \tag{4.4.25}$$

where p^e is the element mean pressure and ψ^e is the eth element "characteristic function," i.e.,

$$\psi^e(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \in \Omega^e \\ 0 & \text{if } \boldsymbol{x} \notin \Omega^e \end{cases} \tag{4.4.26}$$

The smoothed pressure is written

$$\widetilde{p} = \sum_{A=1}^{n_{np}} N_A \widetilde{p}_A \tag{4.4.27}$$

The standard least squares procedure gives rise to the following matrix problem:[7]

$$\boldsymbol{Y}\widetilde{\boldsymbol{p}} = \boldsymbol{P} \tag{4.4.28}$$

where

$$\boldsymbol{Y} = [Y_{AB}] \tag{4.4.29}$$

$$\widetilde{\boldsymbol{p}} = \{\widetilde{p}_B\} \tag{4.4.30}$$

and

$$\boldsymbol{P} = \{P_A\} \tag{4.4.31}$$

[7] The least squares procedure defines $\widetilde{p}$ by minimizing

$$\int_\Omega (\widetilde{p} - p^h)^2 \, d\Omega$$

with respect to the $\widetilde{p}_A$'s. The resulting equations emanate from

$$\frac{\partial}{\partial \widetilde{p}_A} \int_\Omega (\widetilde{p} - p^h)^2 \, d\Omega = 0$$

for $A = 1, 2, \ldots, n_{np}$.

The indices A, B take on the values $1, 2, \ldots, n_{np}$. The construction of $\boldsymbol{Y}$ and $\boldsymbol{P}$ is performed in the usual element-by-element fashion, viz.[8]

$$\boldsymbol{Y} = \mathop{\mathbf{A}}_{e=1}^{n_{el}} (\boldsymbol{y}^e), \qquad \boldsymbol{P} = \mathop{\mathbf{A}}_{e=1}^{n_{el}} (\boldsymbol{p}^e) \tag{4.4.32}$$

in which

$$\boldsymbol{y}^e = [y^e_{ab}], \qquad \boldsymbol{p}^e = \{p^e_a\}, \qquad 1 \le a, b \le n_{en} \tag{4.4.33}$$

$$y^e_{ab} = \int_{\Omega^e} N^e_a N^e_b \, d\Omega, \qquad p^e_a = p^e \int_{\Omega^e} N^e_a \, d\Omega \tag{4.4.34}$$

As it stands, the matrix $\boldsymbol{Y}$ is symmetric and positive-definite and possesses a band-profile structure. Additional simplification may be engendered by replacing $\boldsymbol{Y}$ by an associated diagonal matrix.[9] This is done by approximating the first of (4.4.34); effective procedures are summarized as follows:

$n = 2$; rectilinear case. The 2×2 product, trapezoidal integration rule may be used to diagonalize $\boldsymbol{y}^e$, i.e.,

$$y^e_{ab} = \delta_{ab} j^e(\xi_a, \eta_a) \qquad \text{(no sum on } a\text{)} \tag{4.4.35}$$

where

$$j^e = \det \begin{bmatrix} x^e_{1,\xi} & x^e_{1,\eta} \\ x^e_{2,\xi} & x^e_{2,\eta} \end{bmatrix} \qquad \text{(Jacobian determinant)} \tag{4.4.36}$$

$$\boldsymbol{x}^e = \sum_{a=1}^{n_{en}} N^e_a \boldsymbol{x}^e_a \tag{4.4.37}$$

and ξ_a and η_a are the coordinates of node a in the element "natural" coordinate system. Applying the same integration scheme to the second of (4.4.34) yields

$$p^e_a = p^e j^e(\xi_a, \eta_a) \tag{4.4.38}$$

Further simplification may be achieved by approximating $j^e(\xi_a, \eta_a)$ in (4.4.35) and (4.4.38) by $j^e(0, 0)$. (When Ω^e is a parallelogram, j^e is constant and no loss of accuracy is incurred by this procedure.)

The three-dimensional case is the straightforward generalization of the above, so we omit the details.

$n = 2$; axisymmetric case. If we attempt to apply the above procedure in the axisymmetric case, we encounter a difficulty due to the factor x_1 (i.e., r) in the integrands. Along the x_2-axis, $x_1 = 0$; hence the trapezoidal integration technique will

[8] The "assembly operators" in (4.4.32) are not the same as those used previously. Here, no boundary conditions are taken account of and there is only one degree of freedom per node.

[9] Lee et al. [52] have also found that higher accuracy is attained when $\boldsymbol{Y}$ is diagonal!

produce a zero diagonal entry in $\boldsymbol{Y}$. In this case we employ a "row-sum" diagonalization technique in which

$$y^e_{ab} = \delta_{ab} \int_{\Omega^e} N^e_a \, d\Omega \qquad \text{(no sum on } a\text{)} \tag{4.4.39}$$

The above integration, which also suffices for the second of (4.4.34), may be performed by either one-point or 2×2 Gauss-Legendre integration—the latter scheme being exact.

The procedures just described render the formation, storage, and solution of the matrix equation (4.4.28) very efficient. The results produced tend to be very good at interior nodes but leave something to be desired at boundary nodes. To improve upon the results, a "correction" at each boundary node is performed. The procedure used for four-node elements may be described with the aid of an example.

Consider the mesh illustrated in Fig. 4.4.5(a). The nodes are segregated into four groups. The boundary node corrections are carried out in the following steps in order:

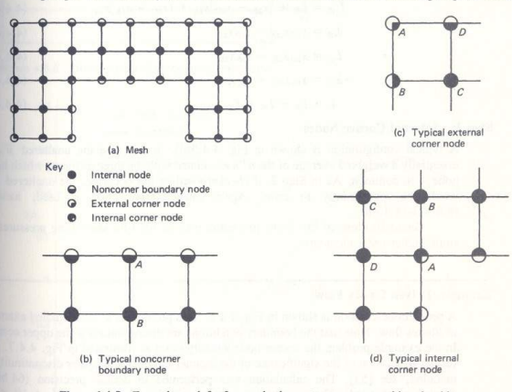


Figure 4.4.5 Example mesh for four-node element, pressure-smoothing algorithm.

Step 1: Noncorner, Boundary Nodes

A typical case of a noncorner, boundary node is depicted in Fig. 4.4.5(b). It may be observed that the unaltered value of $\widetilde{p}_A$ is actually a higher-order approximation to the

pressure at the midpoint of the line joining nodes A and B; see Barlow [54]. Thus we redefine the $\tilde{p}_A$ by way of linear extrapolation, i.e.,

$$\tilde{p}_A \longleftarrow 2\tilde{p}_A - \tilde{p}_B \tag{4.4.40}$$

Step 2: External Corner Nodes

A typical situation is depicted in Fig. 4.4.5(c). The unaltered value of $\tilde{p}_A$ is precisely the constant pressure p^e, because the above procedures reduce to "do-nothing" calculations at external corners. (If checkerboarding was occuring in the p^e's, the value of $\tilde{p}_A$ would be grossly in error.) In this case linear extrapolation is employed through nodes B, C, and D, i.e.,

$$\tilde{p}_A \longleftarrow \frac{\tilde{L}_B\tilde{p}_B + \tilde{L}_C\tilde{p}_C + \tilde{L}_D\tilde{p}_D}{L} \tag{4.4.41}$$

where

$$\tilde{L}_B = L_B + (x_{2C} - x_{2D})x_{1A} + (x_{1D} - x_{1C})x_{2A} \tag{4.4.42}$$

$$\tilde{L}_C = L_C + (x_{2D} - x_{2B})x_{1A} + (x_{1B} - x_{1D})x_{2A} \tag{4.4.43}$$

$$\tilde{L}_D = L_D + (x_{2B} - x_{2C})x_{1A} + (x_{1C} - x_{1B})x_{2A} \tag{4.4.44}$$

$$L_B = x_{1C}x_{2D} - x_{1D}x_{2C} \tag{4.4.45}$$

$$L_C = x_{1D}x_{2B} - x_{1B}x_{2D} \tag{4.4.46}$$

$$L_D = x_{1B}x_{2C} - x_{1C}x_{2B} \tag{4.4.47}$$

$$L = L_B + L_C + L_D \tag{4.4.48}$$

Step 3: Internal Corner Nodes

A typical configuration is shown in Fig. 4.4.5(d). In this case the unaltered $\tilde{p}_A$ is essentially a weighted average of the p^e's associated with the three elements which have node A in common. As in Step 2, if checkerboarding has occurred, the unaltered $\tilde{p}_A$ would be significantly in error. Again linear extrapolation is used; namely (4.4.41)–(4.4.48).

Generalizations of the above procedure may be used for smoothing pressures in some higher-order elements.

Example (Driven Cavity Flow)

A problem description is shown in Fig. 4.4.6. This problem is a much studied example of Stokes flow. Note that the boundary conditions are discontinuous at the upper corner. In the example problem the corner node velocity is set as illustrated in Fig. 4.4.7. For further discussion of the significance of the manner in which the corner discontinuity is modeled, see [53]. The calculation was performed in double precision (64 bits/floating-point word). The penalty parameter was defined by $\lambda/\mu = 10^7$. A 10×10 mesh of bilinear elements was employed with the $S1$ integration scheme. The unsmoothed pressures exhibit significant oscillations, which are removed by the method described above; see Fig. 4.4.8.

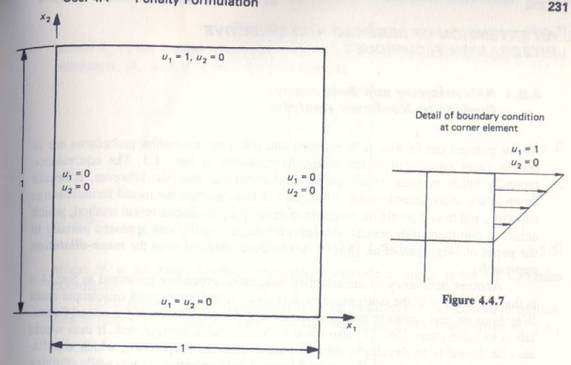


Figure 4.4.7

Figure 4.4.6 Driven cavity flow: problem description.

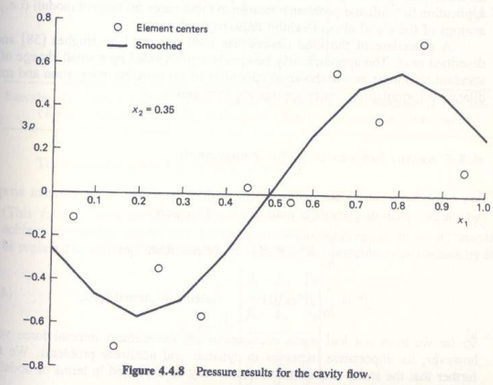


Figure 4.4.8 Pressure results for the cavity flow.

Chapter 2

Some collected class project documents

Local contents

2.1 Documents by Naseer M. Abbasi

Local contents

2.1.1 Review of FEM solution for the torsion problem of a rectangular cross section

Review of FEM solution for the torsion problem of a rectangular cross section

Nasser M. Abbasi

Nov 27,2006 Compiled on September 4, 2021 at 3:48pm

Contents

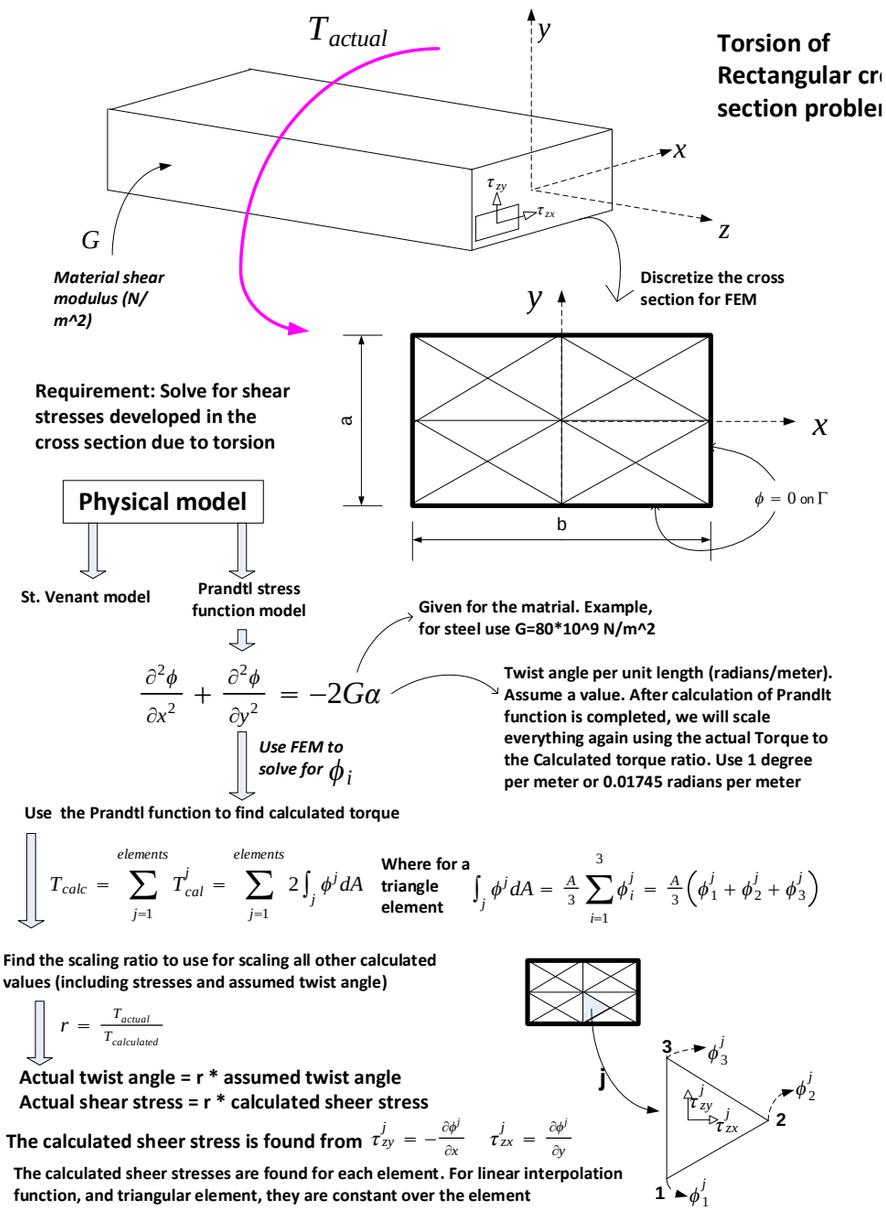
1 Introduction

This is a review of the FEM solution to the torsion problem of a rectangular cross section beam. First a description of the problem is given, then a description of the FEM method is shown, followed by a simple numerical worked example.

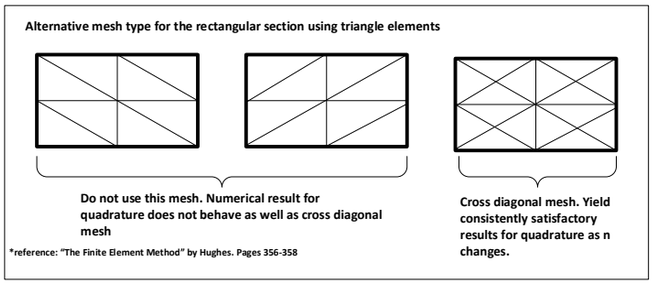
2 The problem

The problem is to solve the Poisson 2D problem for rectangular cross section. This equation is the mathematical model for a beam under torsion as described in the following diagram.

3



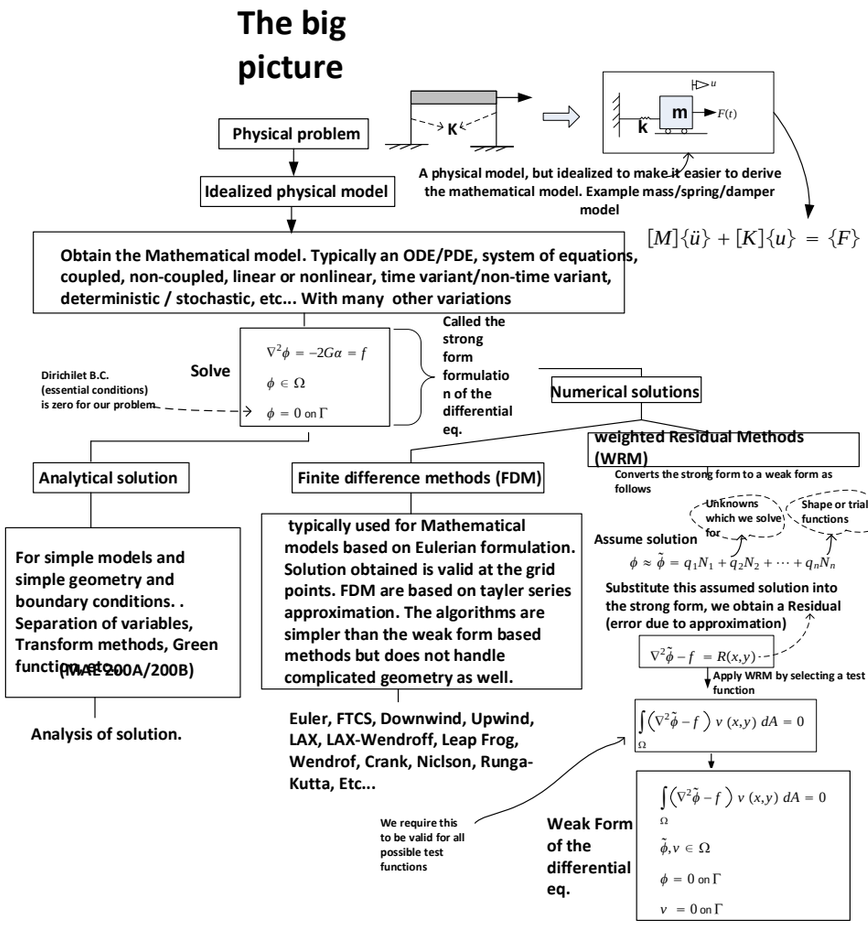
Since we are using a triangle elements for the FEM mesh, the cross section mesh is the preferred mesh to use as shown in this diagram.



3 The big picture

Before going into the details of the FEM solution it might be useful to look at the big picture.

5



The following diagram shows more description of the methods.

Weighted Residual Method Steps

1 Given strong form $\nabla^2 u - f = 0$

2. Select a test function $v = p_1 v_1 + p_2 v_2 + \cdots + p_N v_N$

These are the trial function basis (also called shape functions, or interpolation functions)

3. Find a the combination of coefficients q_i **for the trial function** $\tilde{u} = q_1 u_1 + q_2 u_2 + \cdots + q_N u_N$ **which will make**

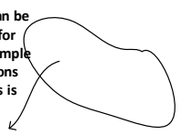
$$\int_\Omega \left(\nabla^2 \tilde{u} - f\right) v \; dA = 0$$

$$\int_\Omega \left(\nabla^2 \tilde{\phi} - f\right) v(x,y)\, dA = 0, \qquad \tilde{\phi}, v \in \Omega \;,\; \phi = 0 \text{ on } \Gamma,\; v = 0 \text{ on } \Gamma$$

Choice of Global vs. Local trial and test functions

$\tilde{u}$ **global function: defined over all of the domain** Ω

Using global trial function can be done for 1-D problems, but for higher dimension, Finding simple polynomials for basis functions over an arbitrary boundaries is hard.



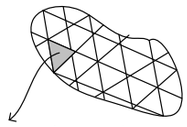
$\tilde{u}, v$ **global functions: the basis are global over all of the domain** Ω

Choice of test (weight) function v

- **Dirac delta function** → **Collocation method**
- **Heaviside step function** → **Subdomain method** → **overlapping**, **Non-overlapping**
- **Weighted integral square error method**

$\tilde{u}$ **local function: defined over an element within the discretized domain** Ω

By discretization using simple elements such as triangles, we are able to come up with simple polynomials for the basis functions and cover the whole domain at the same time



local trial function for element j **is** $\tilde{u}^j = q_1^j N_1 + q_2^j N_2 + q_3^j N_3$

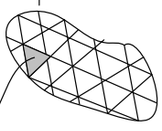
Choice of test (weight) function v

- ***V* is selected from the basis functions used to define the trial function** $\tilde{u}$ → **Galerkin method**

  Assuming we choose $\tilde{u} = q_1 N_1 + q_2 N_2 + \cdots + q_M N_M$

  Then $\int_{\Omega^j} \left(\nabla^2 \tilde{\phi} - f\right) N_i \; dA = 0 \quad i = 1,2,M$

- **V is constant over each element** → **FVM** → **See Prof. Atluri, Vol 1 book, pages 277-280 for details**
- **Other variations of test function in conjunction of using non-symmetric weak form lead to field/boundary integral equations and field/ boundary element method. See text book, chapterX**



$$\int_{\Omega^j} \left(\nabla^2 \tilde{\phi} - f\right) N_1 \; dA = 0$$
$$\vdots$$
$$\int_{\Omega^j} \left(\nabla^2 \tilde{\phi} - f\right) N_M \; dA = 0$$

Assuming we select M basis functions to approximate u with over each element, then we will obtain M equations per element as follows

To reduce the continuity requirement on u within the element, we convert the unsymmetrical weak form above to a symmetric weak from. See next page

4 Mathematical derivation

4.1 Derivation of the symmetric weak form of the 2D Poisson equation

The following diagram shows the steps to obtain the symmetric weak for of the 2D Poisson PDE

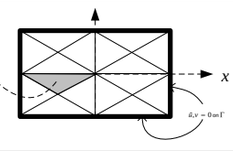
8

Process to Convert weak form to symmetric weak form using integration by parts and divergence theorem

$$\int_{\Omega_j} \left(\nabla^2 \tilde{u} - f\right) v \; dA_j = 0$$

$\tilde{u}, v \in \Omega_j$

$\tilde{u}, v = 0$ on Γ



$$\left(\int_\Omega \frac{\partial^2 \tilde{u}}{\partial x^2} v \, dA + \int_\Omega \frac{\partial^2 \tilde{u}}{\partial x^2} v \, dA - \int_\Omega f v \; dA \right) = 0$$

① Plug in here

Since $\frac{\partial}{\partial x}\left(\frac{\partial \tilde{u}}{\partial x} v\right) = \frac{\partial^2 \tilde{u}}{\partial x^2} v + \frac{\partial \tilde{u}}{\partial x}\frac{\partial v}{\partial x}$

Then $\frac{\partial^2 \tilde{u}}{\partial x^2} v = \frac{\partial}{\partial x}\left(\frac{\partial \tilde{u}}{\partial x} v\right) - \frac{\partial \tilde{u}}{\partial x}\frac{\partial v}{\partial x}$

Similarly, $\frac{\partial^2 \tilde{u}}{\partial y^2} v = \frac{\partial}{\partial y}\left(\frac{\partial \tilde{u}}{\partial y} v\right) - \frac{\partial \tilde{u}}{\partial y}\frac{\partial v}{\partial y}$

Integration by parts

②

$$\int_\Omega \left(\frac{\partial}{\partial x}\left(\frac{\partial \tilde{u}}{\partial x} v\right) - \frac{\partial \tilde{u}}{\partial x}\frac{\partial v}{\partial x} \right) dA + \int_\Omega \frac{\partial}{\partial y}\left(\frac{\partial \tilde{u}}{\partial y} v\right) - \frac{\partial \tilde{u}}{\partial y}\frac{\partial v}{\partial y} dA - \int_\Omega f v \; dA = 0$$

③ Plug in here

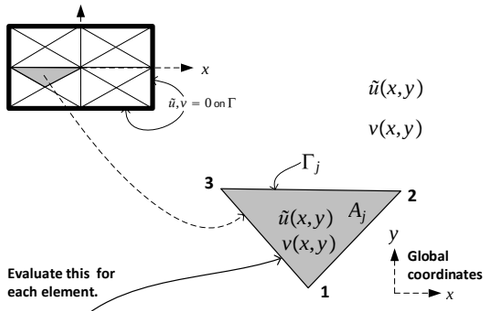
But $\int_{\Omega_j} \frac{\partial}{\partial x}\left(\frac{\partial \tilde{u}}{\partial x} v\right) dA = \int_{\Gamma_j} \left(\frac{\partial \tilde{u}}{\partial x} v\right) n_x \, d\Gamma_j$

and $\int_{\Omega_j} \frac{\partial}{\partial y}\left(\frac{\partial \tilde{u}}{\partial y} v\right) dA = \int_{\Gamma_j} \left(\frac{\partial \tilde{u}}{\partial y} v\right) n_y \, d\Gamma_j$

Divergence theorem

④

$$\int_{\Gamma_j} \left(\frac{\partial \tilde{u}}{\partial x} v\right) n_x \, d\Gamma_j - \int_{\Omega_j} \frac{\partial \tilde{u}}{\partial x}\frac{\partial v}{\partial x} dA_j + \int_{\Gamma_j} \left(\frac{\partial \tilde{u}}{\partial y} v\right) n_y \, d\Gamma_j - \int_{\Omega_j} \frac{\partial \tilde{u}}{\partial y}\frac{\partial v}{\partial y} dA_j - \int_{\Omega_j} f v \; dA_j = 0$$



Evaluate this for each element.

$$\int_{\Gamma_j} \left(\frac{\partial \tilde{u}}{\partial x} v\right) n_x + \left(\frac{\partial \tilde{u}}{\partial y} v\right) n_y \, d\Gamma_j - \int_{\Omega_j} \frac{\partial \tilde{u}}{\partial x}\frac{\partial v}{\partial x} + \frac{\partial \tilde{u}}{\partial y}\frac{\partial v}{\partial y} + f v \, dA_j = 0$$

$\tilde{u}, v$ *are* C^0 *continuous* $\in \Omega_j$ *and at* Γ_j

$\tilde{u}^+ = \tilde{u}^-$ *at the inter – element boundaries*

See Textbook page 282 for more details

9

4.2 Converting the symmetric weak form equation from the global Cartesian coordinates system to natural coordinates system

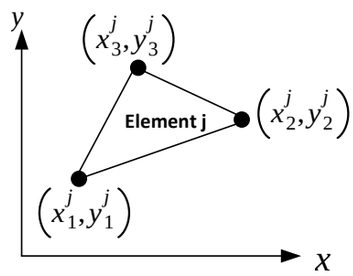
Converting the integral equations from the global Cartesian coordinates system to what is called the natural coordinates system (the local coordinates system) is a standard step used in FEM.

"A local coordinates system that relies on the element geometry for its definition and whose coordinates range between zero and unity within the element is known a natural coordinates system. Such system have the property that one particular coordinate has unit value at one node of the element and zero value at the other nodes: its variation between nodes is linear"[1]

Integration of shape functions when they are written in the natural coordinates are simplified since the origin is now located on the element. These are the main reasons for changing from global coordinates to the natural coordinates. For simple geometries, one can avoid having to do this coordinates transformation, but in general and in practice it is the standard procedure to do.

I found that most of the technical and mathematical difficulties involved are in this step. So more details will be spend on this.

The global coordinates of the element is shown in this diagram



Showing the global coordinates of nodes for element J

Given an equation or expression where the independent variables in the equation are x, y (the global Cartesian coordinates system) and we wish to express this same equation using the independent variables ζ, η, then we perform coordinates transformations.

Given that $x = x\left(\zeta, \eta\right)$ and $y = y\left(\zeta, \eta\right)$, we first find the differentials of the old coordinates system (i.e. dx, dy) in terms of the differentials of the new coordinates system $(d\zeta, d\eta)$

The matrix that represents this mapping between the differentials in the old coordinates system and the new coordinates system us called the Jacobian (some books call the determinant of this matrix as the Jacobian). It is important to note that this mapping is between the differentials of the independent variables in the two coordinates system, and not between the variables themselves.

[1] The FEM for engineers. Kenneth Huebner. 1975

10

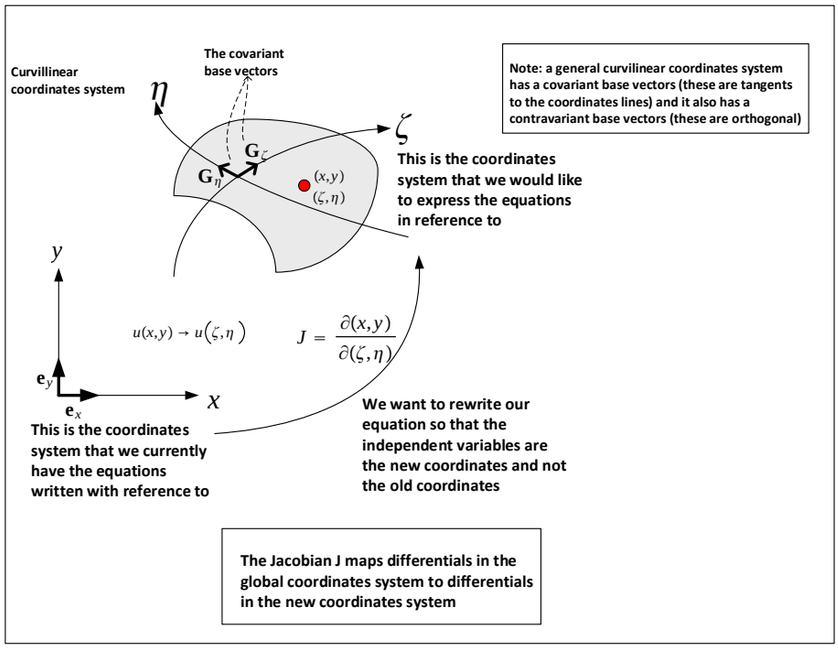
Hence we write

$$\begin{Bmatrix} dx \\ dy \end{Bmatrix} = \overbrace{\begin{bmatrix} \frac{\partial x}{\partial \zeta} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \zeta} & \frac{\partial y}{\partial \eta} \end{bmatrix}}^{J} \begin{Bmatrix} d\zeta \\ d\eta \end{Bmatrix}$$

J is also written as

$$J = \frac{\partial (x, y)}{\partial (\zeta, \eta)}$$

The main use for the Jacobian is in change of variables from one coordinates system to another, and also in performing area and volume integrals.



Hence, converting an integral from the global coordinates to the natural coordinates can be done as follows

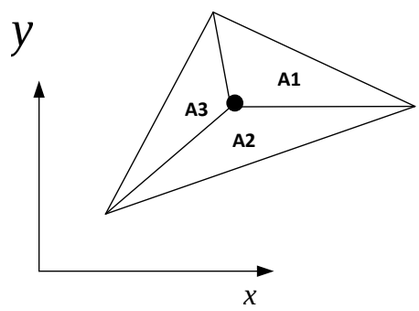
$$\int_A f(x, y) dx dy = \int_{-1}^{1} \int_{-1}^{1} g\left(\zeta, \eta\right) |J| \, d\zeta d\eta$$

When the natural coordinates are area coordinates (which is the case here), we should modify the above to become

11

$$\int_A f\left(x,y\right) dxdy = \int_0^1 \int_0^{1-L_1} g\left(L_1, L_2\right)\ |J|\ dL_2 dL_1$$

The area coordinates (L_1, L_2, L_3) are illustrated in this diagram



$$L_1 + L_2 + L_3 = 1$$

$$L_1 = \frac{A_1}{A}$$

$$L_2 = \frac{A_2}{A}$$

$$L_3 = \frac{A_3}{A}$$

Area coordinates (L1,L2,L3)

It is important to realize that the shape functions N_1, N_2, N_3 used will be the same as the area coordinates.

Let us now start from the symmetric weak form equation, with the goal to convert it to the natural coordinates (see previous diagram for the derivation of this equation)

$$\int_{\Gamma_{external\ boundary}} \left(\frac{\partial u}{\partial x} v\right) n_x + \left(\frac{\partial u}{\partial y} v\right) n_y\, d\Gamma - \int_{\Omega_j} \frac{\partial u}{\partial x}\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\frac{\partial v}{\partial y} dxdy + \int_{\Omega_j} f v dxdy = 0 \tag{1}$$

Since the first integral above is carried along the boundaries of the whole domain itself (not along the boundaries of the individual elements themselves) and since we set the value of the test function v to be zero at the boundaries of the domain, the first part of the above integral is zero. Hence the above integral become

$$-\int_{\Omega_j} \frac{\partial u}{\partial x}\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}\frac{\partial v}{\partial y} dxdy + \int_{\Omega_j} f v dxdy = 0 \tag{1}$$

In the following derivations, everything is done on an element j, hence all the u, v, and element nodes coordinates x_1, y_1, etc.. should have a superscript j on, as in u^j, v^j etc... To make things easier to read, I will not put the superscript j but will add it back at the end.

12

Consider first the second integral from (1) Which can be rewritten as

$$I_1 = -\int_{\Omega_j} \begin{pmatrix} \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{pmatrix} \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{Bmatrix} \mathbf{dxdy} + \int_{\Omega_j} f v dx dy \tag{2}$$

Consider the first integral from above

$$I = \int_{\Omega_j} \begin{pmatrix} \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{pmatrix} \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{Bmatrix} dxdy \tag{2A}$$

The above is written with reference to the global coordinates system. However, We want our trial and test functions to be defined in the natural coordinates system (where things are simpler). So we need a way to transform the above integral (2A) to the natural coordinates system.

Assume we have the mapping $x = x(\zeta, \eta)$ and $y = y(\zeta, \eta)$ (we will see how to obtain this mapping below). This mapping tells us how the global coordinates themselves change as a function of the natural coordinates. Now we can use differentiation chain rule to see how the trial and test functions themselves change relative the global coordinates.

$$\begin{aligned} \frac{\partial u}{\partial \zeta} &= \frac{\partial u}{\partial x}\frac{\partial x}{\partial \zeta} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial \zeta} \\ \frac{\partial u}{\partial \eta} &= \frac{\partial u}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial \eta} \end{aligned}$$

Similarly for the test function

$$\begin{aligned} \frac{\partial v}{\partial \zeta} &= \frac{\partial v}{\partial x}\frac{\partial x}{\partial \zeta} + \frac{\partial v}{\partial y}\frac{\partial y}{\partial \zeta} \\ \frac{\partial v}{\partial \eta} &= \frac{\partial v}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial v}{\partial y}\frac{\partial y}{\partial \eta} \end{aligned}$$

To make things more clear, we rewrite the above using matrix notation. For the trial function

$$\begin{aligned} \begin{Bmatrix} \frac{\partial u}{\partial \zeta} \\ \frac{\partial u}{\partial \eta} \end{Bmatrix} &= \begin{bmatrix} \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{Bmatrix} \\ &= [J] \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{Bmatrix} \end{aligned} \tag{3}$$

13

and similarly for the test function

$$\begin{Bmatrix} \frac{\partial v}{\partial \zeta} \\ \frac{\partial v}{\partial \eta} \end{Bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \begin{Bmatrix} \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \end{Bmatrix}$$
$$= [J] \begin{Bmatrix} \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \end{Bmatrix} \tag{4}$$

From (3) and (4), we see the following inverse transformations

$$\begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \frac{\partial u}{\partial \zeta} \\ \frac{\partial u}{\partial \eta} \end{Bmatrix} \tag{5}$$

and

$$\begin{Bmatrix} \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \frac{\partial v}{\partial \zeta} \\ \frac{\partial v}{\partial \eta} \end{Bmatrix} \tag{6}$$

Now transpose the column vector in (6) to be a row vector because that is how it is laid out in the integral (2A), (and remember to change the order when transposing a product)

$$\begin{pmatrix} \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{pmatrix} = \begin{pmatrix} \frac{\partial v}{\partial \zeta} & \frac{\partial v}{\partial \eta} \end{pmatrix} [J]^{-T}$$

Now we are ready to convert the integral I_2 in eq (2A) to the natural coordinates system (these are area coordinates, notice the integral limits and the order of integration)

$$I_2 = \int_{\Omega_j} \begin{pmatrix} \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{pmatrix} \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{Bmatrix} dxdy$$
$$= \int_0^1 \int_0^{1-\zeta} \begin{pmatrix} \frac{\partial v}{\partial \zeta} & \frac{\partial v}{\partial \eta} \end{pmatrix} [J]^{-T} [J]^{-1} \begin{Bmatrix} \frac{\partial u}{\partial \zeta} \\ \frac{\partial u}{\partial \eta} \end{Bmatrix} \det [J] \, d\eta d\zeta \tag{7}$$

Where we used the standard relationship that

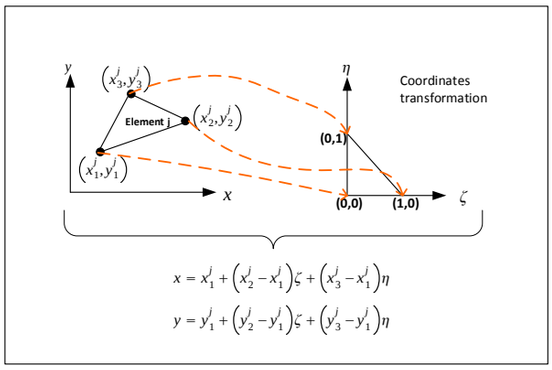
$$dxdy = \det[J]\, d\eta d\zeta$$

Remember to put $d\eta$ first before $d\zeta$ since the inner limit is on ζ.

Now that we have everything in the natural area coordinates system, we can do the integration. One small point left, which is to determine the differentials involved in (7).

For this we now need to decide on the actual form of the trial and test functions and on the mapping between the global and the natural coordinates system. The following explains this part, we will come back to the above integral once we have obtained the differentials $\frac{\partial v}{\partial \zeta}, \frac{\partial v}{\partial \eta}, \frac{\partial u}{\partial \zeta}, \frac{\partial u}{\partial \eta}$ and determined the Jacobian.

The following diagram shows the linear transformation we will use. This is a standard transformation where the natural coordinates are called the area coordinates described more below.



We see from the above diagram that

$$\begin{aligned} x &= x_1^j + \left(x_2^j - x_1^j\right)\zeta + \left(x_3^j - x_1^j\right)\eta \\ y &= y_1^j + \left(y_2^j - y_1^j\right)\zeta + \left(y_3^j - y_1^j\right)\eta \end{aligned} \tag{8}$$

From the above we obtain the following differentials

$$\begin{aligned} \frac{\partial x}{\partial \zeta} &= (x_2 - x_1) \\ \frac{\partial x}{\partial \eta} &= (x_3 - x_1) \\ \frac{\partial y}{\partial \zeta} &= (y_2 - y_1) \\ \frac{\partial y}{\partial \eta} &= (y_3 - y_1) \end{aligned}$$

15

Now, we consider the trial and test functions. based on the above transformation shown in eq (8), We see that the linear trial and test functions can also be written in similar transformation

$$\begin{aligned} u^j &= u_1^j + \left(u_2^j - u_1^j\right)\zeta + \left(u_3^j - u_1^j\right)\eta \\ v^j &= v_1^j + \left(v_2^j - v_1^j\right)\zeta + \left(v_3^j - v_1^j\right)\eta \end{aligned} \tag{9}$$

Again, immediately, we obtain the following differentials from the above expressions

$$\begin{aligned} \frac{\partial u}{\partial \zeta} &= (u_2 - u_1) \\ \frac{\partial u}{\partial \eta} &= (u_3 - u_1) \\ \frac{\partial v}{\partial \zeta} &= (v_2 - v_1) \\ \frac{\partial v}{\partial \eta} &= (v_3 - v_1) \end{aligned}$$

Hence the Jacobian can now be evaluated (see eq(3) for reference)

$$\begin{aligned} [J] &= \begin{bmatrix} \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} \\ &= \begin{bmatrix} (x_2 - x_1) & \left(y_2 - y_1\right) \\ (x_3 - x_1) & \left(y_3 - y_1\right) \end{bmatrix} \end{aligned} \tag{10}$$

And its inverse is

$$[J]^{-1} = \begin{bmatrix} \left(y_3 - y_1\right) & \left(y_1 - y_2\right) \\ (x_1 - x_3) & (x_2 - x_1) \end{bmatrix} \frac{1}{\det [J]}$$

And

$$[J]^{-T} = \begin{bmatrix} \left(y_3 - y_1\right) & (x_1 - x_3) \\ \left(y_1 - y_2\right) & (x_2 - x_1) \end{bmatrix} \frac{1}{\det [J]}$$

Now that we have all the differentials needed, we can now go back to the integral in eq (7) and compute it:

$$I_2 = \int_0^1 \int_0^{1-\zeta} \begin{pmatrix} \frac{\partial v}{\partial \zeta} & \frac{\partial v}{\partial \eta} \end{pmatrix} [J]^{-T} [J]^{-1} \begin{Bmatrix} \frac{\partial u}{\partial \zeta} \\ \frac{\partial u}{\partial \eta} \end{Bmatrix} \det[J]\, d\eta d\zeta$$

$$= \int_0^1 \int_0^{1-\zeta} \begin{bmatrix} (v_2 - v_1) & (v_3 - v_1) \end{bmatrix} [J]^{-T} [J]^{-1} \begin{Bmatrix} (u_2 - u_1) \\ (u_3 - u_1) \end{Bmatrix} \det[J]\, d\eta d\zeta$$

$$= \int_0^1 \int_0^{1-\zeta} \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix} \begin{Bmatrix} -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{Bmatrix} [J]^{-T} [J]^{-1} \begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix} \det[J]\, d\eta d\zeta$$

$$= \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix} \overbrace{\left(\int_0^1 \int_0^{1-\zeta} \begin{Bmatrix} -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{Bmatrix} [J]^{-T} [J]^{-1} \begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \det[J]\, d\eta d\zeta \right)}^{\text{This is the local stiffness matrix for element j}} \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \end{Bmatrix} \tag{11}$$

Now we can evaluate K^j.

$$K^j = \int_0^1 \int_0^{1-\zeta} \begin{Bmatrix} -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{Bmatrix} [J]^{-T} [J]^{-1} \begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \det[J]\, d\eta d\zeta$$

The integrand is

$$I_3 = \begin{Bmatrix} -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{Bmatrix} [J]^{-T} [J]^{-1} \begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \det[J]$$

$$= \begin{Bmatrix} -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{Bmatrix} \overbrace{\begin{bmatrix} (y_3 - y_1) & (x_1 - x_3) \\ (y_1 - y_2) & (x_2 - x_1) \end{bmatrix} \frac{1}{\det[J]}}^{J^{-T}} \overbrace{\begin{bmatrix} (y_3 - y_1) & (y_1 - y_2) \\ (x_1 - x_3) & (x_2 - x_1) \end{bmatrix} \frac{1}{\det[J]}}^{J^{-1}} \begin{bmatrix} -1 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \det[J]$$

$$= \frac{1}{\det[J]} \begin{bmatrix} b_1^2 + c_1^2 & b_2 b_1 + c_2 c_1 & b_3 b_1 + c_3 c_1 \\ b_2 b_1 + c_2 c_1 & b_2^2 + c_2^2 & b_3 b_2 + c_3 c_2 \\ b_3 b_1 + c_3 c_1 & b_3 b_2 + c_3 c_2 & b_3^2 + c_3^2 \end{bmatrix}$$

Where

$$b_1 = y_2 - y_3,\ b_2 = y_3 - y_1,\ b_3 = y_1 - y_2,\ c_1 = x_3 - x_2,\ c_2 = x_1 - x_3,\ c_3 = x_2 - x_1$$

Hence

$$K^j = \int_0^1 \int_0^{1-\zeta} \frac{1}{\det[J]} \begin{bmatrix} b_1^2 + c_1^2 & b_2 b_1 + c_2 c_1 & b_3 b_1 + c_3 c_1 \\ b_2 b_1 + c_2 c_1 & b_2^2 + c_2^2 & b_3 b_2 + c_3 c_2 \\ b_3 b_1 + c_3 c_1 & b_3 b_2 + c_3 c_2 & b_3^2 + c_3^2 \end{bmatrix} d\eta d\zeta$$

But the integrand is a constant, hence we take it out of the integral

$$K^j = \frac{1}{\det[J]} \begin{bmatrix} b_1^2 + c_1^2 & b_2 b_1 + c_2 c_1 & b_3 b_1 + c_3 c_1 \\ b_2 b_1 + c_2 c_1 & b_2^2 + c_2^2 & b_3 b_2 + c_3 c_2 \\ b_3 b_1 + c_3 c_1 & b_3 b_2 + c_3 c_2 & b_3^2 + c_3^2 \end{bmatrix} \int_0^1 \int_0^{1-\zeta} d\eta d\zeta$$

Now we evaluate $\int_0^1 \int_0^{1-\zeta} d\eta d\zeta$

$$\begin{aligned} \int_0^1 \int_0^{1-\zeta} d\eta d\zeta &= \int_0^1 \left(\int_0^{1-\zeta} d\eta \right) d\zeta \\ &= \int_0^1 \left(\eta \right)_0^{1-\zeta} d\zeta \\ &= \int_0^1 1 - \zeta d\zeta \\ &= \left(\zeta - \frac{\zeta^2}{2} \right)_0^1 \\ &= 1 - \frac{1}{2} \\ &= \frac{1}{2} \end{aligned}$$

Hence

$$K^j = \frac{1}{2 \det[J]} \begin{bmatrix} b_1^2 + c_1^2 & b_2 b_1 + c_2 c_1 & b_3 b_1 + c_3 c_1 \\ b_2 b_1 + c_2 c_1 & b_2^2 + c_2^2 & b_3 b_2 + c_3 c_2 \\ b_3 b_1 + c_3 c_1 & b_3 b_2 + c_3 c_2 & b_3^2 + c_3^2 \end{bmatrix}$$

But from (10) we see that

$$\det[J] = x_3 \left(y_1 - y_2 \right) + x_1 \left(y_2 - y_3 \right) + x_2 \left(y_3 - y_1 \right)$$

and the area of a triangle with corners at $\left(x_1, y_1 \right), \left(x_2, y_2 \right), \left(x_3, y_3 \right)$ is given by

$$A = \frac{1}{2}\det\begin{pmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{pmatrix} = \frac{1}{2}\left(x_1\left(y_2 - y_3\right) + x_2\left(y_3 - y_1\right) + x_3\left(y_1 - y_2\right)\right)$$

Hence we get

$$\det\left([J]^j\right) = 2A^j$$

Therefore we can replace $\det\left([J]^j\right)$ by $2A^j$ everywhere. Rewrite the local stiffness matrix in terms of the local element area:

$$K^j = \frac{1}{4A^j}\begin{bmatrix} b_1^2 + c_1^2 & b_2b_1 + c_2c_1 & b_3b_1 + c_3c_1 \\ b_2b_1 + c_2c_1 & b_2^2 + c_2^2 & b_3b_2 + c_3c_2 \\ b_3b_1 + c_3c_1 & b_3b_2 + c_3c_2 & b_3^2 + c_3^2 \end{bmatrix}$$

Now that we have K^j we plug it back into eq (11)

$$\begin{aligned} I_2 &= \begin{bmatrix} v_1^j & v_2^j & v_3^j \end{bmatrix} K^j \begin{Bmatrix} u_1^j \\ u_2^j \\ u_3^j \end{Bmatrix} \\ &= \begin{bmatrix} v_1^j & v_2^j & v_3^j \end{bmatrix} \frac{1}{4A^j}\begin{bmatrix} b_1^2 + c_1^2 & b_2b_1 + c_2c_1 & b_3b_1 + c_3c_1 \\ b_2b_1 + c_2c_1 & b_2^2 + c_2^2 & b_3b_2 + c_3c_2 \\ b_3b_1 + c_3c_1 & b_3b_2 + c_3c_2 & b_3^2 + c_3^2 \end{bmatrix} \begin{Bmatrix} u_1^j \\ u_2^j \\ u_3^j \end{Bmatrix} \end{aligned}$$

And now that we completed this integral we go back to eq (2) :

$$I_1 = -\overbrace{\int_{\Omega_j}\begin{pmatrix} \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{pmatrix}\begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \end{Bmatrix} dA}^{\text{This is the integral we just completed above}} + \int_{\Omega_j} f v dA$$

We need to work on the second integral above $\int_{\Omega_j} f v dA$ and transform it to the natural coordinates.

19

$$
\begin{aligned}
I &= \int_{\Omega_j} f^j v^j dx dy \\
&= \begin{pmatrix} v_1^j & v_2^j & v_3^j \end{pmatrix} \int_{\Omega_j} f^j dx dy \\
&= \begin{pmatrix} v_1^j & v_2^j & v_3^j \end{pmatrix} \begin{Bmatrix} Q_1^j \\ Q_2^j \\ Q_3^j \end{Bmatrix}
\end{aligned}
$$

Where

$$
\begin{aligned}
Q_1^j &= \int_0^1 \int_0^{1-\zeta} f_1^j N_1 \det[J]^j \, d\eta d\zeta \\
&= \int_0^1 \int_0^{1-\zeta} f_1^j \left(1-\zeta-\eta\right) \det[J]^j \, d\eta d\zeta \\
&= f_1^j \det[J]^j \int_0^1 \left(\int_0^{1-\zeta} \left(1-\zeta-\eta\right) d\eta \right) d\zeta \\
&= f_1^j \det[J]^j \int_0^1 \left(\eta - \zeta\eta - \frac{\eta^2}{2} \right)_0^{1-\zeta} d\zeta \\
&= f_1^j \det[J]^j \int_0^1 \left((1-\zeta) - \zeta(1-\zeta) - \frac{(1-\zeta)^2}{2} \right) d\zeta \\
&= f_1^j \det[J]^j \int_0^1 \left(1 - \zeta - \zeta + \zeta^2 - \frac{\left(1+\zeta^2-2\zeta\right)}{2} \right) d\zeta \\
&= f_1^j \frac{\det[J]^j}{2} \int_0^1 \left(2 - 4\zeta + 2\zeta^2 - 1 - \zeta^2 + 2\zeta\right) d\zeta \\
&= f_1^j \frac{\det[J]^j}{2} \int_0^1 \left(1 - 2\zeta + \zeta^2\right) d\zeta \\
&= f_1^j \frac{\det[J]^j}{2} \left(\zeta - 2\frac{\zeta^2}{2} + \frac{\zeta^3}{3} \right)_0^1 \\
&= f_1^j \frac{\det[J]^j}{2} \left(1 - 1 + \frac{1}{3} \right) \\
&= f_1^j \det[J]^j \left(\frac{1}{6} \right)
\end{aligned}
$$

And

20

$$
\begin{aligned}
Q_2^j &= \int_0^1 \int_0^{1-\zeta} f_2^j N_2 \det[J]^j \, d\eta d\zeta \\
&= \det[J]^j \; f_2^j \int_0^1 \int_0^{1-\zeta} \zeta d\eta d\zeta \\
&= \det[J]^j \; f_2^j \int_0^1 \left(\int_0^{1-\zeta} \zeta d\eta \right) d\zeta \\
&= \det[J]^j \; f_2^j \int_0^1 \left(\zeta \eta \right)_0^{1-\zeta} d\zeta \\
&= \det[J]^j \; f_2^j \int_0^1 \left(\zeta \left(1-\zeta\right) \right) d\zeta \\
&= \det[J]^j \; f_2^j \int_0^1 \left(\zeta - \zeta^2 \right) d\zeta \\
&= \det[J]^j \; f_2^j \left(\frac{\zeta^2}{2} - \frac{\zeta^3}{3} \right)_0^1 \\
&= \det[J]^j \; f_2^j \left(\frac{1}{2} - \frac{1}{3} \right) \\
&= \det[J]^j \; f_2^j \left(\frac{1}{6} \right)
\end{aligned}
$$

And

$$
\begin{aligned}
Q_3^j &= \int_0^1 \int_0^{1-\zeta} f_3^j N_3 \det[J]^j \, d\eta d\zeta \\
&= \det[J]^j \; f_3^j \int_0^1 \int_0^{1-\zeta} \eta d\eta d\zeta \\
&= \det[J]^j \; f_3^j \int_0^1 \left(\frac{\eta^2}{2} \right)_0^{1-\zeta} d\zeta \\
&= \det[J]^j \; f_3^j \int_0^1 \frac{\left(1-\zeta\right)^2}{2} d\zeta \\
&= \det[J]^j \; f_3^j \int_0^1 \frac{\left(1-2\zeta+\zeta^2\right)}{2} d\zeta \\
&= \frac{\det[J]^j}{2} \; f_3^j \left(\zeta - \frac{2\zeta^2}{2} + \frac{\zeta^3}{3} \right)_0^1 \\
&= \frac{\det[J]^j}{2} \; f_3^j \left(1 - 1 + \frac{1}{3} \right) \\
&= \det[J]^j \; f_3^j \left(\frac{1}{6} \right)
\end{aligned}
$$

Hence

$$
\begin{aligned}
I_3 &= \begin{pmatrix} v_1^j & v_2^j & v_3^j \end{pmatrix} \begin{Bmatrix} Q_1^j \\ Q_2^j \\ Q_3^j \end{Bmatrix} \\
&= \begin{pmatrix} v_1^j & v_2^j & v_3^j \end{pmatrix} \begin{Bmatrix} \det[J]^j f_1^j\left(\frac{1}{6}\right) \\ \det[J]^j f_2^j\left(\frac{1}{6}\right) \\ \det[J]^j f_3^j\left(\frac{1}{6}\right) \end{Bmatrix} \\
&= \frac{\det[J]^j}{6} \begin{pmatrix} v_1^j & v_2^j & v_3^j \end{pmatrix} \begin{Bmatrix} f_1^j \\ f_2^j \\ f_3^j \end{Bmatrix}
\end{aligned}
$$

But $\det[J]^j = 2A^j$ Hence the above becomes

$$
I_3 = \frac{A^j}{3} \begin{pmatrix} v_1^j & v_2^j & v_3^j \end{pmatrix} \begin{Bmatrix} f_1^j \\ f_2^j \\ f_3^j \end{Bmatrix}
$$

Now we have the integral in eq (2) completed. We now have our local equations completed. Here it is. We next need to assemble them.

$$
-\frac{1}{4A^j} \begin{bmatrix} v_1^j & v_2^j & v_3^j \end{bmatrix} \begin{bmatrix} K_{1,1}^j & K_{1,2}^j & K_{1,3}^j \\ K_{2,1}^j & K_{2,2}^j & K_{2,3}^j \\ K_{3,1}^j & K_{3,2}^j & K_{3,3}^j \end{bmatrix} \begin{Bmatrix} u_1^j \\ u_2^j \\ u_3^j \end{Bmatrix} + \overbrace{\frac{A^j}{3} \begin{pmatrix} v_1^j & v_2^j & v_3^j \end{pmatrix} \begin{Bmatrix} f_1^j \\ f_2^j \\ f_3^j \end{Bmatrix}}^{\text{This is the above integral we just completed}} = 0
$$

$$
-\frac{1}{4A^j} \begin{bmatrix} K_{1,1}^j & K_{1,2}^j & K_{1,3}^j \\ K_{2,1}^j & K_{2,2}^j & K_{2,3}^j \\ K_{3,1}^j & K_{3,2}^j & K_{3,3}^j \end{bmatrix} \begin{Bmatrix} u_1^j \\ u_2^j \\ u_3^j \end{Bmatrix} + \frac{A^j}{3} \begin{Bmatrix} f_1^j \\ f_2^j \\ f_3^j \end{Bmatrix} = 0
$$

$$
\frac{1}{4A^j} \begin{bmatrix} K_{1,1}^j & K_{1,2}^j & K_{1,3}^j \\ K_{2,1}^j & K_{2,2}^j & K_{2,3}^j \\ K_{3,1}^j & K_{3,2}^j & K_{3,3}^j \end{bmatrix} \begin{Bmatrix} u_1^j \\ u_2^j \\ u_3^j \end{Bmatrix} = \frac{1}{3} A^j \begin{Bmatrix} f_1^j \\ f_2^j \\ f_3^j \end{Bmatrix}
$$

Where

$$\begin{bmatrix} K^j_{1,1} & K^j_{1,2} & K^j_{1,3} \\ K^j_{2,1} & K^j_{2,2} & K^j_{2,3} \\ K^j_{3,1} & K^j_{3,2} & K^j_{3,3} \end{bmatrix} = \begin{bmatrix} b_1^2 + c_1^2 & b_2 b_1 + c_2 c_1 & b_3 b_1 + c_3 c_1 \\ b_2 b_1 + c_2 c_1 & b_2^2 + c_2^2 & b_3 b_2 + c_3 c_2 \\ b_3 b_1 + c_3 c_1 & b_3 b_2 + c_3 c_2 & b_3^2 + c_3^2 \end{bmatrix}$$

4.3 Note on the shape functions

Looking at the trial function in eq (9), repeated here

$$u^j = u^j_1 + \left(u^j_2 - u^j_1\right)\zeta + \left(u^j_3 - u^j_1\right)\eta$$

Hence we see that

$$\begin{aligned} u^j &= u^j_1 + u^j_2\zeta - u^j_1\zeta + u^j_3\eta - u^j_1\eta \\ &= u^j_1\overbrace{(1-\zeta-\eta)}^{N_1} + u^j_2\overbrace{\zeta}^{N_2} + u^j_3\overbrace{\eta}^{N_3} \end{aligned} \tag{2}$$

And since we are looking for a trial function to be of the form $u_1\,(Basis_1) + u_2\,(Basis_2) + u_3\,(Basis_3)$ we see from the above that the 3 shape or basis functions are the following

$$\begin{aligned} N_1 &= 1 - \zeta - \eta \\ N_2 &= \zeta \\ N_3 &= \eta \end{aligned}$$

And since we are using the Galerkin method, where the test function uses the same basis functions as the trial function, we can write the test function as

$$v^j = v^j_1 N_1 + v^j_2 N_2 + v^j_3 N_3$$

5 Assembly of the global stiffness matrix

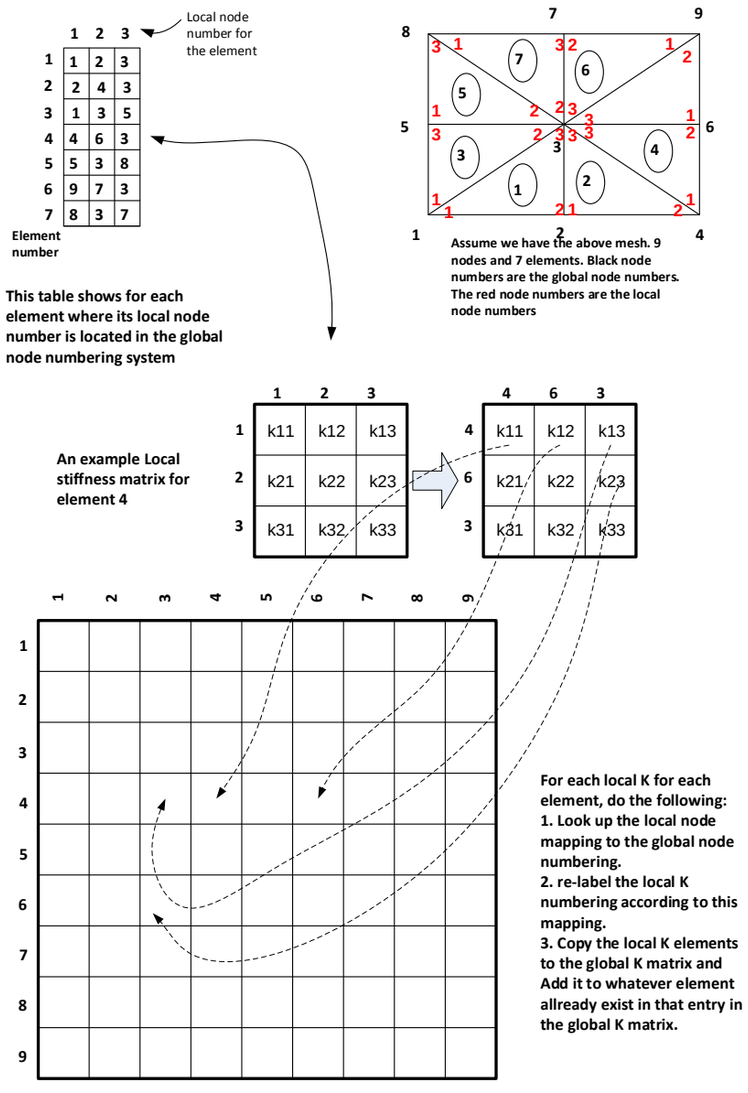
The global stiffness matrix K is always square and symmetric and positive definite. (At least for structural analysis). Recall a positive definite matrix K is one such that for any nonzero vector x we always have $x^* Ax > 0$ where x^* is the conjugate of x. Properties of positive definite matrix is that all its eigenvalues are positive, and it has positive determinant, and hence a positive definite matrix is always invertible.

In addition, the global stiffness matrix is banded. This means that all non-zero elements are found along bands close to the main diagonal of the matrix. Within the band itself, some values can be zero.

The width of the band is a function of the numbering of the nodes used. Different node numbering can result in smaller band width. We want to have as small a band width as possible to take advantage of some numerical methods that can utilize banded matrices.

Band width can be reduced if we keep the node numbering in each element as close as possible to each others.

Now that we have found the local stiffness matrix K^j for element j we can assemble the global stiffness matrix as shown in this diagram. The direct stiffness construction method is used. This is explained in the following diagram



6 Assembly of the global load vector

This follows in similar fashion as above. The 3 elements Load vector $\left\{f\right\}$ for element j is added to the entries of the global load vector $\{F\}$ using the node numbering mapping.

7 Modification of the final global stiffness matrix and load vectors and final solution

Now we have the following equation

$$[K]\{u\} = \{F\}$$

Where K is the assembled global stiffness matrix and $\{F\}$ is the assembled global load vector. Before we solve for $\{u\}$, which is the stress function at all the nodes, we must modify K and F to take care of the given boundary conditions. I attach below 2 pages from a book which gives a good explanation and small example on this point.

Now that we have the modified K^* and F^* you can solve for $\{u\}$ using your favorite linear equations solver.

Appendix 3

MODIFYING THE SYSTEM OF EQUATIONS

The system of equations

$$[K]\{\Phi\}=\{F\}$$

or

$$[K]\{U\}=\{F\}+\{P\}$$

obtained by using the direct stiffness procedure must be modified whenever some of the values in $\{\Phi\}$ or $\{U\}$ are known. All field problems except some problems involving convection heat transfer must have some of the boundary values specified and all solid mechanics problems must have displacements specified to eliminate rigid body motion. Therefore, the modification of the system of equations to incorporate known nodal conditions is more the rule than the exception.

Our objective here is to discuss and then illustrate a systematic procedure for modifying $[K]$ and $\{F\}$ such that we satisfy two criteria. First, we must obtain the correct answers for all values in $\{\Phi\}$ or $\{U\}$. Second, we do not want to change the size of $[K]$, $\{F\}$, and $\{P\}$ because this leads to programming difficulties. We shall consider the steady-state situation first and then discuss the modification of equations associated with time-dependent field problems.

III.1 STEADY-STATE EQUATIONS

The modification of the system of equations $[K]\{\Phi\}=\{F\}$ is a two-step procedure once the subscript of the known nodal parameter is available. For example, suppose that Φ_5 has a known value. The modification proceeds as follows.

1. All of the coefficients in row five are set equal to zero except the diagonal term, which is left unaltered. In equation form, $K_{5j}=0$, $j=1,\ldots,n$ and $j\neq 5$. The associated term in the column vector $\{F\}$, F_5, is replaced by the product $K_{55}\Phi_5$.
2. All of the remaining equations are modified by subtracting the product $K_{j5}\Phi_5$ from F_j and then setting $K_{j5}=0$, $j=1,\ldots,n$, $j\neq 5$.

ILLUSTRATIVE EXAMPLE

Modify the following system of equations when $\Phi_1 = 150$ and $\Phi_5 = 40$.

$$\begin{bmatrix} 55 & -46 & 0 & 0 & 0 \\ -46 & 140 & -46 & 0 & 0 \\ 4 & -46 & 110 & -46 & 4 \\ 0 & 0 & -46 & 142 & -46 \\ 0 & 0 & 4 & -46 & 65 \end{bmatrix} \begin{Bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \end{Bmatrix} = \begin{Bmatrix} 500 \\ 2000 \\ 1000 \\ 2000 \\ 900 \end{Bmatrix}$$

To implement step one, we set all of the coefficients in rows one and five to zero except the diagonal terms, which are left unaltered. The corresponding terms in $\{F\}$, F_1 and F_5, are then replaced by $F_1 = K_{11}\Phi_1$ and $F_5 = K_{55}\Phi_5$, respectively. This step yields

$$\begin{bmatrix} 55 & 0 & 0 & 0 & 0 \\ -46 & 140 & -46 & 0 & 0 \\ 4 & -46 & 110 & -46 & 4 \\ 0 & 0 & -46 & 142 & -46 \\ 0 & 0 & 0 & 0 & 65 \end{bmatrix} \begin{Bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \end{Bmatrix} = \begin{Bmatrix} 8250 \\ 2000 \\ 1000 \\ 2000 \\ 2600 \end{Bmatrix}$$

The second step involves the elimination of the columns of coefficients that multiply Φ_1 and Φ_5. This is accomplished by transferring the coefficients involving Φ_1 and Φ_5 to the right-hand side. For example, F_2 becomes $2000 + 46\Phi_1$, or 8900. Completion of this step gives

$$\begin{bmatrix} 55 & 0 & 0 & 0 & 0 \\ 0 & 140 & -46 & 0 & 0 \\ 0 & -46 & 110 & -46 & 0 \\ 0 & 0 & -46 & 142 & 0 \\ 0 & 0 & 0 & 0 & 65 \end{bmatrix} \begin{Bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \Phi_4 \\ \Phi_5 \end{Bmatrix} = \begin{Bmatrix} 8250 \\ 8900 \\ 240 \\ 3840 \\ 2600 \end{Bmatrix}$$

III.2 TIME-DEPENDENT EQUATIONS

The incorporation of specified nodal values in time-dependent problems is more complicated because the solution procedure involves combinations of $[C]$ and $[K]$, namely $[A]$ and $[P]$. We shall place the same requirement on the time-dependent solution that was placed on the steady-state solution. We want to keep the dimensions of $[C]$ and $[K]$ and thus $[A]$ and $[P]$ the same after modification as they were before modification.

The algorithm for modifying $[C]$ and $[K]$ is more easily understood once we have looked at a specific problem. Let us reconsider the problem in Section 14.5 without the heat source at node one. Instead we assume that $\Phi_1 = 40°C$ for all time values. The vector of initial conditions $\{\Phi\}_a$ becomes $\{\Phi\}_a^T = [40 \quad 0 \quad 0]$.

8 Conclusion

The following are the main steps in solving the torsion problem described in this report.

Physical problem

Generate the Mathematical model

Strong form (ODE/PDE)

Convert to Unsymmetrical Weak form. WRM

Convert to symmetric Weak form

Determine the element type to use

Determine trial and test functions, determine the shape functions

Convert integrals to natural coordinates

Compute the local stiffness and load vectors

Assemble global and load matrices

Adjust final matrices for given boundary conditions

Solve the final system. Find the stress function

Use the stress function to determine the calculated applied torque

Determine the scaling ratio by comparing the calculated torque to the actual torque

Solve for the sheer stresses, apply the scaling ratio

9 References

1. Methods of computer modeling in Engineering & the sciences. Volume 1. Satya N. Atluri. Tech Science Press
2. Lecture notes, MAE 207. Spring and Fall 2006. UCI. Instructor: Professor Atluri SN.
3. Applied finite element analysis. Larry Segerlind.
4. The finite element method for engineers. Kenneth Huebner.
5. Mathematical methods in the physical sciences. 2nd edition. Mary Boas.
6. Fellow students reports and code from MAE 207 projects: Roy Culver , Paul Nylandres, Q Wang. see other related reports on my MAE 207 class web page

2.1.2 Report on Weighted Residual methods and FEM

Examples solving an ODE using finite elements method. Mathematica and Matlab implementation and animation

Nasser M. Abbasi

Sept 28,2006 Compiled on September 4, 2021 at 9:03pm

Contents

This report is a basic review of weighted residual methods and FEM. Some basic differential equations are used to illustrate the method. Mathematica and Matlab code written to solve numerically a first and second order ODE using FEM.

1 Introduction

This is a basic review of Finite Elements Methods from Mathematical point of view with examples of how it can be used to numerically solve first and second order ODE's. Currently I show how to use FEM to solve first and second order ODE. I am also working on a detailed derivation and implementation using FEM to solve the 2D Poisson's equation but this work is not yet completed.

FEM is a numerical method for solving differential equations (ordinary or partial). It can also be used to solve non-linear differential equations but I have not yet studied how this is done. FEM is a more versatile numerical method than the finite difference methods for solving differential equations as it supports more easily different types of geometry and boundary conditions, in addition the solution of the differential equation found using FEM can be used at any point in the domain and not just on the grid points as the case is with finite difference methods. On the other hand FEM is more mathematically complex method, and its implementation is not as straight forward as with finite difference methods.

2

Considering only ordinary differential equations with constant coefficients over the x domain (real line).

$$a\frac{dy}{dx} + b\, y(x) = f(x)$$
$$a\frac{dy}{dx} + b\, y(x) - f(x) = 0$$

defined over $0 \le x \le 1$ with the boundary condition $y(0) = y_0$. In the above, only when $y(x)$ is the exact solution, call it $y_e(x)$, do we have the above identity to be true.

In other words, only when $y = y_e$ we can write that $a\frac{dy_e}{dx} + b\, y_e(x) - f(x) = 0$. Such a differential equations can be represented as an operator L

$$L := \left(y, x\right) \rightarrow a\frac{dy}{dx} + by(x) - f(x) \tag{1}$$

If we know the exact solution, call it $y_e(x)$ then we write

$$L\left(y_e, x\right) \rightarrow a\frac{dy_e}{dx} + by_e\,(x) - f(x)$$
$$\rightarrow 0$$

FEM is based on the weighted residual methods (WRM) where we assume that the solution of an differential equation is the sum of weighted basis functions represented by the symbol ϕ_i in here. This is in essence is similar to Fourier series, where we represent a function as a weighted sums of series made up of the basis functions which happened to be in that case the sin and cosine functions.

So the first step in solving the differential equation is to assume that the solution, called $\tilde{y}(x)$, can be written as

$$\tilde{y}(x) = \sum_{j=1}^{N} q_j\, \phi_j(x)$$

Where q_j are unknown coefficients (the weights) to be determined. Hence the main computational part of FEM will be focused on determining these coefficients.

When we substitute this assumed solution in the original ODE such as shown in the above example, equation (1) now becomes

$$L\left(\tilde{y}, x\right) = a\frac{d\tilde{y}}{dx} + b\tilde{y}\,(x) - f(x)$$
$$= R(x)$$

Where $R(x)$ is called the differential equation residual, which is a function over x and in general will not be zero due to the approximate nature of our assumed solution. Our goal is to to determine the coefficients q_j which will make $R(x)$ the minimum over the domain of the solution.

The optimal case is for $R(x)$ to be zero over the domain. One method to be able to achieve this is by forcing $R(x)$ to meet the following requirement

$$\int_{\Omega} R(x)\, v_i(x)\, dx = 0$$

for all possible sets of function $v_i(x)$ which are also defined over the same domain. The functions $v_i(x)$ are linearly independent from each others. If we can make $R(x)$ satisfy the above for each one of these functions, then this implies that $R(x)$ is zero. And the solution $\tilde{y}(x)$ will be as close as possible to the exact solution. We will find out in FEM that the more elements we use, the closer to the exact solution we get. This property of convergence when it comes to FEM is important, but not analyzed here.

Each one of these functions $v_i(x)$ is called a test function (or a weight function), hence the name of this method.

In the galerkin method of FEM, the test functions are chosen from the same class of functions as the trial functions as will be illustrated below.

By making $R(x)$ satisfy the above integral equation (called the weak form of the original differential equation) for N number of test functions, where N is the number of the unknown coefficients q_i, then we obtain a set of N algebraic equations, which we can solve for q_i.

The above is the basic outline of all methods based on the weighted residual methods. The choice of the trial basis functions, and the choice of the test functions, determine the method used. Different numerical schemes use different types of trial and test functions.

In the above, the assumed solution $\tilde{y}(x)$ is made up of a series of trial functions (the basis). This solution is assumed to be valid over the whole domain. This is called a global trial function. In methods such as Finite Elements and Finite volume, the domain itself is descritized, and the assumed solution is made up of a series of solutions, each of which is defined over each element resulting from the discretization process.

In addition, in FEM, the unknown coefficients, called q_i above, have a physical meaning, they are taken as the solution values at each node. The trial functions themselves are generated by using polynomial interpolation between the nodal values. The polynomial can be linear, quadratic or cubic polynomial or higher order.

Lagrangian interpolation method is normally used for this step. The order of the polynomial is determined by the number of unknowns at the nodes. For example, if our goal is to determine the axial displacement at each node, then we have 2 unknowns, one at each end of the element. Hence a linear interpolation will be sufficient in this case, since a linear polynomial $a_0 + a_1 x$ contain 2 unknowns, the a_1 and a_0. If in addition to the axial displacement, we wish to also solve for the rotation at each end of the element, hence we have a total of 4 unknowns, 2 at each end of the element, which are the displacement and the rotation.

Hence in this case the minimum interpolating polynomial needed will be a cubic polynomial $a_0 + a_1 x + a_2 x^2 + a_3 x^3$. In the examples below, we assume that we are only solving for axial displacement, hence a linear polynomial will be sufficient.

At first, we will work with global trial functions to illustrate how to use weighted residual method.

2 Weighted Residual method. Global trial functions.

The best way to learn how to use WRM is by working over and programming some examples.

We analyze the solution in terms of errors and the effect of changing N on the result.

2.1 First example. First order ODE

Given the following ODE

$$\frac{dy}{dx} - y(x) = 0$$

defined over $0 \leq x \leq 1$ with the boundary condition $y(0) = 1$, we wish to solve this numerically using the WRM. This ODE has an exact solution of $y = e^x$.

The solution using WRM will always follow these steps.

STEP 1

Assume a solution that is valid over the domain $0 \leq x \leq 1$ to be a series solution of trial (basis) functions. We start by selecting a trial functions. The assumed solution takes the form of

$$\tilde{y}(x) = \tilde{y}_0 + \sum_{j=1}^{N} q_j \, \phi_j(x)$$

Where $\phi_j(x)$ is the trial function which we have to choose, and q_j are the N unknown coefficients to be determined subject to a condition which will be shown below. $\tilde{y}_0$ is the assumed solution which needs to be valid only at the boundary conditions. Hence in this example, since we are given that the solution must be 1 at the initial condition $x = 0$, then $\tilde{y}_0 = 1$ will satisfy this boundary condition. Hence our trial solution is

$$\tilde{y} = 1 + \sum_{j=1}^{N} q_j \, \phi_j(x)$$

STEP 2

Now we decide on what trial function $\phi(x)$ to use. For this example, we can select the trial functions to be polynomials in x or trigonometric functions. Let us choose a polynomial $\phi_j(x) = x^j$, hence our assumed solution becomes

$$\tilde{y} = 1 + \sum_{j=1}^{N} q_j \, x^j \tag{1}$$

Now we need to determine the coefficients q_j, and then our solution will be complete. This is done in the following step.

STEP 3

Substituting the above assumed solution back into the original ODE, we obtain the residual $R(x)$

$$\frac{d\tilde{y}}{dx} - \tilde{y}(x) = R(x)$$

$R(x)$ is the ODE residual. This is the error which will result when the assumed solution is used in place of the exact solution.

Hence from (1), we find the residual to be

$$\begin{aligned} R(x) &= \frac{d}{dx}\left(1+\sum_{j=1}^{N} q_j\, x^j\right) - \left(1+\sum_{j=1}^{N} q_j\, x^j\right) \\ &= \sum_{j=1}^{N} q_j\, j x^{j-1} - \left(1+\sum_{j=1}^{N} q_j\, x^j\right) \\ &= -1 + \sum_{j=1}^{N} q_j\, \left(j x^{j-1} - x^j\right) \end{aligned} \tag{2}$$

Our goal now is to reduce this residual to minimum. The way we achieve this is by requiring that the residual satisfies the following integral equation

$$\int_0^1 v_i(x) R(x) dx = 0 \qquad i = 1 \cdots N \tag{3}$$

The above is a set of N equations. The integration is carried over the whole domain, and $v_i(x)$ is a weight (test) function, which we have to also select. Depending on the numerical scheme used, the test function will assume different forms.

For the Galerkin method, we select the test function to be from the same family of functions as the trial (basis) functions. Hence in this example, let us select the test function to the following polynomial

$$v_i(x) = x^{i-1} \tag{4}$$

STEP 4

We now choose a value for N and solve the set of equations generated from (3). Let us pick $N = 3$, hence $R(x)$ becomes

$$\begin{aligned} R(x) &= -1 + \sum_{j=1}^{3} q_j \left(j x^{j-1} - x^j\right) \\ &= -1 + q_1\left(1 - x^1\right) + q_2\, \left(2x - x^2\right) + q_3 \left(3x^2 - x^3\right) \end{aligned}$$

Substituting the above in (3) gives

$$\int_{x=0}^{x=1} x^{i-1} R(x) dx = 0 \qquad i = 1 \cdots N$$

$$\int_{x=0}^{x=1} x^{i-1} \left(-1 + q\, \left(1 - x^1\right) + q_2 \left(2x - x^2\right) + q_3\, \left(3x^2 - x^3\right)\right) dx = 0 \qquad i = 1 \cdots 3$$

6

The above generates $N = 3$ equations to solve. They are

$$\int_{x=0}^{x=1} \left(-1 + q_1 \left(1 - x^1\right) + q_2 \left(2x - x^2\right) + q_3 \left(3x^2 - x^3\right)\right) dx = 0 \qquad i = 1$$
$$\int_{x=0}^{x=1} x\left(-1 + q_1 \left(1 - x^1\right) + q_2 \left(2x - x^2\right) + q_3 \left(3x^2 - x^3\right)\right) dx = 0 \qquad i = 2$$
$$\int_{x=0}^{x=1} x^2\left(-1 + q_1 \left(1 - x^1\right) + q_2 \left(2x - x^2\right) + q_3 \left(3x^2 - x^3\right)\right) dx = 0 \qquad i = 3$$

Perfoming the integration above, gives the following 3 equations

$$-1 - q_1\left(-\frac{1}{2}\right) - q_2\left(-\frac{2}{3}\right) - q_3\left(-\frac{3}{4}\right) = 0$$
$$-\frac{1}{2} - q_1\left(-\frac{1}{6}\right) - q_2\left(-\frac{5}{12}\right) - q_3\left(-\frac{11}{20}\right) = 0$$
$$-\frac{1}{3} - q_1\left(-\frac{1}{12}\right) - q_2\left(-\frac{3}{10}\right) - q_3\left(-\frac{13}{30}\right) = 0$$

Which can be written in matrix form as

$$\begin{bmatrix} \frac{1}{2} & \frac{2}{3} & \frac{3}{4} \\ \frac{1}{6} & \frac{5}{12} & \frac{11}{20} \\ \frac{1}{12} & \frac{3}{10} & \frac{13}{30} \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} 1 \\ \frac{1}{2} \\ \frac{1}{3} \end{bmatrix}$$

The solution is

$$q_1 = \frac{72}{71}, q_2 = \frac{30}{71}, q_3 = \frac{20}{71}$$

Hence our assumed series solution is now complete, using the above coefficients, and from equation (1) we write

$$\tilde{y} = 1 + \sum_{j=1}^{N} q_j\, x^j$$
$$\tilde{y} = 1 + q_1\, x + q_2 x^2 + q_3 x^3$$

Hence

$$\tilde{y}(x) = 1 + \frac{72}{71}x + \frac{30}{71}x^2 + \frac{20}{71}x^3$$

Let use compare the above solution to the exact solution $y = e^x$ by comparing the values of the solution over a number of points. This is done using the following small Mathematica code

7

```
In[77]:= Remove["Global`*"]
         exact[x_] := Exp[x]
         approx[x_] := 1 + 72/71 x + 30/71 x^2 + 20/71 x^3
         error[x_] := Abs[exact[x] - approx[x]]
         s = Table[{x, exact[x], approx[x], error[x]}, {x, 0, 1, .1}];
         TableForm[s, TableHeadings → {None, {"x", "exact", "approx", "error"}}]
```

Out[82]//TableForm=

x	exact	approx	error
0	1	1	0
0.1	1.10517	1.10592	0.000744575
0.2	1.2214	1.22197	0.000569073
0.3	1.34986	1.34986	3.47354×10^{-7}
0.4	1.49182	1.49127	0.000557092
0.5	1.64872	1.64789	0.000833947
0.6	1.82212	1.82141	0.00071035
0.7	2.01375	2.01352	0.000231581
0.8	2.22554	2.22592	0.000374564
0.9	2.4596	2.46028	0.000678579
1.	2.71828	2.71831	0.0000280307

Figure 1: Using $N = 3$

To make this more useful, we can examine how the error changes as N changes. The following Mathematica code determines the solution and calculates the same table as above for $N = 1 \cdots 5$

```
In[615]:=
Remove["Global`*"]
ode[y_, x_] := D[y[x], x] - y[x];
sol = Flatten[DSolve[{ode[y, x] == 0, y[0] == 1}, y'[x], x]];
exactSolution[x_] = y[x] /. sol;
trialFunction[x_, j_] := x^j ;
weight[x_, j_] := x^(j-1) ;
eq[i_, residue_] := ∫_from^to weight[x, i] residue dx == 0;
solution[maxN_, from_, to_] := Module[{coeff, residue, j, a},
   coeff = Array[a, maxN];
   yapprox[x_] := 1 + ∑_{j=1}^{maxN} coeff[[j]] trialFunction[x, j];
   residue = ode[yapprox, x] ;
   s = Table[eq[j, residue], {j, 1, maxN}];
   sol = Flatten[Solve[s, coeff]]
  ];

from = 0;
to = 1;
approxSolution[x_, sol_] := 1 + ∑_{j=1}^{maxN} sol[[j, 2]] trialFunction[x, j];
error[x_, sol_] := Abs[exactSolution[x] - approxSolution[x, sol]];
Do[{sol = solution[maxN, from, to];
   Print["N=", maxN, " Approx Solution is ", approxSolution[x, sol]];
   s = Table[{x, exactSolution[x], approxSolution[x, sol], error[x, sol]},
     {x, 0, 1, .1}];
   Print[TableForm[s, TableHeadings → {None, {"x", "exact", "approx", "error"}}]];
  }, {maxN, 1, 5}];
```

Figure 2: Code used for $N = 5$

8

N=1 Approx Solution is $1+2x$

x	exact	approx	error
0	1	1	0
0.1	1.10517	1.2	0.0948291
0.2	1.2214	1.4	0.178597
0.3	1.34986	1.6	0.250141
0.4	1.49182	1.8	0.308175
0.5	1.64872	2.	0.351279
0.6	1.82212	2.2	0.377881
0.7	2.01375	2.4	0.386247
0.8	2.22554	2.6	0.374459
0.9	2.4596	2.8	0.340397
1.	2.71828	3.	0.281718

N=2 Approx Solution is $1+\frac{6x}{7}+\frac{6x^2}{7}$

x	exact	approx	error
0	1	1	0
0.1	1.10517	1.09429	0.0108852
0.2	1.2214	1.20571	0.0156885
0.3	1.34986	1.33429	0.0155731
0.4	1.49182	1.48	0.0118247
0.5	1.64872	1.64286	0.00586413
0.6	1.82212	1.82286	0.000738342
0.7	2.01375	2.02	0.00624729
0.8	2.22554	2.23429	0.00874479
0.9	2.4596	2.46571	0.00611117
1.	2.71828	2.71429	0.00399611

N=3 Approx Solution is $1+\frac{72x}{71}+\frac{30x^2}{71}+\frac{20x^3}{71}$

x	exact	approx	error
0	1	1	0
0.1	1.10517	1.10592	0.000744575
0.2	1.2214	1.22197	0.000569073
0.3	1.34986	1.34986	3.47354×10^{-7}
0.4	1.49182	1.49127	0.000557092
0.5	1.64872	1.64789	0.000833947
0.6	1.82212	1.82141	0.00071035
0.7	2.01375	2.01352	0.000231581
0.8	2.22554	2.22592	0.000374564
0.9	2.4596	2.46028	0.000678579
1.	2.71828	2.71831	0.0000280307

N=4 Approx Solution is $1+\frac{1000x}{1001}+\frac{510x^2}{1001}+\frac{20x^3}{143}+\frac{10x^4}{143}$

x	exact	approx	error
0	1	1	0
0.1	1.10517	1.10514	0.0000290599
0.2	1.2214	1.22141	7.83125×10^{-6}
0.3	1.34986	1.3499	0.0000382953
0.4	1.49182	1.49186	0.0000354422
0.5	1.64872	1.64873	5.00303×10^{-6}
0.6	1.82212	1.82209	0.0000288903
0.7	2.01375	2.01371	0.0000394208
0.8	2.22554	2.22553	0.000014455
0.9	2.4596	2.45963	0.0000242615
1.	2.71828	2.71828	1.10177×10^{-7}

N=5 Approx Solution is $1+\frac{18090x}{18089}+\frac{9030x^2}{18089}+\frac{3080x^3}{18089}+\frac{630x^4}{18089}+\frac{252x^5}{18089}$

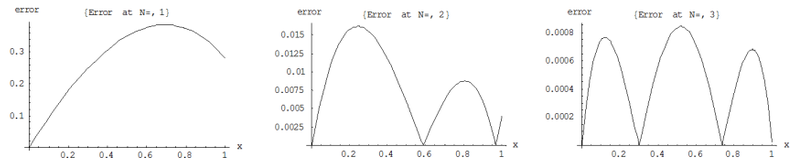
x	exact	approx	error
0	1	1	0
0.1	1.10517	1.10517	4.8554×10^{-7}
0.2	1.2214	1.2214	1.42918×10^{-6}
0.3	1.34986	1.34986	1.13938×10^{-6}
0.4	1.49182	1.49183	6.37093×10^{-7}
0.5	1.64872	1.64872	1.71012×10^{-6}
0.6	1.82212	1.82212	9.14353×10^{-7}
0.7	2.01375	2.01375	8.89238×10^{-7}
0.8	2.22554	2.22554	1.46473×10^{-6}
0.9	2.4596	2.4596	3.20763×10^{-7}
1.	2.71828	2.71828	2.76651×10^{-10}

Figure 3: Result for $N = 5$

This code below plots the absolute error as N changes. Notice that the number of peaks in the error plot is also N which is the polynomial order (the trial solution) used to approximate the exact solution, which is to be expected.

9

```
In[738]:= p = {};
          Do[{sol = solution[maxN, from, to];
            p = Append[p, Plot[error[x, sol], {x, 0, 1}, PlotRange → All, AxesLabel → {"x", "error"}, PlotLabel → {"Error at N=", maxN},
              DisplayFunction → Identity]];
          }, {maxN, 1, 6}];
          Show[GraphicsArray[ {p[[1]], p[[2]], p[[3]] }]]
          Show[GraphicsArray[ {p[[4]], p[[5]], p[[6]] }]]
```




```
Out[740]= - GraphicsArray -
```

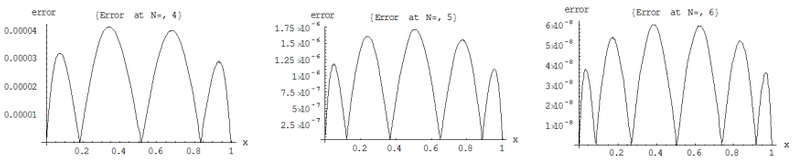


Figure 4: Error using $N = 5$

2.2 Second example. 4th order ODE

Now we will use a more complex example and repeat the above steps. We now want to numerically solve the following

$$\frac{d^4y}{dx^4} + y(x) = 1$$

defined over $0 \le x \le 1$ with the boundary conditions $y(0) = 0, y(1) = 0, y'(0), y'(1) = 0$. This problem is taken from Professor S.N.Atluri text book 'Methods of computer modeling in engineering and the sciences' Volume 1, page 47-50. Professor Atluri used a trigonometric functions for the trial function

$$\tilde{y} = \sum_{i=1}^{N} q_i \sin\left((2i-1)\pi x\right)$$

Which already satisfies the boundary conditions. For the test function, the same function as above is used hence the test (weight) function is

$$v_j(x) = \sin\left(\left(2j-1\right)\pi x\right)$$

The book above then reduces the residual equation to a symmetric form by doing integration by parts before solving it for the coefficients. In here, we will use the unsymmetrical weak form and compare the results with those shown in the above textbook. We now start again with the same steps as we did in the above example.

step 1

Selecting the trial solution.

$$\tilde{y}(x) = \tilde{y}_0 + \sum_{j=1}^{N} q_j \, \phi_j(x)$$

$\tilde{y}_0 = 0$ as this will satisfy the boundary conditions. Hence the trial solution is

$$\tilde{y}(x) = \sum_{j=1}^{N} q_j \phi_j(x)$$

step 2

Selecting trial basis function $\phi_j(x)$. As mentioned above, we select $\phi_j(x) = \sin\left(\left(2j-1\right)\pi x\right)$, hence the trial solution is

$$\tilde{y} = \sum_{j=1}^{N} q_j \sin\left(\left(2j-1\right)\pi x\right)$$

step 3

Substituting the above assumed solution into the original ODE, gives the differential equation residual $R(x)$

$$\frac{d^4\tilde{y}}{dx^4} + \tilde{y}(x) - 1 = R(x)$$

$$\frac{d^4}{dx^4}\left(\sum_{j=1}^{N} q_j \, \phi_j(x)\right) + \left(\sum_{j=1}^{N} q_j \, \phi_j(x)\right) - 1 = R(x)$$

Notice the requirement above that the trial basis functions must be 4 times differentiable, which is the case here. From above we obtain

$$R(x) = \left(\sum_{j=1}^{N} \left(1 + \left(\left(2j-1\right)\pi\right)^4\right) q_j \sin\left(\left(2j-1\right)\pi x\right)\right) - 1$$

Our goal now is to reduce this residual to minimum. The way we achieve this is by requiring that the residual satisfies the following weak form integral equation

$$\int_{\Omega} v_i(x) R(x) dx = 0 \qquad i = 1 \cdots N \tag{3}$$

The above is a set of N equations. The integration is carried over the whole domain, and $v_i(x)$ is a weight (test) function, which we have to also select. As mentioned above, in this problem we select the test function to be

$$v_i(x) = \sin\left(\left(2i-1\right)\pi x\right) \tag{4}$$

step 4

Deciding on a value for N and solving the set of equations generated from (3). Let us pick $N = 3$, hence $R(x)$ becomes

$$\begin{aligned} R(x) &= \left(\sum_{j=1}^{3} \left(1 + \left(\left(2j-1\right)\pi\right)^4\right) \left(q_j \sin\left(2j-1\right)\pi x\right)\right) - 1 \\ &= \left(\left(1+\pi^4\right)\left(q_1 \sin \pi x\right) + \left(1 + (3\pi)^4\right)\left(q_2 \sin\left(3\pi x\right)\right) + \left(1 + (5\pi)^4\right)\left(q_3 \sin 5\pi x\right)\right) - 1 \end{aligned}$$

11

Hence (3) becomes

$$\int_{\Omega} v_i(x) R(x) dx = 0 \qquad i = 1 \cdots N$$

$$\int_0^1 (\sin(2i-1)\pi x)\, R(x) dx = 0 \qquad i = 1 \cdots N$$

$$\int_0^1 (\sin(2i-1)\pi x) \left\{ \left(\left(1+\pi^4\right) \left(q_1 \sin \pi x\right) + \left(1+(3\pi)^4\right) \left(q_2 \sin 3\pi x\right) + \left(1+(5\pi)^4\right) \left(q_3 \sin 5\pi x\right) \right) - 1 \right\} dx = 0 \qquad i = 1 \cdots N$$

The above generates N equations to solve for the coefficients q_i

$$\int_0^1 (\sin \pi x) \left\{ \left(\left(1+\pi^4\right) \left(q_1 \sin \pi x\right) + \left(1+(3\pi)^4\right) \left(q_2 \sin 3\pi x\right) + \left(1+(5\pi)^4\right) \left(q_3 \sin 5\pi x\right) \right) - 1 \right\} dx = 0 \qquad i = 1$$

$$\int_0^1 (\sin 3\pi x) \left\{ \left(\left(1+\pi^4\right) \left(q_1 \sin \pi x\right) + \left(1+(3\pi)^4\right) \left(q_2 \sin (3\pi x)\right) + \left(1+(5\pi)^4\right) \left(q_3 \sin 5\pi x\right) \right) - 1 \right\} dx = 0 \qquad i = 2$$

$$\int_0^1 (\sin 5\pi x) \left\{ \left(\left(1+\pi^4\right) \left(q_1 \sin \pi x\right) + \left(1+(3\pi)^4\right) \left(q_2 \sin 3\pi x\right) + \left(1+(5\pi)^4\right) \left(q_3 \sin 5\pi x\right) \right) - 1 \right\} dx = 0 \qquad i = 3$$

Carrying the integration above and simplifying and solving for q_i gives the numerical solution.

This below is a Mathematica code which solves this problem for different N values, and compares the error as N changes. The error shown is the percentage error in the solution (approximate compared to exact) for up to $N = 10$. The result below agrees well with the result in Professor's Atluri textbook.

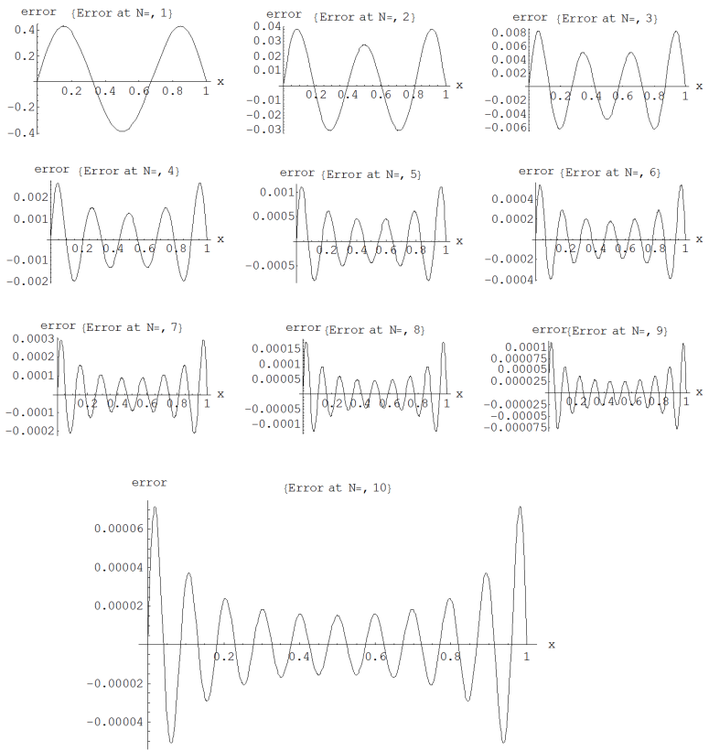


Figure 5: Error for different N using FEM for second order ODE

3 Finite element method

3.1 Example one. First order ODE, linear interpolation

Let us first summarize what we have done so far. Given a differential equation defined over domain Ω, we assume its solution to be of the form $\tilde{u}(x) = \sum_{j=1}^{N} q_j \, \phi_j(x)$.

The function $\phi_j(x)$ is called the j^{th} basis function. $q_j \, \phi_j(x)$ is called a trial function.

The function $\phi_j(x)$ is made up of functions called the shape functions N_k as they are normally called in structural mechanics books.

q_j are the unknown coefficients which are determined by solving N set of equations generated by setting N integrals of the form $\int_{\Omega} R(x) \, v_i(x) \, dx$ to zero. Where $R(x)$ is the differential equation residual. In all what follows N is the taken as the number of nodes.

13

In FEM, we also carry the same basic process as was described above, the differences are the following:

Now we divide the domain itself into a number of elements. Earlier we did not do this.

Next, the $\phi_j(x)$ function is found by assuming the solution to be an interpolation between the nodes of the element. The solution values at the nodes are the q_i and are of course unknown except at the boundaries as given by the problem.

We start by deciding on what interpolation between the nodes to use. We will use polynomial interpolation here. Then $q_j\phi_j(x)$ will become the interpolation function.

In addition, the coefficients q_j represent the solution at the node j. These are the unknowns, which we will solve for by solving the weak form integral equation as many times as there are unknowns to solve for.

By solving for the nodal values, we can then use the interpolating function again to find the solution at any point between the nodes.

This diagram illustrates the above, using the first example given above to solve a differential equation $\frac{du}{dx} - u(x) = 0$ with the given boundary condition of $u(0) = 1$ and defined over $0 \le x \le 1$

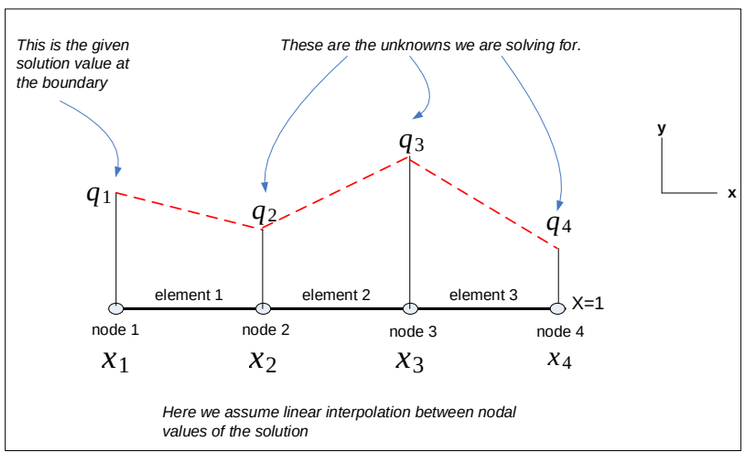


Figure 6: Finite elements with linear interpolation

Using linear interpolation, the solution $u(x)$ when x is located inside first element, is found from

$$u(x) = q_1 + \text{slope } (x - x_0)$$

But the slope of the linear interpolating line over the first element is $\frac{q_2 - q_1}{x_2 - x_1}$, hence the above becomes

$$\begin{aligned} u(x) &= q_1 + \frac{q_2 - q_1}{x_2 - x_1}(x - x_1) \\ &= q_1 \frac{(x_2 - x)}{(x_2 - x_1)} + q_2 \frac{(x - x_1)}{(x_2 - x_1)} \end{aligned}$$

The above is the linear interpolating polynomial. We could also have used the formula of Lagrangian interpolation to arrive at the same result.

The above is the approximate solution which is valid over the first element only. Using superscript to indicate the element number, and assuming we have equal division between nodes of length say h (i.e. element length is h) then we write

$$u^1(x) = q_1 \frac{(x_2 - x)}{h} + q_2 \frac{(x - x_1)}{h}$$

Again, the above is valid for $x_1 \le x \le x_2$. We now do the same for the second element

$$u^2(x) = q_2 \frac{(x_3 - x)}{h} + q_3 \frac{(x - x_2)}{h}$$

The above is valid for $x_2 \le x \le x_3$. And finally for the 3rd element

$$u^3(x) = q_3 \frac{(x_4 - x)}{h} + q_4 \frac{(x - x_3)}{h}$$

The above is valid for $x_2 \le x \le x_3$. This is now illustrated in the following diagram

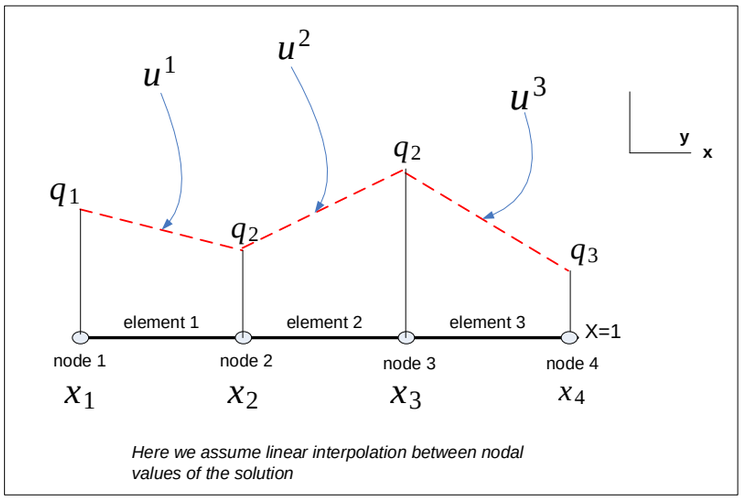


Figure 7: Finite elements with linear interpolation second diagram

Since our goal is to express the global approximate solution $u(x)$ as a series sum of basis functions each multiplied by q_j, we now rewrite each of the u^j to allow this, as follows

$$\begin{aligned} u^1(x) &= q_1 \overbrace{\frac{(x_2 - x)}{h}}^{N_1^1(x)} + q_2 \overbrace{\frac{(x - x_1)}{h}}^{N_2^1(x)} \\ &= q_1 N_1^1(x) + q_2 N_2^1(x) \end{aligned}$$

The above is valid for $x_1 \le x \le x_2$. Notice the use of the following notation: Since each element will have defined on it two shape functions, $N_1(x)$ and $N_2(x)$, one per node, then we use a superscript to indicate the element number. Hence for element 1, we will write its two shapes functions as $N_1^1(x)$ and $N_2^1(x)$.

We now do the same for the second element

$$\begin{aligned} u^2 &= q_2 \overbrace{\frac{(x_3 - x)}{h}}^{N_1^2(x)} + q_3 \overbrace{\frac{(x - x_2)}{h}}^{N_2^2(x)} \\ &= q_2 N_1^2(x) + q_3 N_2^2(x) \end{aligned}$$

The above is valid for $x_2 \le x \le x_3$. And finally for the 3rd element

$$\begin{aligned} u^3 &= q_3 \overbrace{\frac{(x_4 - x)}{h}}^{N_1^3(x)} + q_4 \overbrace{\frac{(x - x_3)}{h}}^{N_2^3(x)} \\ &= q_3 N_1^3 + q_4 N_2^3 \end{aligned}$$

The above is valid for $x_2 \le x \le x_3$. The global trial function is

$$\begin{aligned} u(x) &= u^1 + u^2 + u^3 \\ &= \left(q_1 N_1^1 + q_2 N_2^1\right) + \left(q_2 N_1^2 + q_3 N_2^2\right) + \left(q_3 N_1^3 + q_4 N_2^3\right) \\ &= q_1 N_1^1 + q_2 \left(N_2^1 + N_1^2\right) + q_3 \left(N_2^2 + N_1^3\right) + q_4 \left(N_2^3\right) \end{aligned}$$

The shape function for node 1 is

$$\begin{aligned} \phi_1 &= N_1^1 \\ &= \frac{x_2 - x}{h} \end{aligned}$$

And the shape function for node 2 is

$$\begin{aligned} \phi_2 &= N_2^1 + N_1^2 \\ &= \frac{(x - x_1)}{h} + \frac{(x_3 - x)}{h} \end{aligned}$$

The shape function for node 3 is

$$\begin{aligned} \phi_3 &= N_2^2 + N_1^3 \\ &= \frac{(x - x_2)}{h} + \frac{(x_4 - x)}{h} \end{aligned}$$

The shape function for the last node is

$$\begin{aligned} \phi_4 &= N_2^3 \\ &= \frac{x - x_3}{h} \end{aligned}$$

We see that the shape function for any internal node is

$$\phi_j = \frac{x - x_{j-1}}{h} + \frac{x_{j+1} - x}{h}$$

16

The approximate solution is therefore

$$u(x) = \sum_{i=1}^{Number\ Nodes} q_i \phi_i$$

This completes the first part, which was to express the global approximate solution as sum of basis functions, each multiplied by an unknowns q coefficients.

The diagram below illustrates the above.

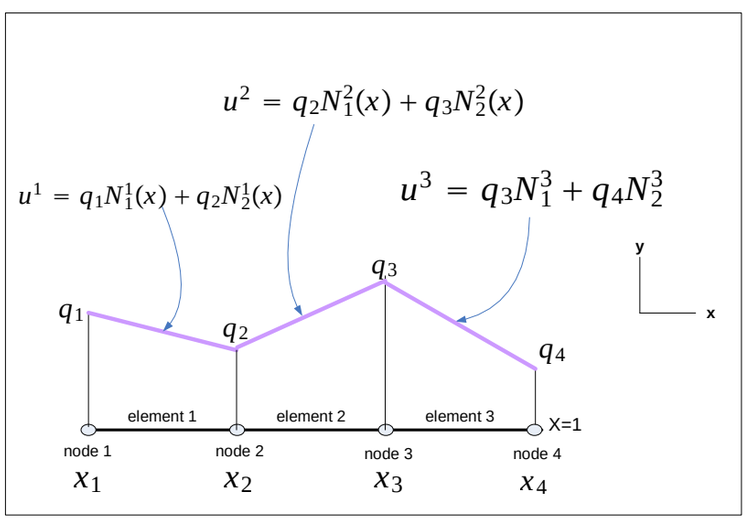


Figure 8: Showing solutions at each element

The diagram below illustrates the numbering used.

17

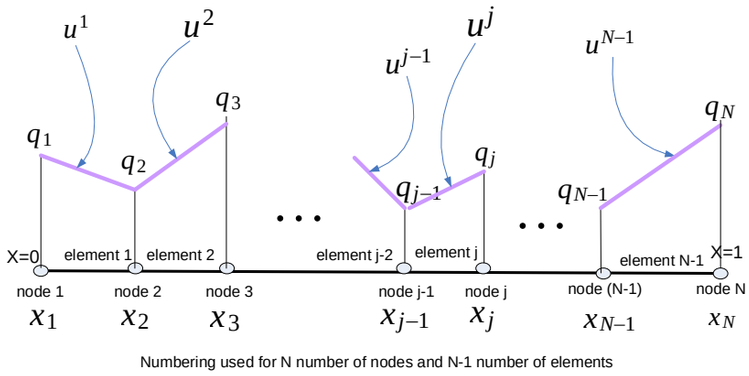


Figure 9: Numbering used for interpolation and elements

Next the residual $R(x)$ is found by substituting the above global solution into the original differential equation as we did before.

Let The differential equation to solve be

$$a\frac{du}{dx} + bu(x) = f(x)$$

Defined over $0 \le x \le 1$ with the initial condition $u(0) = u_0$.

N is the number of nodes, therefore the residual is

$$R(x) = a\frac{d}{dx}\sum_{i=1}^{N} q_i\phi_i + b\sum_{i=1}^{N} q_i\phi_i - f(x)$$
$$R(x) = \sum_{i=1}^{N} q_i\left(a\phi_i' + b\phi_i\right) - f(x)$$

The test functions are

$$v_j = \phi_j \qquad j = 1 \cdots N$$

The weak form of the differential equation is

$$\int_{x=x_1}^{x=x_N} \phi_j(x)R(x)dx = 0 \qquad j = 1 \cdots N$$
$$\int_{x=x_1}^{x=x_N} \phi_j(x)\left(\left[\sum_{i=1}^{N} q_i\left(a\phi_i' + b\phi_i\right)\right] - f(t)\right)dx = 0 \qquad j = 1 \cdots N$$

N equations are obtained from the above, which are solved for the q_i.

The derivatives of the shape functions are found first. Assuming in this example that the domain is divided into 3 elements results in

i	ϕ_i	ϕ'_i
1	$\frac{(x_2-x)}{h}$	$\frac{-1}{h}$
2	$\left\{\frac{(x-x_1)}{h}, \frac{(x_3-x)}{h}\right\}$	$\left\{\frac{1}{h}, \frac{-1}{h}\right\}$
3	$\frac{(x-x_2)}{h}$	$\frac{1}{h}$

In general, if there are N nodes, then for the first node

$$\phi_1 = \frac{(x_2 - x)}{h}$$
$$\phi'_1 = \frac{-1}{h}$$

And for the last node

$$\phi_N = \frac{(x - x_{N-1})}{h}$$
$$\phi'_N = \frac{1}{h}$$

And for any node in the middle

$$\phi_i = \frac{(x - x_{i-1})}{h}$$
$$\phi'_i = \frac{1}{h}$$

for $x_{i-1} \le x \le x_i$ and

$$\phi_i = \frac{(x_{i+1} - x)}{h}$$
$$\phi'_i = \frac{-1}{h}$$

for $x_i \le x \le x_{i+1}$.

Looking back at the weak form integral above, it is evaluated as follows

$$\int_{x=x_1}^{x=x_N} \phi_j \left(\left[\sum_{i=1}^{N} q_i \left(a\phi'_i + b\phi_i \right) \right] - f(x) \right) dx = 0 \qquad j = 1 \cdots N$$

For the first node only, $j = 1$, the following results

$$\int_{x=x_1}^{x=x_N} \phi_1 \left(\left[\sum_{i=1}^{N} q_i \left(a\phi'_i + b\phi_i \right) \right] - f(x) \right) dx = 0$$

Since ϕ_1 is not zero only over $x_1 \le x \le x_2$, and given that $\phi_1 = \frac{x_2-x}{h}, \phi'_1 = \frac{-1}{h}, \phi_2 = \frac{(x-x_1)}{h}$ due to the

range of integration limits, and that $\phi_2' = \frac{1}{h}$, then the above can simplifies to

$$\int_{x=x_1}^{x=x_2} \phi_1 \left(\left[q_1 \left(a\phi_1' + b\phi_1 \right) + q_2 \left(a\phi_2' + b\phi_2 \right) \right] - f(x) \right) dx = 0$$

$$\int_{x=x_1}^{x=x_2} \frac{(x_2 - x)}{h} \left(\left[q_1 \left(\frac{-a}{h} + b\frac{(x_2 - x)}{h} \right) + q_2 \left(a\frac{1}{h} + b\frac{(x - x_1)}{h} \right) \right] - f(x) \right) dx = 0$$

The above simplifies further to

$$-q_1 \left(\frac{(3a + 2b(x_1 - x_2))(x_1 - x_2)^2}{6h^2} \right) - q_2 \frac{(-3a + b(x_1 - x_2))(x_1 - x_2)^2}{6h^2} - \int_{x=x_1}^{x=x_2} \phi_1 f(x) = 0$$

$$-q_1 \left(\frac{(3a + 2b(x_1 - x_2))(x_1 - x_2)^2}{6h^2} \right) - q_2 \frac{(-3a + b(x_1 - x_2))(x_1 - x_2)^2}{6h^2} - \frac{1}{h} \int_{x=x_1}^{x=x_2} (x_2 - x) f(x) = 0$$

Since $x_2 - x_1 = h$ and $x_1 - x_2 = -h$ the above reduces to

$$-q_1 \left(\frac{(3a - 2bh)h^2}{6h^2} \right) - q_2 \frac{(-3a - bh)h^2}{6h^2} - \frac{1}{h} \int_{x=x_1}^{x=x_2} (x_2 - x) f(x) = 0$$

$$q_1 \left(\frac{2bh - 3a}{6} \right) + q_2 \left(\frac{bh + 3a}{6} \right) - \frac{1}{h} \int_{x=x_1}^{x=x_2} (x_2 - x) f(x) = 0$$

The above equation gives the first row in the global stiffness matrix for any first order linear ODE of the form $a\frac{du}{dx} + bu(x) = f(x)$. The above shows that numerical integration is only needed to be performed on the term $\int_{x=x_1}^{x=x_2} (x_2 - x) f(x)$.

Next the last equation is found, which will be the last row of the stiffness matrix. For the last node only $j = N$ the following results

$$\int_{x=x_{N-1}}^{x=x_N} \phi_N \left(\left[\sum_{i=1}^{N} q_i \left(a\phi_i' + b\phi_i \right) \right] - f(x) \right) dx = 0$$

Since ϕ_N domain of influence is $x_{N-1} \le x \le x_N$, the above simplifies to

$$\int_{x=x_{N-1}}^{x=x_N} \phi_N \left(\left[q_{N-1} \left(a\phi_{N-1}' + b\phi_{N-1} \right) + q_N \left(a\phi_N' + b\phi_N \right) \right] - f(x) \right) dx = 0$$

Since $\phi_{N-1} = \frac{(x_N - x)}{h}, \phi_{N-1}' = \frac{-1}{h}, \phi_N = \frac{x - x_{N-1}}{h}, \phi_N' = \frac{1}{h}$ the above becomes

$$\int_{x=x_{N-1}}^{x=x_N} \frac{x - x_{N-1}}{h} \left(\left[q_{N-1} \left(\frac{-a}{h} + b\frac{(x_N - x)}{h} \right) + q_N \left(a\frac{1}{h} + b\frac{x - x_{N-1}}{h} \right) \right] - f(x) \right) dx = 0$$

Which simplifies to

$$N\text{–}1\frac{(3a+b\,(x_{N-1}-x_N))\,(x_{N-1}-x_N)^2}{6h^2}-q_N\frac{(-3a+2b\,(x_{N-1}-x_N))\,(x_{N-1}-x_N)^2}{6h^2}-\frac{1}{h}\int_{x=x_{N-1}}^{x=x_N}(x-x_{N-1})\,f(x)\,dx=0$$

Letting $x_{N-1}-x_N=-h$ in the above becomes

$$q_{N-1}\left(\frac{bh-3a}{6}\right)+q_N\frac{(3a+2bh)}{6}-\frac{1}{h}\int_{x=x_{N-1}}^{x=x_N}(x-x_{N-1})\,f(x)=0$$

Hence the last row of the stiffness matrix can be determined directly except for the term under the integral which needs to be evaluated using integration.

Now the equation that represents any internal node is found. This will be any row in the global stiffness matrix between the first and the last row.

For any j other than 1 or N the following results

$$\int_{x=x_{j-1}}^{x=x_j}\phi_j\left(\left[\sum_{i=1}^{N}q_i\left(a\phi_i'+b\phi_i\right)\right]-f\,(x)\right)dx+\int_{x=x_j}^{x=x_{j+1}}\phi_j\left(\left[\sum_{i=1}^{N}q_i\left(a\phi_i'+b\phi_i\right)\right]-f(x)\right)dx=0$$

Where the integral was broken into two parts to handle the domain of influence of the shape functions.

$$\int_{x=x_{j-1}}^{x=x_j}\phi_j\left(\;q_{j-1}\left(a\phi_{j-1}'+b\phi_{j-1}\right)+q_j\left(a\phi_j'+b\phi_j\right)-f(x)\right)dx+\int_{x=x_j}^{x=x_{j+1}}\phi_j\left(q_j\left(a\phi_j'+b\phi_j\right)+q_{j+1}\left(a\phi_{j+1}'+b\phi_{j+1}\right)-f(x)\right)dx=0$$

For

$$x_{j-1}\le x\le x_j,\phi_j=\frac{x-x_{j-1}}{h},\phi_j'=\frac{1}{h},\phi_{j-1}=\frac{x_j-x}{h},\phi_{j-1}'=\frac{-1}{h}$$
$$x_j\le x\le x_{j+1},\phi_j=\frac{x_{j+1}-x}{h},\phi_j'=\frac{-1}{h},\phi_{j+1}=\frac{x-x_j}{h},\phi_{j+1}'=\frac{1}{h}$$

Hence the weak form integral can be written as

$$\int_{x=x_{j-1}}^{x=x_j}\frac{x-x_{j-1}}{h}\left(q_{j-1}\left(\frac{-a}{h}+b\frac{x_j-x}{h}\right)+q_j\left(\frac{a}{h}+b\frac{x-x_{j-1}}{h}\right)-f(x)\right)dx$$
$$+\int_{x=x_j}^{x=x_{j+1}}\frac{x_{j+1}-x}{h}\left(q_j\left(\frac{-a}{h}+b\frac{x_{j+1}-x}{h}\right)+q_{j+1}\left(\frac{a}{h}+b\frac{x-x_j}{h}\right)-f(x)\right)dx=0$$

Which simplifies to

$$q_{j-1}\left(\frac{\left(-3a-b\left(x_{j-1}-x_j\right)\right)\left(x_{j-1}-x_j\right)^2}{6h^2}\right)+q_j\frac{\left(\left(3a-2b\left(x_{j-1}-x_j\right)\right)\left(x_{j-1}-x_j\right)^2\right)}{6h^2}-\frac{1}{h}\int_{x=x_{j-1}}^{x=x_j}\left(x-x_{j-1}\right)f(x)dx+$$
$$q_j\left(\frac{\left(-3a-2b\left(x_j-x_{j+1}\right)\right)\left(x_j-x_{j+1}\right)^2}{6h^2}\right)+q_{j+1}\left(\frac{\left(3a-b\left(x_j-x_{j+1}\right)\right)\left(x_j-x_{j+1}\right)^2}{6h^2}\right)-\frac{1}{h}\int_{x=x_j}^{x=x_{j+1}}\left(x_{j+1}-x\right)f(x)dx=0$$

For equal distance between elements, $x_j - x_{j+1} = -h, x_{j-1} - x_j = -h$ the above simplifies to

$$q_{j-1}\left(\frac{(-3a+bh)\,h^2}{6h^2}\right) + q_j\frac{\left((3a+2bh)\,h^2\right)}{6h^2} - \frac{1}{h}\int_{x=x_{j-1}}^{x=x_j}\left(x-x_{j-1}\right)f(x)dx + q_j\left(\frac{(-3a+2bh)\,h^2}{6h^2}\right) + q_{j+1}\left(\frac{(3a+bh)\,h^2}{6h^2}\right) - \frac{1}{h}\int_{x=x_j}^{x=x_{j+1}}\left(x_{j+1}-x\right)f(x)\,dx = 0$$

Combining gives

$$q_{j-1}\left(\frac{-3a+bh}{6}\right) + q_j\left(\frac{2bh}{3}\right) + q_{j+1}\left(\frac{3a+bh}{6}\right) - \frac{1}{h}\left(\int_{x=x_{j-1}}^{x=x_j}\left(x-x_{j-1}\right)f(x)dx + \int_{x=x_j}^{x=x_{j+1}}\left(x_{j+1}-x\right)f(x)\,dx\right) = 0$$

The above gives the expression for any row in the stiffness matrix other than the first and the last row. Hence the global stiffness matrix is

$$\begin{bmatrix} \frac{2bh-3a}{6} & \frac{bh+3a}{6} & 0 & 0 & 0 & \dots & 0 \\ \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & 0 & 0 & \dots & 0 \\ 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & 0 & \dots & 0 \\ 0 & 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & 0 \\ 0 & 0 & 0 & 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} \\ 0 & 0 & 0 & 0 & 0 & \frac{bh-3a}{6} & \frac{3a+2bh}{6} \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ \vdots \\ \vdots \\ q_{N-1} \\ q_N \end{bmatrix} = \begin{bmatrix} \frac{1}{h}\int_{x=x_1}^{x=x_2}(x_2-x)\,f(x) \\ \frac{1}{h}\left(\int_{x=x_1}^{x=x_2}(x-x_1)\,f(x)dx + \int_{x=x_2}^{x=x_3}(x_3-x)\,f(x)dx\right) \\ \frac{1}{h}\left(\int_{x=x_2}^{x=x_3}(x-x_2)\,f(x)dx + \int_{x=x_3}^{x=x_4}(x_4-x)\,f(x)dx\right) \\ \vdots \\ \frac{1}{h}\left(\int_{x=x_{j-1}}^{x=x_j}\left(x-x_{j-1}\right)f(x)dx + \int_{x=x_j}^{x=x_{j+1}}\left(x_{j+1}-x\right)f(x)dx\right. \\ \vdots \\ \frac{1}{h}\left(\int_{x=x_{N-2}}^{x=x_{N-1}}(x-x_{N-2})\,f(x)dx + \int_{x=x_{N-1}}^{x=x_N}(x_N-x)\,f(x)dx\right. \\ \frac{1}{h}\int_{x=x_{N-1}}^{x=x_N}(x-x_{N-1})\,f(x) \end{bmatrix}$$

The above shows that the stiffness matrix can be build quickly without the use of any numerical integration in this example. Integration is needed to evaluate the force (or load) vector. Depending on the forcing function, this can be simple or difficult to perform.

Once the load vector is calculated, the unknowns q_i are solved for. But before doing that, q_1 is first

replaced by the initial condition given in the problem

$$\begin{bmatrix} \frac{2bh-3a}{6} & \frac{bh+3a}{6} & 0 & 0 & 0 & ... & 0 \\ \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & 0 & 0 & ... & 0 \\ 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & 0 & ... & 0 \\ 0 & 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & ... & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & 0 \\ 0 & 0 & 0 & 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} \\ 0 & 0 & 0 & 0 & 0 & \frac{bh-3a}{6} & \frac{3a+2bh}{6} \end{bmatrix} \begin{bmatrix} u_0 \\ q_2 \\ q_3 \\ \vdots \\ \vdots \\ \vdots \\ q_{N-1} \\ q_N \end{bmatrix} = \begin{bmatrix} \frac{1}{h}\int_{x=x_1}^{x=x_2}(x_2-x)\,f(x) \\ \frac{1}{h}\left(\int_{x=x_1}^{x=x_2}(x-x_1)\,f(x)dx+\int_{x=x_2}^{x=x_3}(x_3-x)\,f(x)dx\right) \\ \frac{1}{h}\left(\int_{x=x_2}^{x=x_3}(x-x_2)\,f(x)dx+\int_{x=x_3}^{x=x_4}(x_4-x)\,f(x)dx\right) \\ \vdots \\ \frac{1}{h}\left(\int_{x=x_{j-1}}^{x=x_j}\left(x-x_{j-1}\right)f(x)dx+\int_{x=x_j}^{x=x_{j+1}}\left(x_{j+1}-x\right)f(x)dx\right. \\ \vdots \\ \frac{1}{h}\left(\int_{x=x_{N-2}}^{x=x_{N-1}}(x-x_{N-2})\,f(x)dx+\int_{x=x_{N-1}}^{x=x_N}(x_N-x)\,f(x)dx\right. \\ \frac{1}{h}\int_{x=x_{N-1}}^{x=x_N}(x-x_{N-1})\,f(x) \end{bmatrix}$$

Now the first row is removed (and remembering to multiply u_0 by the first entry in the second row) gives

$$\begin{bmatrix} \frac{-3a+bh}{6}u_0 & \frac{2bh}{3} & \frac{3a+bh}{6} & 0 & 0 & ... & 0 \\ 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & 0 & ... & 0 \\ 0 & 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & ... & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & 0 \\ 0 & 0 & 0 & 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} \\ 0 & 0 & 0 & 0 & 0 & \frac{bh-3a}{6} & \frac{3a+2bh}{6} \end{bmatrix} \begin{bmatrix} q_2 \\ q_3 \\ \vdots \\ \vdots \\ \vdots \\ q_{N-1} \\ q_N \end{bmatrix} = \begin{bmatrix} \frac{1}{h}\left(\int_{x=x_1}^{x=x_2}(x-x_1)\,f(x)dx+x_3-xf(x)dx\right) \\ \frac{1}{h}\left(\int_{x=x_2}^{x=x_3}(x-x_2)\,f(x)dx+\int_{x=x_3}^{x=x_4}(x_4-x)\,f(x)dx\right) \\ \vdots \\ \frac{1}{h}\left(\int_{x=x_{j-1}}^{x=x_j}\left(x-x_{j-1}\right)f(x)dx+\int_{x=x_j}^{x=x_{j+1}}\left(x_{j+1}-x\right)f(x)\right. \\ \vdots \\ \frac{1}{h}\left(\int_{x=x_{N-2}}^{x=x_{N-1}}(x-x_{N-2})\,f(x)dx+\int_{x=x_{N-1}}^{x=x_N}(x_N-x)\,f(x)\right. \\ \frac{1}{h}\int_{x=x_{N-1}}^{x=x_N}(x-x_{N-1})\,f(x) \end{bmatrix}$$

The first column is now removed after moving the first entry in the first row to the RHS to become

part of the load vector

$$\begin{bmatrix} \frac{2bh}{3} & \frac{3a+bh}{6} & 0 & 0 & ... & 0 \\ \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & 0 & ... & 0 \\ 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & ... & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} & 0 \\ 0 & 0 & 0 & \frac{-3a+bh}{6} & \frac{2bh}{3} & \frac{3a+bh}{6} \\ 0 & 0 & 0 & 0 & \frac{bh-3a}{6} & \frac{3a+2bh}{6} \end{bmatrix} \begin{bmatrix} q_2 \\ q_3 \\ \vdots \\ \vdots \\ \vdots \\ q_{N-1} \\ q_N \end{bmatrix} = \begin{bmatrix} \frac{1}{h}\left(\int_{x=x_1}^{x=x_2} (x-x_1) f(x)dx + \int_{x=x_2}^{x=x_3} (x_3-x) f(x)dx \right) - \left(\frac{-3a+bh}{6} u_0 \right) \\ \frac{1}{h}\left(\int_{x=x_2}^{x=x_3} (x-x_2) f(x)dx + \int_{x=x_3}^{x=x_4} (x_4-x) f(x)dx \right) \\ \vdots \\ \frac{1}{h}\left(\int_{x=x_{j-1}}^{x=x_j} \left(x-x_{j-1}\right) f(x)dx + \int_{x=x_j}^{x=x_{j+1}} \left(x_{j+1}-x\right) f(x)dx \right) \\ \vdots \\ \frac{1}{h}\left(\int_{x=x_{N-2}}^{x=x_{N-1}} (x-x_{N-2}) f(x)dx + \int_{x=x_{N-1}}^{x=x_N} (x_N-x) f(x)dx \right) \\ \frac{1}{h} \int_{x=x_{N-1}}^{x=x_N} (x-x_{N-1}) f(x) \end{bmatrix}$$

Now the system above is solved for $q_2, q_3, \cdots, q_N$. Once the q_i are found, the solution is

$$u(x) = \sum_{i=1}^{Number\ Nodes} q_i \phi_i$$

The Appendix shows a Mathematica code which solve the above general first order ODE. 4 first order ODE solved for illustration and the solution is compared to the exact solution. Animation was made to help illustrate this. the RMS error was calculated. The animations can be access by clicking on the links below (shows only in the HTML version of this report).

3.2 Example Two. 2nd order ODE, Boundary value problem. Linear interpolation. Symmetric weak form.

Now the FEM formulation is given for a boundary value, second order ODE with constant coefficients of the form

$$a\frac{d^2u}{dx^2} + b\frac{du}{dx} + cu(x) = f(x)$$

with the initial conditions $u(x_0) = u_0$ (called essential Dirichlet condition), and the Neumann boundary condition $\frac{du}{dx} = \beta$ at L, for $x_0 \le x \le L$

As before, the solution is assumed to be of the form

$$u = \sum_{i=1}^{N} q_i \phi_i$$

And the unknowns q_i are found by solving N algebraic equations

$$\int_{x_0}^{L} \phi_j R(x)dx = 0 \qquad j = 1 \cdots N$$

Where $R(x)$ is the ODE residual obtained by substituting the assumed solution into the original ODE. Hence the above becomes (For simplicity, $j = 1 \cdots N$ is not written each time as it is assumed to be the case)

$$\int_{x_0}^{L} \phi_j \left(a\frac{d^2u}{dx^2} + b\frac{du}{dx} + cu(x) - f(x) \right) dx = 0$$

$$\int_{x_0}^{L} \phi_j a\frac{d^2u}{dx^2} dx + \int_{x_0}^{L} \phi_j \left(b\frac{du}{dx} + cu(x) - f(x) \right) dx = 0$$

Applying integration by parts on $\int_{x_0}^{L} \phi_j a\frac{d^2u}{dx^2} dx$. Since

$$\frac{d}{dx}\left(a\ \phi_j \frac{du}{dx} \right) = a\phi_j \frac{d^2u}{dx^2} + a\phi_j' \frac{du}{dx}$$

then integrating both sides of the above gives

$$\int_{x_0}^{L} \frac{d}{dx}\left(a\ \phi_j \frac{du}{dx} \right) dx = \int_{x_0}^{L} a\phi_j \frac{d^2u}{dx^2} dx + \int_{x_0}^{L} a\phi_j' \frac{du}{dx} dx$$

$$\left[a\phi_j \frac{du}{dx} \right]_{x_0}^{L} = \int_{x_0}^{L} a\phi_j \frac{d^2u}{dx^2} dx + \int_{x_0}^{L} a\phi_j' \frac{du}{dx} dx$$

Hence

$$\int_{x_0}^{L} a\phi_j \frac{d^2u}{dx^2} dx = \left[a\phi_j \frac{du}{dx} \right]_{x_0}^{L} - \int_{x_0}^{L} a\phi_j' \frac{du}{dx} dx$$

Substituting the above in equation A1 gives

$$\left[a\phi_j \frac{du}{dx} \right]_{x_0}^{L} - \int_{x_0}^{L} a\phi_j' \frac{du}{dx} dx + \int_{x_0}^{L} \phi_j \left(b\frac{du}{dx} + cu(x) - f(x) \right) dx = 0 \qquad \text{(A2)}$$

Considering the term

$$\left[a\phi_j \frac{du}{dx} \right]_{x_0}^{L} = a\phi_j \frac{du}{dx}\bigg|_{x=L} - a\phi_j \frac{du}{dx}\bigg|_{x=x_0}$$

First considering the term $a\phi_j \frac{du}{dx}\Big|_{x=L}$. Since $\frac{du}{dx} = \beta$ at $x = L$ then this term becomes $a\beta \left[\phi_j\right]_{x=L}$. But $\left[\phi_j\right]_{x=L}$ is non zero only for the N^{th} shape function evaluated at $x = L$. Since linear interpolation is used, the N^{th} shape function is $\phi_N = \frac{x - x_{N-1}}{h}$ which have the value of 1 at $x = x_N$. Hence

$$a\phi_j \frac{du}{dx}\bigg|_{x=L} = a\beta \qquad \text{when} j = N, \text{otherwise}, 0$$

Now considering the term $a\phi_j \frac{du}{dx}\Big|_{x=x_0}$, since at $x = x_0$ all shape functions will be zero except for ϕ_1 which has the value of 1 at $x = x_0$. Hence this simplifies to $a\frac{du}{dx}\Big|_{x=x_0}$

25

Recalling that $\frac{du}{dx} = \frac{d}{dx}\sum_{i=1}^{N} q_i\phi_i = \sum_{i=1}^{N} q_i\phi_i'$. But at $x = x_0$ only ϕ_1 is defined and its has the slope of $\frac{-1}{h}$, hence

$$a\frac{du}{dx}\bigg|_{x=x_0} = \frac{-aq_1}{h} \text{ when } j = 1, \text{ otherwise } 0$$

However, $q_1 = u_0$ since that is by definition the initial condition. Finally one obtains

$$\left[a\phi_j\frac{du}{dx}\right]_{x_0}^{L} = \left\{ \begin{array}{ll} \frac{aq_1}{h} & j = 1 \\ a\beta & j = N \end{array} \right\}$$

Hence the symmetric weak form equation (A2) can now be simplified more giving

$$\left\{ \begin{array}{ll} \frac{aq_1}{h} & j = 1 \\ a\beta & j = N \end{array} \right\} - \int_{x_0}^{L} a\phi_j'\frac{du}{dx}dx + \int_{x_0}^{L} \phi_j\left(b\frac{du}{dx} + cu(x) - f(x)\right)dx = 0 \qquad j = 1\cdots N \tag{A3}$$

The trial function obtain by linear interpolation are used. These are shown in this table

i	ϕ_i	ϕ_i'
1	$\frac{(x_2-x)}{h}$	$\frac{-1}{h}$
$1 < i < N$	$\left\{\frac{(x-x_{i-1})}{h}, \frac{(x_{i+1}-x)}{h}\right\}$	$\left\{\frac{1}{h}, \frac{-1}{h}\right\}$
N	$\frac{(x-x_{N-1})}{h}$	$\frac{1}{h}$

The global stiffness matrix is now constructed. For the first equation (which corresponds to the first row in the global stiffness matrix) the result is

$$\frac{aq_1}{h} - \int_{x_0}^{L} a\phi_1'\frac{du}{dx}dx + \int_{x_0}^{L} \phi_1\left(b\frac{du}{dx} + cu\,(x) - f(x)\right)dx = 0$$

Since $u = \sum_{i=1}^{N} q_i\phi_i$, and since ϕ_1 domain of influence is only from x_1 to x_2 then above becomes

$$\frac{aq_1}{h} - \int_{x_1}^{x_2} a\phi_1'\frac{d}{dx}\left(q_1\phi_1 + q_2\phi_2\right)dx + \int_{x_1}^{x2} \phi_1\left(b\frac{d}{dx}\left(q_1\phi_1 + q_2\phi_2\right) + c\left(q_1\phi_1 + q_2\phi_2\right) - f(x)\right)dx = 0$$

But $\phi_1 = \frac{(x_2-x)}{h}$ and $\phi_1' = \frac{-1}{h}$ and over the domain from x_1 to x_2, $\phi_2 = \frac{(x-x_1)}{h}, \phi_2' = \frac{1}{h}$, hence the above becomes

$$\frac{aq_1}{h} - \int_{x_1}^{x_2} \frac{-a}{h}\left(\frac{-q_1}{h} + \frac{q_2}{h}\right)dx + \int_{x_1}^{x2} \frac{(x_2-x)}{h}\left(b\left(\frac{-q_1}{h} + \frac{q_2}{h}\right) + c\left(q_1\frac{(x_2-x)}{h} + q_2\frac{(x-x_1)}{h}\right) - f(x)\right)dx = 0$$

The above simplifies to

$$\frac{aq_1}{h} - \frac{q_1}{h^2}\left(ax_1 - \frac{b(x_1-x_2)^2}{2} - \frac{c(x_1-x_2)^3}{3} - ax_2\right) + \frac{q_2}{h^2}\left(-ax_1 + \frac{b\,(x_1-x_2)^2}{2} - \frac{c\,(x_1-x_2)^3}{6} + ax_2\right) - \frac{1}{h}\int_{x_1}^{x2}(x_2-x)\,f(x) +$$

$$-\frac{q_1}{h^2}\left(-ah + ax_1 - \frac{b\,(x_1-x_2)^2}{2} - \frac{c\,(x_1-x_2)^3}{3} - ax_2\right) + \frac{q_2}{h^2}\left(-ax_1 + \frac{b\,(x_1-x_2)^2}{2} - \frac{c\,(x_1-x_2)^3}{6} + ax_2\right) - \frac{1}{h}\int_{x_1}^{x2}(x_2-x)\,f(x) +$$

The above gives the first row in the stiffness matrix. Since it is assumed that each element will have the same length, hence $x_1 - x_2 = -h$, and the above becomes

$$-\frac{q_1}{h^2}\left(-ah + ax_1 - \frac{bh^2}{2} + \frac{ch^3}{3} - ax_2\right) + \frac{q_2}{h^2}\left(-ax_1 + \frac{bh^2}{2} + \frac{ch^3}{6} + ax_2\right) - \frac{1}{h}\int_{x_1}^{x2}(x_2 - x)\, f(x) = 0$$

$$-q_1\left(-\frac{a}{h} + \frac{ax_1}{h^2} - \frac{b}{2} + \frac{ch}{3} - \frac{ax_2}{h^2}\right) + q_2\left(-\frac{ax_1}{h^2} + \frac{b}{2} + \frac{ch}{6} + \frac{ax_2}{h^2}\right) - \frac{1}{h}\int_{x_1}^{x2}(x_2 - x)\, f(x) = 0$$

For the last equation, which will be the last row in the global stiffness matrix

$$a\beta - \int_{x_{N-1}}^{x_N} a\phi'_N \frac{du}{dx}dx + \int_{x_{N-1}}^{x_N} \phi_N\left(b\frac{du}{dx} + cu(x) - f(x)\right)dx = 0$$

Since $u = \sum_{i=1}^{N} q_i\phi_i$, and since ϕ_N domain of influence is only from x_{N-1} to x_N The above becomes

$$a\beta - \int_{x_{N-1}}^{x_N} a\phi'_N \frac{d}{dx}\left(q_{N-1}\phi_{N-1} + q_N\phi_N\right)dx + \int_{x_{N-1}}^{x_N} \phi_N\left(b\frac{d}{dx}\left(q_{N-1}\phi_{N-1} + q_N\phi_N\right) + c\left(q_{N-1}\phi_{N-1} + q_N\phi_N\right) - f(x)\right)dx = 0$$

But $\phi_N = \frac{(x - x_{N-1})}{h}$ and $\phi'_N = \frac{1}{h}$ and over the domain from x_{N-1} to x_N, $\phi_{N-1} = \frac{(x_N - x)}{h}, \phi'_{N-1} = \frac{-1}{h}$ hence the above becomes

$$a\beta - \int_{x_{N-1}}^{x_N} \frac{a}{h}\left(q_{N-1}\left(\frac{-1}{h}\right) + q_N\frac{1}{h}\right)dx + \int_{x_{N-1}}^{x_N} \frac{(x - x_{N-1})}{h}\left(b\left(q_{N-1}\left(\frac{-1}{h}\right) + q_N\frac{1}{h}\right) + c\left(q_{N-1}\frac{(x_N - x)}{h} + q_N\frac{(x - x_{N-1})}{h}\right) - f(x)\right)dx =$$

The above simplifies to

$$a\beta - \frac{q_{N-1}}{h^2}\left(-ax_{N-1} - \frac{b(x_{N-1} - x_N)^2}{2} - \frac{c(x_{N-1} - x_N)^3}{6} + ax_N\right) + \frac{q_N}{h^2}\left(+ax_{N-1} + \frac{b(x_{N-1} - x_N)^2}{2} - \frac{c(x_{N-1} - x_N)^3}{3} - ax_N\right) - \frac{1}{h}$$

The above represents the last row in the global stiffness matrix. Since it is assumed that each element will have the same length, hence $x_{N-1} - x_N = -h$, and the above becomes

$$a\beta - \frac{q_{N-1}}{h^2}\left(-ax_{N-1} - \frac{bh^2}{2} + \frac{ch^3}{6} + ax_N\right) + \frac{q_N}{h^2}\left(ax_{N-1} + \frac{bh^2}{2} + \frac{ch^3}{3} - ax_N\right) - \frac{1}{h}\int_{x_{N-1}}^{x_N}(x - x_{N-1})\, f(x) = 0$$

$$a\beta - q_{N-1}\left(-\frac{ax_{N-1}}{h^2} - \frac{b}{2} + \frac{ch}{6} + \frac{ax_N}{h^2}\right) + q_N\left(\frac{ax_{N-1}}{h^2} + \frac{b}{2} + \frac{ch}{3} - \frac{ax_N}{h^2}\right) - \frac{1}{h}\int_{x_{N-1}}^{x_N}(x - x_{N-1})\, f(x) = 0$$

The expression for any row in between the first and the last rows is now found. For a general node j the result is

$$-\int_{x_{j-1}}^{x_{j+1}} a\phi'_j \frac{du}{dx}dx + \int_{x_{j-1}}^{x_{j+1}} \phi_j\left(b\frac{du}{dx} + cu(x) - f(x)\right)dx = 0$$

27

Breaking the integral into halves to make it easier to write the trial functions over the domain of influence gives

$$-\left(\int_{x_{j-1}}^{x_j} a\phi_j'\frac{du}{dx}dx+\int_{x_j}^{x_{j+1}} a\phi_j'\frac{du}{dx}dx\right)+\int_{x_{j-1}}^{x_j}\phi_j\left(b\frac{du}{dx}+cu(x)-f(x)\right)dx+\int_{x_j}^{x_{j+1}}\phi_j\left(b\frac{du}{dx}+cu(x)-f(x)\right)dx=0$$

Considering the first domain $x_{j-1}\le x\le x_j$ gives

$$-\int_{x_{j-1}}^{x_j} a\phi_j'\frac{du}{dx}dx+\int_{x_{j-1}}^{x_j}\phi_j\left(b\frac{du}{dx}+cu(x)-f(x)\right)dx$$

Over this range, $u=q_{j-1}\phi_{j-1}+q_j\phi_j$ hence $\frac{du}{dx}=q_{j-1}\phi_{j-1}'+q_j\phi_j'$ where $\phi_{j-1}=\frac{x_j-x}{h},\phi_{j-1}'=\frac{-1}{h},\phi_j=\frac{x-x_{j-1}}{h},\phi_j'=\frac{1}{h}$ hence the above becomes

$$-\int_{x_{j-1}}^{x_j} a\frac{1}{h}\left(q_{j-1}\left(\frac{-1}{h}\right)+q_j\frac{1}{h}\right)dx+\int_{x_{j-1}}^{x_j}\frac{x-x_{j-1}}{h}\left(b\left(q_{j-1}\left(\frac{-1}{h}\right)+q_j\frac{1}{h}\right)+c\left(q_{j-1}\frac{x_j-x}{h}+q_j\frac{x-x_{j-1}}{h}\right)-f(x)\right)dx \tag{A4}$$

Now considering the second domain $x_j\le x\le x_{j+1}$ gives

$$-\int_{x_j}^{x_{j+1}} a\phi_j'\frac{du}{dx}dx+\int_{x_j}^{x_{j+1}}\phi_j\left(b\frac{du}{dx}+cu(x)-f(x)\right)dx$$

Over this range, $u=q_j\phi_j+q_{j+1}\phi_{j+1}$ hence $\frac{du}{dx}=q_j\phi_j'+q_{j+1}\phi_{j+1}'$ where $\phi_j=\frac{x_{j+1}-x}{h},\phi_j'=\frac{-1}{h},\phi_{j+1}=\frac{x-x_j}{h},\phi_{j+1}'=\frac{1}{h}$ hence the above becomes

$$-\int_{x_j}^{x_{j+1}}\left(\frac{-a}{h}\right)\left(\frac{-q_j}{h}+\frac{q_{j+1}}{h}\right)dx+\int_{x_j}^{x_{j+1}}\frac{x_{j+1}-x}{h}\left(b\left(\frac{-q_j}{h}+q_{j+1}\frac{1}{h}\right)+c\left(q_j\frac{x_{j+1}-x}{h}+q_{j+1}\frac{x-x_j}{h}\right)-f(x)\right)dx$$

Combine A4 and A5 and simplifying gives

$$\frac{q_{j-1}}{h^2}\left(-ax_{j-1}-\frac{b\left(x_{j-1}-x_j\right)^2}{2}-\frac{c\left(x_{j-1}-x_j\right)^3}{6}+ax_j\right)+\frac{q_j}{h^2}\left(ax_{j-1}+\frac{b\left(x_{j-1}-x_j\right)^2}{2}-\frac{c\left(x_{j-1}-x_j\right)^3}{3}-\frac{b\left(x_j-x_{j+1}\right)^2}{2}-\frac{c\left(x_j-\right.}{}\right.$$

$$+\frac{q_{j+1}}{h^2}\left(-ax_j+\frac{b\left(x_j-x_{j+1}\right)^2}{2}-\frac{c\left(x_j-x_{j+1}\right)^3}{6}+ax_{j+1}\right)-\int_{x_{j-1}}^{x_j}\frac{x-x_{j-1}}{h}f(x)dx-\int_{x_j}^{x_{j+1}}\frac{x_{j+1}-x}{h}dx=0$$

Since we are assuming each element will have the same length, hence $x_{j-1}-x_j=-h, x_j-x_{j+1}=-h$ and the above becomes

$$q_{j-1}\left(-\frac{ax_{j-1}}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_j}{h^2}\right)+q_j\left(\frac{ax_{j-1}}{h^2}+\frac{2ch}{3}-\frac{ax_{j+1}}{h^2}\right)$$

$$+q_{j+1}\left(\frac{-ax_j}{h^2}+\frac{b}{2}+\frac{ch}{6}+\frac{ax_{j+1}}{h^2}\right)-\int_{x_{j-1}}^{x_j}\frac{x-x_{j-1}}{h}f(x)dx-\int_{x_j}^{x_{j+1}}\frac{x_{j+1}-x}{h}f(x)dx=0$$

28

The above gives any row in the stiffness matrix other than the first and the last row. Now we can write the global stiffness matrix as

$$\begin{bmatrix} \left(-\frac{a}{h}+\frac{ax_1}{h^2}-\frac{b}{2}+\frac{ch}{3}-\frac{ax_2}{h^2}\right) & \left(-\frac{ax_1}{h^2}+\frac{b}{2}-\frac{ch}{6}+\frac{ax_2}{h^2}\right) & 0 & 0 \\ \left(-\frac{ax_1}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_2}{h^2}\right) & \left(\frac{ax_1}{h^2}+\frac{2ch}{3}-\frac{ax_3}{h^2}\right) & \left(\frac{-ax_2}{h^2}+\frac{b}{2}+\frac{ch}{6}+\frac{ax_3}{h^2}\right) & 0 \\ 0 & \left(-\frac{ax_{j-1}}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_j}{h^2}\right) & \left(\frac{ax_{j-1}}{h^2}+\frac{2ch}{3}-\frac{ax_{j+1}}{h^2}\right) & \left(\frac{-ax_j}{h^2}+\frac{b}{2}+\frac{ch}{6}+\frac{ax_{j+1}}{h^2}\right) \\ 0 & 0 & \ddots & \cdots \\ 0 & 0 & \left(-\frac{ax_{N-1}}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_N}{h^2}\right) & \left(\frac{ax_{N-1}}{h^2}+\frac{b}{2}+\frac{ch}{3}-\frac{ax_N}{h^2}\right) \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ \vdots \\ q_j \\ \vdots \\ q_{N-1} \\ q_N \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{h}\int_{x_1}^{x2}(x_2-x)\,f(x) \\ \vdots \\ \vdots \\ \vdots \\ \frac{1}{h}\int_{x_{j-1}}^{x_j}\left(x-x_{j-1}\right)f(x)dx+\frac{1}{h}\int_{x_j}^{x_{j+1}}\left(x_{j+1}-x\right)f(x)dx \\ \vdots \\ \vdots \\ \frac{1}{h}\int_{x_{N-1}}^{x_N}(x-x_{N-1})\,f(x)-a\beta \end{bmatrix}$$

We must first replace q_1 by the initial condition given in the problem, and we must remove the

first row after that as follows

$$\begin{bmatrix} \left(-\frac{a}{h}+\frac{ax_1}{h^2}-\frac{b}{2}+\frac{ch}{3}-\frac{ax_2}{h^2}\right) & \left(-\frac{ax_1}{h^2}+\frac{b}{2}-\frac{ch}{6}+\frac{ax_2}{h^2}\right) & 0 & 0 \\ \left(-\frac{ax_1}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_2}{h^2}\right) & \left(\frac{ax_1}{h^2}+\frac{2ch}{3}-\frac{ax_3}{h^2}\right) & \left(\frac{-ax_2}{h^2}+\frac{b}{2}+\frac{ch}{6}+\frac{ax_3}{h^2}\right) & 0 \\ 0 & 0 & \ddots & \cdots \\ 0 & 0 & \ddots & \cdots \\ 0 & 0 & \left(-\frac{ax_{N-1}}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_N}{h^2}\right) & \left(\frac{ax_{N-1}}{h^2}+\frac{b}{2}+\frac{ch}{3}-\frac{ax_N}{h^2}\right) \end{bmatrix} \begin{bmatrix} u_0 \\ q_2 \\ \vdots \\ \vdots \\ q_j \\ \vdots \\ q_{N-1} \\ q_N \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{h}\int_{x_1}^{x2}(x_2-x)\,f(x) \\ \vdots \\ \vdots \\ \vdots \\ \frac{1}{h}\int_{x_{j-1}}^{x_j}\left(x-x_{j-1}\right)f(x)dx+\frac{1}{h}\int_{x_j}^{x_{j+1}}\left(x_{j+1}-x\right)f(x)dx \\ \vdots \\ \vdots \\ \frac{1}{h}\int_{x_{N-1}}^{x_N}(x-x_{N-1})\,f(x)-a\beta \end{bmatrix}$$

30

Multiply the first element in the second row by u_0 and removing the first row gives

$$\begin{bmatrix} \left(-\frac{ax_1}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_2}{h^2}\right)u_0 & \left(\frac{ax_1}{h^2}+\frac{2ch}{3}-\frac{ax_3}{h^2}\right) & \left(\frac{-ax_2}{h^2}+\frac{b}{2}+\frac{ch}{6}+\frac{ax_3}{h^2}\right) & 0 \\ 0 & \left(-\frac{ax_2}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_3}{h^2}\right) & \left(\frac{ax_2}{h^2}+\frac{2ch}{3}-\frac{ax_4}{h^2}\right) & \left(\frac{-ax_3}{h^2}+\frac{b}{2}+\frac{ch}{6}+\frac{ax_4}{h^2}\right) \\ 0 & 0 & \ddots & \ddots \\ 0 & 0 & \left(-\frac{ax_{N-1}}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_N}{h^2}\right) & \left(\frac{ax_{N-1}}{h^2}+\frac{b}{2}+\frac{ch}{3}-\frac{ax_N}{h^2}\right) \end{bmatrix} \begin{bmatrix} q_2 \\ \vdots \\ \vdots \\ q_j \\ \vdots \\ q_{N-1} \\ q_N \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{h}\int_{x_1}^{x_2}(x-x_2)\,f(x)dx + \frac{1}{h}\int_{x_2}^{x_3}(x_3-x)\,dx \\ \vdots \\ \vdots \\ \frac{1}{h}\int_{x_{j-1}}^{x_j}\left(x-x_{j-1}\right)f(x)dx + \frac{1}{h}\int_{x_j}^{x_{j+1}}\left(x_{j+1}-x\right)f(x)dx \\ \vdots \\ \vdots \\ \frac{1}{h}\int_{x_{N-1}}^{x_N}(x-x_{N-1})\,f(x) - a\beta \end{bmatrix}$$

Now moving the first element of the first row above to the RHS and removing the first column

gives

$$\begin{bmatrix} \left(\frac{ax_1}{h^2}+\frac{2ch}{3}-\frac{ax_3}{h^2}\right) & \left(\frac{-ax_2}{h^2}+\frac{b}{2}+\frac{ch}{6}+\frac{ax_3}{h^2}\right) & 0 & 0 \\ 0 & \left(-\frac{ax_2}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_3}{h^2}\right) & \left(\frac{ax_2}{h^2}+\frac{2ch}{3}-\frac{ax_4}{h^2}\right) & \left(\frac{-ax_j}{h^2}+\frac{b}{2}+\frac{ch}{6}+\frac{ax_{j+1}}{h^2}\right) \\ \vdots & \ddots & \cdots & \ddots \\ 0 & 0 & \left(-\frac{ax_{N-1}}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_N}{h^2}\right) & \left(\frac{ax_{N-1}}{h^2}+\frac{b}{2}+\frac{ch}{3}-\frac{ax_N}{h^2}\right) \end{bmatrix} \begin{bmatrix} q_2 \\ \vdots \\ \vdots \\ q_j \\ \vdots \\ q_{N-1} \\ q_N \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{h}\int_{x_1}^{x_2}(x-x_2)\,f(x)dx + \frac{1}{h}\int_{x_2}^{x_3}(x_3-x)\,dx - \left(-\frac{ax_1}{h^2}-\frac{b}{2}+\frac{ch}{6}+\frac{ax_2}{h^2}\right)u_0 \\ \vdots \\ \vdots \\ \frac{1}{h}\int_{x_{j-1}}^{x_j}\left(x-x_{j-1}\right)f(x)dx + \frac{1}{h}\int_{x_j}^{x_{j+1}}\left(x_{j+1}-x\right)f(x)dx \\ \vdots \\ \vdots \\ \frac{1}{h}\int_{x_{N-1}}^{x_N}(x-x_{N-1})\,f(x) - a\beta \end{bmatrix}$$

Now we solve for the vector $\left[q_2, q_3, \cdots, q_N\right]^T$ and this completes our solution.

A Matlab implementation is below which solves any second order ODE. This code was used to generate an animation. This animation can be accessed by clicking on the link below.

4 References

1. Methods of computer modeling in engineering and the sciences. Volume 1. By Professor Satya N. Atluri. Tech Science Press.
2. Class lecture notes. MAE 207. Computational methods. UCI. Spring 2006. Instructor: Professor SN Atluri.
3. Computational techniques for fluid dynamics, Volume I. C.A.J.Fletcher. Springer-Verlag

2.1.3 Solving $u'''' + u = 1$ by collocation method using Mathematica

```
In[41]:= Remove["Global`*"]

In[42]:= (* This code solves u''''+u=1 by Collocation. By Nasser M. Abbasi*)

         p       = 1; (*right hand side of ode*)
         nBasis  = 5;
         nPoints = 5;
         coeff   = Table[a_n, {n, 1, nBasis}]
         basisFunction[x_, n_] := coeff[[n]] (x (x - 1))^n
         u[x_]     := Sum_{n=1}^{nBasis} basisFunction[x, n]
         error[x_] := u''''[x] + u[x] - p
         A = Table[error[x] /. x -> {n/(2 nPoints - 1)}, {n, 1, nPoints - 1}];
         A = Flatten[Append[A, {a_1 + a_2}]];

Out[44]= {a_1, a_2, a_3, a_4, a_5}

In[50]:= Print["Set of equations to solve are "]
         A // MatrixForm
         N[%] // MatrixForm
```

Out[51]//MatrixForm=

$$\begin{pmatrix} -1 - \frac{8\,a_1}{81} + \frac{157\,528\,a_2}{6561} + \frac{19\,367\,560\,a_3}{531\,441} - \frac{302\,169\,320\,a_4}{43\,046\,721} - \frac{1\,114\,877\,888\,a_5}{3\,486\,784\,401} \\ -1 - \frac{14\,a_1}{81} + \frac{157\,660\,a_2}{6561} + \frac{5\,193\,568\,a_3}{531\,441} - \frac{377\,717\,720\,a_4}{43\,046\,721} + \frac{11\,936\,808\,016\,a_5}{3\,486\,784\,401} \\ -1 - \frac{2\,a_1}{9} + \frac{1948\,a_2}{81} - \frac{5840\,a_3}{729} + \frac{1960\,a_4}{6561} + \frac{58\,288\,a_5}{59\,049} \\ -1 - \frac{20\,a_1}{81} + \frac{157\,864\,a_2}{6561} - \frac{8\,983\,448\,a_3}{531\,441} + \frac{340\,439\,704\,a_4}{43\,046\,721} - \frac{10\,726\,498\,400\,a_5}{3\,486\,784\,401} \\ a_1 + a_2 \end{pmatrix}$$

Out[52]//MatrixForm=

$$\begin{pmatrix} -1. - 0.0987654\,a_1 + 24.0098\,a_2 + 36.4435\,a_3 - 7.01957\,a_4 - 0.319744\,a_5 \\ -1. - 0.17284\,a_1 + 24.0299\,a_2 + 9.77261\,a_3 - 8.7746\,a_4 + 3.42344\,a_5 \\ -1. - 0.222222\,a_1 + 24.0494\,a_2 - 8.01097\,a_3 + 0.298735\,a_4 + 0.987112\,a_5 \\ -1. - 0.246914\,a_1 + 24.061\,a_2 - 16.9039\,a_3 + 7.90861\,a_4 - 3.07633\,a_5 \\ a_1 + a_2 \end{pmatrix}$$

```
In[53]:= Solve[A == 0, coeff]
         N[%]
```

Out[53]=

$$\left\{\left\{a_1 \to -\frac{2\,179\,796\,858\,395\,129\,608\,273}{52\,844\,426\,917\,618\,938\,573\,842},\ a_2 \to \frac{2\,179\,796\,858\,395\,129\,608\,273}{52\,844\,426\,917\,618\,938\,573\,842},\ a_3 \to \frac{15\,566\,753\,176\,631\,152\,407}{105\,688\,853\,835\,237\,877\,147\,684},\ a_4 \to -\frac{5\,183\,668\,474\,801\,954\,587}{211\,377\,707\,670\,475\,754\,295\,368},\ a_5 \to -\frac{13\,877\,316\,209\,958\,489}{422\,755\,415\,340\,951\,508\,590\,736}\right\}\right\}$$

Out[54]= $\left\{\left\{a_1 \to -0.0412493,\ a_2 \to 0.0412493,\ a_3 \to 0.000147289,\ a_4 \to -0.0000245233,\ a_5 \to -3.28259 \times 10^{-8}\right\}\right\}$

Printed by Wolfram Mathematica Student Edition

2.1.4 report on the FEM solution to torsion problem of a rectangular cross section

Solving the torsion problem for isotropic matrial with a rectangular cross section using the FEM and FVM methods with triangular elements MAE 207, Computational methods. UCI Fall 2006

Nasser M. Abbasi

2006 Compiled on September 4, 2021 at 6:34pm

Contents

1 Introduction

We consider bar made of isotropic martial with rectangular cross section subjected to twisting torque T. The following diagram illustrate the basic geometry.

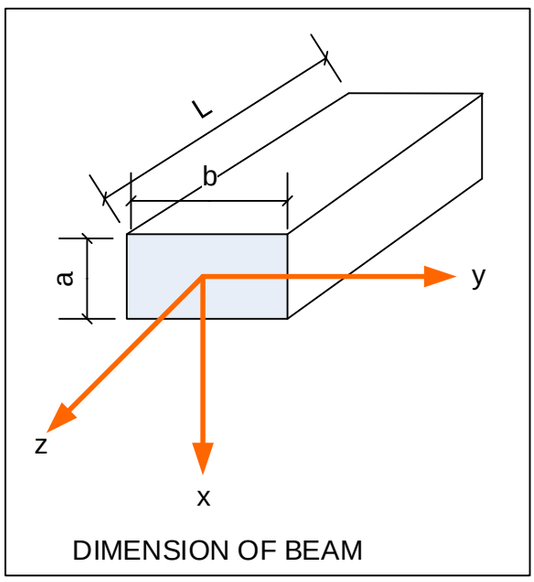


Figure 1: basic geometry

Experiments show that rectangular cross sections do wrap and that cross sections do not remain plane as shown in this diagram (in the case of a circular cross section, cross section do NOT wrap).

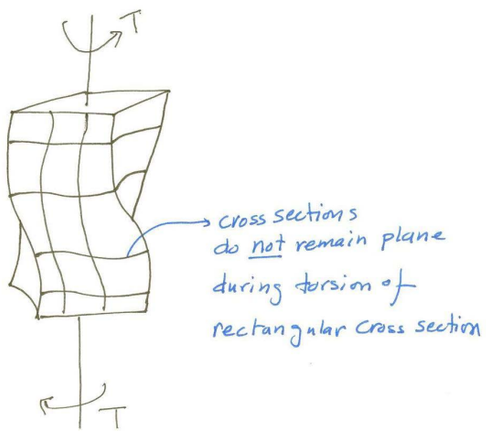


Figure 2: cross section wrap

This is another diagram showing a bar under torsion

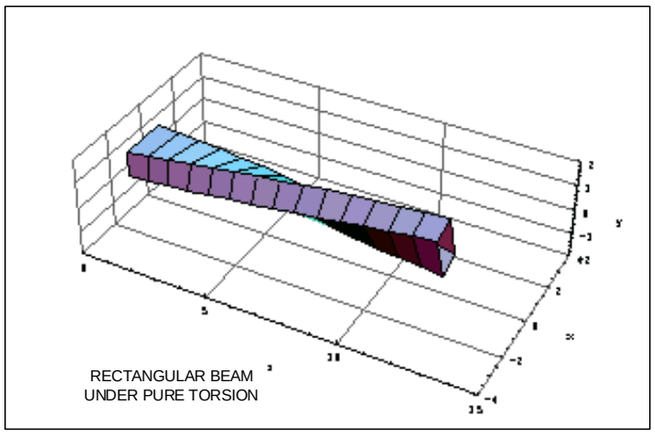


Figure 3: bar under torsion

1.1 Problem setup

1.1.1 What are the assumptions?

1. The twist rate (called k in this problem) and defined as $\frac{d\alpha}{dz}$ where α is the twist angle is assumed to be constant.
2. Cross section can wrap also in the z direction (i.e. the cross section does not have to remain in the xy plane) but if this happens, all cross sections will wrap in the z section by the same amount.
3. Material is isotropic

1.1.2 What is the input and what is the output?

The input to the problem are the following (these are the known or given):

1. The width b and height a of the cross section.
2. Material Modulus of rigidity or sheer modulus G which is the ratio of the shearing stress τ to the shearing strain γ
3. The applied torque T
4. J the torsion constant for the a rectangular cross section. For a rectangular section of dimensions a, b it is given by

$$J = \frac{16}{3} a^3 b \left(1 - \frac{192\, a}{b\pi^5} \sum_{n=1,3,5\ldots}^{\infty} \frac{1}{n^5} \tanh\left(\frac{n\, \pi b}{2\, a} \right) \right) \tag{1}$$

Hence the torsional rigidity GJ is known since G is given (material) and J is from above (geometry).

1.1.3 The output from the problem (the things we need to calculate)

1. The stress distribution in the cross section (stress tensor field). Once this is found then using the material constitutive relation we can the strain tensor field.
2. The angle of twist α as a function of z (the length of the beam).

2 Analytical solution using Prandtl stress function

First we solve for the Prandtl stress function $\Phi\left(x,y\right)$ by solving the Poisson equation

$$\nabla^2\Phi\left(x,y\right) = -2GK$$

Where G is the sheer modulus and k is the twist rate (which was assumed to be constant).

The boundary conditions ($\Phi\left(x,y\right)$ at any point on the edge of the cross section and at the ends of the beam) is an arbitrary constant. We take this constant to be zero. Hence at the cross section boundary we have

$$\Phi = 0$$

The analytical solution to the above equation is from book Theory of elasticity by S. P. Timoshenko and J. N. Goodier

$$\Phi\left(x,y\right) = \frac{32\ Gk\ a^2}{\pi^3}\sum_{n=1,3,5,\cdots}^{\infty}\frac{1}{n^3}\left(-1\right)^{\frac{(n-1)}{2}}\left(1-\frac{\cosh\left(\frac{n\pi y}{2a}\right)}{\cosh\left(\frac{n\pi b}{2a}\right)}\right)\cos\left(\frac{n\pi x}{2a}\right) \tag{2}$$

where the linear twist k

$$k = \frac{T}{GJ}$$

Hence (2) becomes

$$\Phi\left(x,y\right) = \frac{32\ T\ a^2}{J\pi^3}\sum_{n=1,3,5,\cdots}^{\infty}\frac{1}{n^3}\left(-1\right)^{\frac{(n-1)}{2}}\left(1-\frac{\cosh\left(\frac{n\pi y}{2a}\right)}{\cosh\left(\frac{n\pi b}{2a}\right)}\right)\cos\left(\frac{n\pi x}{2a}\right)$$

Where J is given by (1)

6

2.1 Stress components

$$\tau_{yz} = -\frac{\partial \Phi}{\partial x}$$
$$\tau_{xz} = \frac{\partial \Phi}{\partial y}$$
$$\tau_{yx} = 0$$
$$\sigma_x = \sigma_y = \sigma_z = 0$$

Hence

$$\tau_{yz} = -\frac{\partial \Phi}{\partial x} = \frac{16Gka}{\pi^2} \sum_{n=1,3,5,\cdots}^{\infty} \frac{1}{n^2} (-1)^{\frac{(n-1)}{2}} \left(1 - \frac{\cosh\left(\frac{n\pi y}{2a}\right)}{\cosh\left(\frac{n\pi b}{2a}\right)}\right) \sin\left(\frac{n\pi x}{2a}\right)$$

and

$$\tau_{xz} = \frac{\partial \Phi}{\partial y} = \frac{16\ T\ a}{J\pi^2} \sum_{n=1,3,5,\cdots}^{\infty} \frac{1}{n^2} (-1)^{\frac{(n-1)}{2}} \left(-\cos\left(\frac{n\pi x}{2a}\right) sech\left(\frac{bn\pi}{2a}\right) \sinh\left(\frac{n\pi y}{2a}\right)\right)$$

Timoshenko gives the maximum sheer stress, which is $\sqrt{\tau_{yz}^2 + \tau_{xz}^2}$ as

$$\tau_{\max} = \frac{16Gka}{\pi^2} \sum_{n=1,3,5,\cdots}^{\infty} \frac{1}{n^2} \left(1 - \frac{1}{\cosh\left(\frac{n\pi b}{2a}\right)}\right)$$

2.2 Strain components

Given that E is Young's modulus for the material, υ is Poisson's ratio for the material, and $G = \frac{E}{2(1+\upsilon)}$ we can now obtain the strain components from the constitutive equations (stress-strain equations) since we have determined the stress components from the above solution.

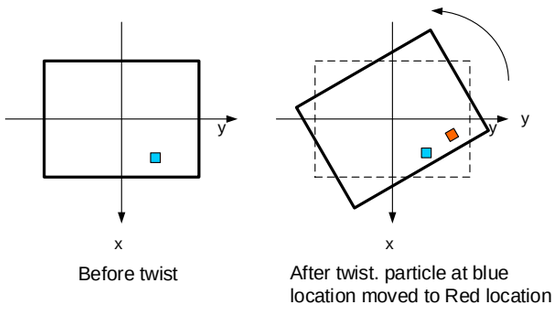
$$\varepsilon_x = \frac{1}{E}\left(\sigma_x - \upsilon\left(\sigma_y + \sigma_z\right)\right) = 0$$
$$\varepsilon_y = \frac{1}{E}\left(\sigma_y - \upsilon\left(\sigma_x + \sigma_z\right)\right) = 0$$
$$\varepsilon_z = \frac{1}{E}\left(\sigma_z - \upsilon\left(\sigma_x + \sigma_y\right)\right) = 0$$
$$\gamma_{xy} = \frac{2\left(1+\upsilon\right)}{E}\tau_{xy} = \frac{1}{G}\tau_{xy} = 0$$
$$\gamma_{yz} = \frac{2\left(1+\upsilon\right)}{E}\tau_{yz} = \frac{1}{G}\tau_{yz}$$
$$\gamma_{xz} = \frac{2\left(1+\upsilon\right)}{E}\tau_{xz} = \frac{1}{G}\tau_{xz}$$

Hence only γ_{yz} and γ_{xz} are non-zero.

2.3 Determining the twist angle α

If we look at a cross section of the bar at some distance z from the end of the bar, the angle that this specific cross section has twisted due to the torque is α.

8



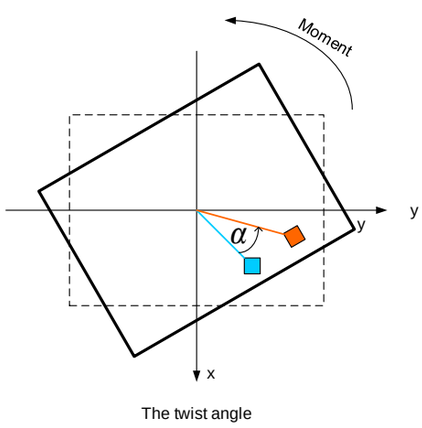


Figure 4: twist angles

This angle is given by the solution to the equation

$$\frac{d\alpha(z)}{dz} = k$$

But k is the linear twist and is given by $k = \frac{T}{GJ}$ hence the above equation becomes

$$\frac{d\alpha(z)}{dz} = \frac{T}{GJ}$$

Hence

$$\alpha(z) = \frac{T}{GJ}z + C_1$$

Where C_1 is the constant of integration. Assuming $\alpha = 0$ at $z = 0$ we obtain that

$$\alpha(z) = \frac{T}{GJ}z$$

and using the expression J given in equation (1) above we can determine α for each z.

2.4 Displacement calculations

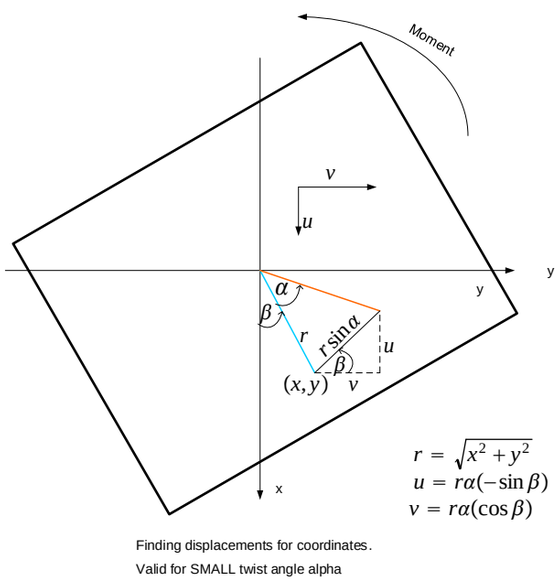


Figure 5: Displacement calculations

$$r = \sqrt{x^2 + y^2}$$
$$u = r\alpha\left(-\sin\beta\right)$$
$$v = r\alpha\left(\cos\beta\right)$$

we see that

$$\sin\beta = \frac{y}{r}$$
$$\cos\beta = \frac{x}{r}$$

Hence

$$u = -\alpha y$$
$$v = \alpha x$$

Where $\alpha = \frac{T}{GJ} z$

3 References

1. Mathematica Structural Mechanics help page
2. MIT course 16.20 lecture notes. MIT open course website.
3. Theory of elasticity by S. P. Timoshenko and J. N. Goodier. chapter 10

2.1.5 Comparing analytical solution with the FEM solution

On verification of FEM solution to Poisson equation by comparing it to the analytical solution shown in Timoshenko MAE 207, Computational methods. UCI Spring 2006

Nasser M. Abbasi

May 28, 2006 Compiled on September 4, 2021 at 7:57pm

Contents

1

1 Introduction

A Matlab function was written which plots the analytical solution given by Timoshenko/-Goodier on pages 310-311 in the `Theory of Elasticity`. The analytical solution was compared to the solution generated from the FEM solution. See this page for more information and background on the problem and the analytical solution.

The absolute and percentage differences between the solutions was obtained and compared.

The matlab code used was provided thanks to Qing Wang which solves the problem by FEM. It was called to obtain the FEM solution. Minor changes made in the code to allow one to call it as a function and to use the same contour levels. The appendix contains the Matlab code.

2 Conclusion

The solution by FEM agrees to a very good approximation with the analytical solution.

21 by 41 nodes were used for FEM, we see that, in absolute value, the maximum difference was 0.00009481788675562430 At nodes (11, 10) with symmetry at the other half of the cross section as shown in the plot below (The plot is here on a separate page)

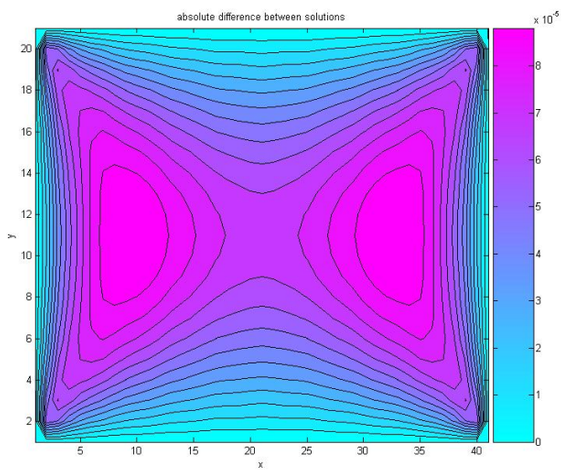


Figure 1: Solution curves

By making the grid smaller, better approximation can be obtained. This analysis was done using 21 by 41 nodes nodes for the rectangular cross section. More elements should

result in better approximation.

In terms of percentage differences, the maximum percentage difference occurred at the 4 corners.

Given the above grid size, we see from the plots that there is a maximum of 1% difference between the analytical solution and the FEM solution. This occurred near the 4 corners of the cross section and was smallest in the middle. We need to better investigate why this is.

This below is a plot showing the percentage difference between both solutions.

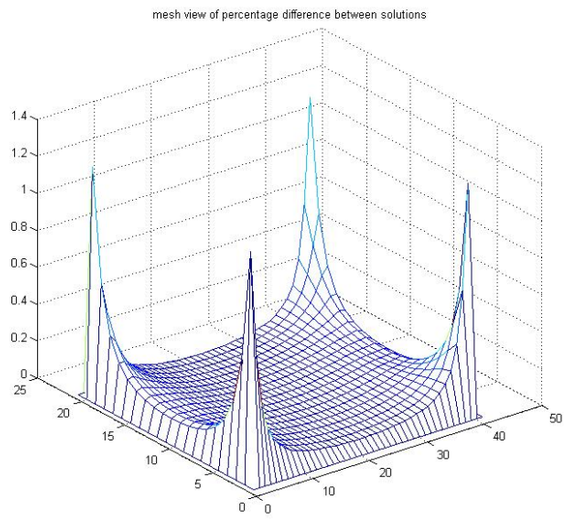


Figure 2: difference in percentage

Below shows a listing of the nodal values for the first 4 columns in the solution matrix.

4

Location of largest % difference between Analytical and FEM solutions are at node near the corners

ANALYTICAL SOLUTION
First 4 columns

```
K>> Z(:,1:4)

ans =

                    0                    0                    0                    0
   0.00000000000000   0.00462202023451   0.00793863989011   0.01050343454601
   0.00000000000000   0.00792356825926   0.01408098533147   0.01899279506363
   0.00000000000000   0.01045229194647   0.01893172530215   0.02587172262713
   0.00000000000000   0.01244143498555   0.02279893291526   0.03143697003105
   0.00000000000000   0.01401063607998   0.02587080239600   0.03589618891092
   0.00000000000000   0.01522981822719   0.02826689465814   0.03939313152396
   0.00000000000000   0.01614281412972   0.03006552477127   0.04202706968694
   0.00000000000000   0.01677777563594   0.03131825945426   0.04386554462739
   0.00000000000000   0.01715230618335   0.03205781625163   0.04495228153952
   0.00000000000000   0.01727611422266   0.03230238760955   0.04531187952838
   0.00000000000000   0.01715230618335   0.03205781625163   0.04495228153952
   0.00000000000000   0.01677777563594   0.03131825945426   0.04386554462739
   0.00000000000000   0.01614281412972   0.03006552477127   0.04202706968694
   0.00000000000000   0.01522981822719   0.02826689465814   0.03939313152396
   0.00000000000000   0.01401063607998   0.02587080239600   0.03589618891092
   0.00000000000000   0.01244143498555   0.02279893291526   0.03143697003105
   0.00000000000000   0.01045229194647   0.01893172530215   0.02587172262713
   0.00000000000000   0.00792356825926   0.01408098533147   0.01899279506363
   0.00000000000000   0.00462202023451   0.00793863989011   0.01050343454601
                    0                    0                    0                    0

K>>
```

FEM SOLUTION
First 4 columns

```
K>> fem_solution(:,1:4)

ans =

                    0   -0.00000000000000   -0.00000000000000    0.00000000000000
                    0    0.00456426271831    0.00788479376139    0.01046187148736
  -0.00000000000000    0.00787225711184    0.01401304083990    0.01892889195460
                    0    0.01041172488916    0.01886622053176    0.02580070707539
                    0    0.01240842191303    0.02273940932260    0.03136616626871
                    0    0.01398255344036    0.02581682857690    0.03582812145294
                    0    0.01520496327151    0.02821723009170    0.03932817386136
                    0    0.01612006955400    0.03001895465701    0.04196477327024
  -0.00000000000000    0.01675636028748    0.03127374571212    0.04380519396305
                    0    0.01713162588381    0.03201447394092    0.04489309870804
                    0    0.01725566930684    0.03225942545973    0.04525308465025
                    0    0.01713162588381    0.03201447394092    0.04489309870804
    0.00000000000000    0.01675636028748    0.03127374571212    0.04380519396305
                    0    0.01612006955400    0.03001895465701    0.04196477327024
                    0    0.01520496327152    0.02821723009170    0.03932817386136
                    0    0.01398255344036    0.02581682857690    0.03582812145294
                    0    0.01240842191303    0.02273940932260    0.03136616626871
                    0    0.01041172488916    0.01886622053176    0.02580070707539
    0.00000000000000    0.00787225711184    0.01401304083990    0.01892889195460
                    0    0.00456426271831    0.00788479376139    0.01046187148736
                    0                    0                    0                    0

K>>
```

Figure 3: numerical difference

The plot below shows the absolute difference between the analytical and the FEM solutions in 3D mesh.

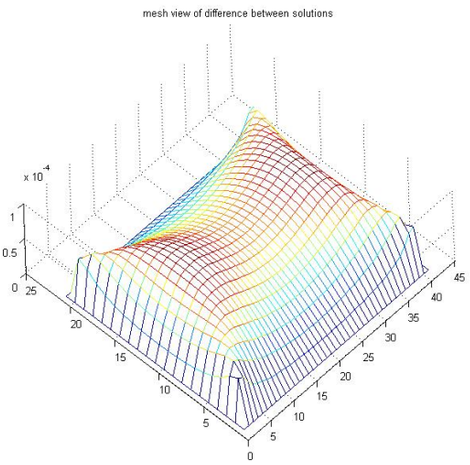


Figure 4: 3D mesh plot of the difference

The plot below is the analytical solution (The wrapping function, i.e. the solution function $\Phi(x, y)$ shown using larger number of contours)

5

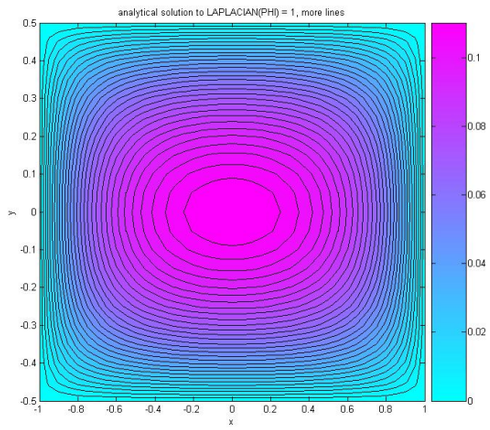


Figure 5: analytical solution

The plot below is a contour plot of the analytic solution.

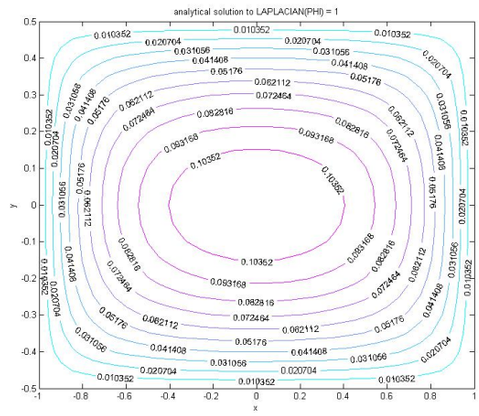


Figure 6: contour plot of the analytic solution

The plot below is a contour plot of the FEM solution to compare with the above.

6

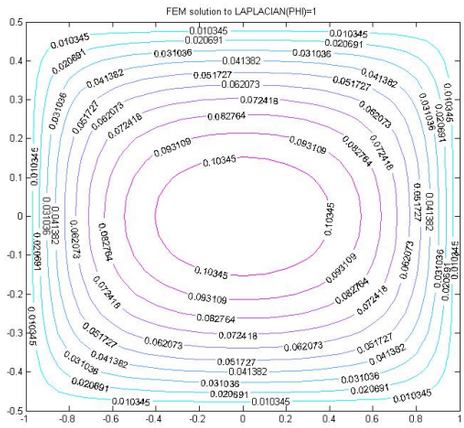


Figure 7: FEM contour plot

The plot below is the above 2 contour plots side-by-side.

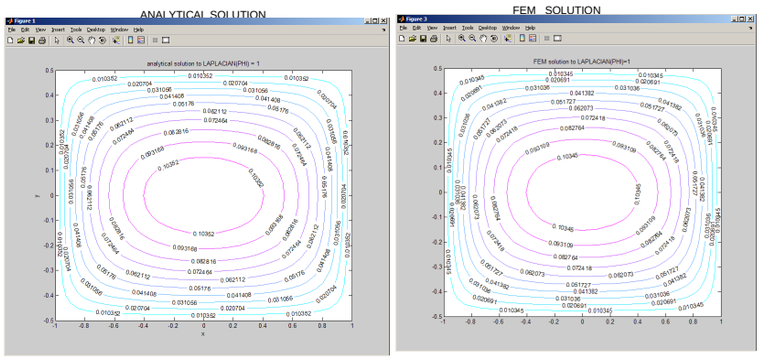


Figure 8: contour plots side by side

This below is a mesh plot of the analytical solution and FEM solution side by side.

7

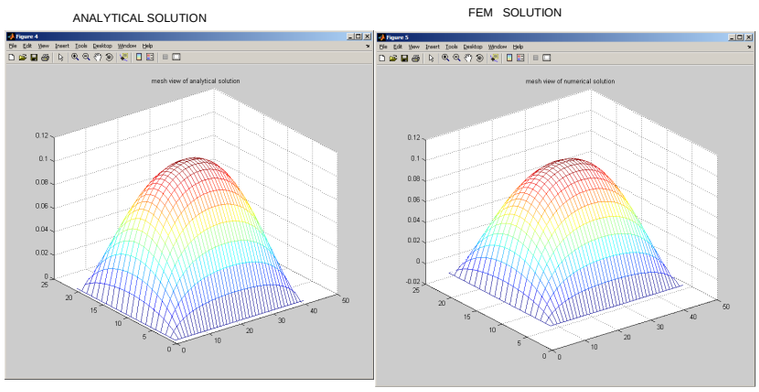


Figure 9: mesh plot side by side

8

3 Appendix

3.1 Modified Qing Wang matlab function

```
function fem_solution=poisson_fem_as_function
% original code by Qing Wang, UCI
%
%minor changes by Nasser M. Abbasi, UCI.  on May 28, 2006:
% - made it a function to be able to call it.
% - changes to contour levels to make it the same as analytical
%   soltution for better comparison.
% - changed the grid to go from -a/2,a/2 instead of 0,a and smiliarly
%   to the b side.
%
% SOLVE THE FOLLOWING PROBLEM USING FEM
%
%    LAPLACIAN( PHI ) = 1
%

format long;
fc=1.0;    % f constant
bc=0.;     % The Dirichlet boundary condition, constant
W=1.0;   % the width of the rectangle
L=2.0;    % the length of the rectangle
N=20;      % the number of elements in Y direction
M=40;       % the number of elements in X direction
dx = L/M;    % dx
dy = W/N;    % dy
D=(N+1)*(M+1);    % The dimension of the global stiffniss matrix or number of nodes
s=zeros(D,D);      % global stiffness matrix
f=zeros(D,1);       % load factor
v=zeros(N+1,M+1);    % the collocated results for plotting
xindex=0;
yindex=0;
x1=0; x2=0; x3=0; x4=0;   % (x1,x2,x3) and (x4,x2,x3) are summits of the two trangles in one
j1=0; j2=0;
% generating the global stiffness matrix and load vectors
for yindex = 1:N
    for xindex = 1:M
        x1 = xindex + (yindex-1)*(M+1);
        x2 = 1+ x1;
        x3 = xindex + yindex*(M+1);
        x4 = 1+ x3;

        j1 = dx*dy;  % the Jacobian of the first trangle
        s(x1,x1) = s(x1,x1)+(dx^2+dy^2)/(2*j1);
        s(x1,x2) = s(x1,x2)-dy^2/(2*j1);
```

```
        s(x1,x3) = s(x1,x3)-dx^2/(2*j1);
        s(x2,x1) = s(x2,x1)-dy^2/(2*j1);
        s(x2,x2) = s(x2,x2)+dy^2/(2*j1);
        s(x2,x3) = s(x2,x3);
        s(x3,x1) = s(x3,x1)-dx^2/(2*j1);
        s(x3,x2) = s(x3,x2);
        s(x3,x3) = s(x3,x3)+dx^2/(2*j1);
        f(x1,1) = f(x1,1)+ fc*j1/6;
        f(x2,1) = f(x2,1)+ fc*j1/6;
        f(x3,1) = f(x3,1)+ fc*j1/6;   %  Process the first trangle in one block

        j2 = dx*dy;   % the Jacobian of the second trangle
        s(x4,x4) = s(x4,x4)+(dx^2+dy^2)/(2*j2);
        s(x4,x2) = s(x4,x2)-dx^2/(2*j2);
        s(x4,x3) = s(x4,x3)-dy^2/(2*j2);
        s(x2,x4) = s(x2,x4)-dx^2/(2*j2);
        s(x2,x2) = s(x2,x2)+dx^2/(2*j2);
        s(x2,x3) = s(x2,x3);
        s(x3,x4) = s(x3,x4)-dy^2/(2*j2);
        s(x3,x2) = s(x3,x2);
        s(x3,x3) = s(x3,x3)+dy^2/(2*j2);
        f(x4,1) = f(x4,1)+ fc*j2/6;
        f(x2,1) = f(x2,1)+ fc*j2/6;
        f(x3,1) = f(x3,1)+ fc*j2/6;  %  Process the second trangle
    end
end

% applying BC
for xindex=1:M+1
    s(xindex,:) = zeros(1,D); s(xindex,xindex)=1; f(xindex,1)=bc;
    s(xindex+N*(M+1),:) = zeros(1,D);
    s(xindex+N*(M+1),xindex+N*(M+1))=1;
    f(xindex+N*(M+1),1)=bc;
end         % the bottome and upper BC
for yindex=1:N-1
    s(1+yindex*(M+1),:)=zeros(1,D);
    s(1+yindex*(M+1),1+yindex*(M+1))=1;
    f(1+yindex*(M+1),1)=bc;
    s((1+yindex)*(M+1),:)=zeros(1,D);
    s((1+yindex)*(M+1),(1+yindex)*(M+1))=1;
    f((1+yindex)*(M+1),1)=bc;
end         % the left and right side BC

q=s\f;      % solve q

% generating visulized results
for yindex=1:N+1
```

10

```
    for xindex=1:M+1
        v(yindex,xindex) = q((M+1)*(yindex-1)+xindex);
    end
end

figure;
[X,Y] = meshgrid(-L/2:dx:L/2,-W/2:dy:W/2);
[C,h] = contour(X,Y,v,10);
clabel(C,h)
colormap cool
title('FEM solution to LAPLACIAN(PHI)=1');
%figure;
%contour(s); title('Global stiffness matrix contour plot');

fem_solution = v;
```

11

3.2 Code used for plotting

Here is the Matlab function used to plot the analytical solution. This function makes a call to the above FEM function.

```
function nma_verify_MAE207_solution
%function nma_verify_MAE207_solution
%
% Display the solution of the poisson equation by
% plotting the analytical solution to the torsion
% problem on a rectangular cross section.
%
% see report and references on
% http://12000.org/my_courses/spring_MAE_207/
% by Nasser Abbasi.
%
% The cross section is
%
%     a   a
%   +---+---+
%   |       | b
%   |       +
%   |       | b
%   +-------+
%
% The origin is in the middle of the cross section:
%
%       y
%       ^
%       |
%       |
%       |
%   +---+---+
%   |   |   |
%   |   +---+-----------------> x
%   |       |
%   +-------+
%
% This Analytical solution is to the following problem
%
%   LAPLACIAN( PHI ) = - 2 G K
%
% To verify the analytical solution against the FEM solution, which solves
%
%   LAPLACIAN( PHI ) = 1
%
% then in this analytical solution we set G*K=-0.5
% But k = T/(G*J), hence we need to have
```

12

```
%
%             T/J = -0.5
%
% change history
% name date    changes
% nma  052806  started

close all;
clear all;

a = 1;    % meter
b = 0.5;  % meter
T_OVER_J = 0.5;   % SET THIS TO MATCH FEM CONSTANT f=1, see above
% I think the Matlab FEM code should have used -1
% but it is easier to change this to +0.5 to match them

dx = 2*a/40;
dy = 2*b/20;

%applied_torque = 1 % Newton-meter   NOT USED

%J = get_torsion_constant(a,b);  NOT USED

[X,Y]    = meshgrid(-a:dx:a, -b:dy:b);
big_term = 0;

NUMBER_TERMS = 100;

for n = 1:2:NUMBER_TERMS
    big_term = big_term + (1/n^3)*(-1)^( (n-1)/2)* ...
        (1- (cosh(n*pi*Y/(2*a))./(cosh(n*pi*b/(2*a))))).*cos(n*pi*X/(2*a));
end
%Z = 32*applied_torque*a^2/(J*pi^3) * big_term;
Z = 32*T_OVER_J*a^2/(pi^3) * big_term;
%
%  DONE. Now do plots
%
figure;
[C,h] = contour(X,Y,Z,10);
clabel(C,h)
colormap cool
title('analytical solution to LAPLACIAN(PHI) = 1');
xlabel('x'); ylabel('y');
axis square

figure;
[C,h] = contourf(X,Y,Z,30);
```

13

```
colormap cool
title('analytical solution to LAPLACIAN(PHI) = 1, more lines');
xlabel('x'); ylabel('y');

%
% now obtain the FEM solution and compare point-to-point
%

fem_solution = poisson_fem_as_function();

difference      = fem_solution-Z;
abs_difference  = abs(difference);

figure;
[C,h] = contourf(abs_difference,13);
%clabel(C,h)
colormap cool
title('absolute difference between solutions');
xlabel('x'); ylabel('y');

figure;
mesh(abs_difference);
title('mesh view of absolute difference between solutions');


figure;
mesh(Z); title('mesh view of analytical solution');

figure;
mesh(fem_solution); title('mesh view of numerical solution');

figure;
[nRow,nCol]=size(Z);
per_diff=zeros(nRow,nCol);
for i=1:nRow
    for j=1:nCol
        if abs(Z(i,j))>eps
            per_diff(i,j)= 100* abs_difference(i,j)/Z(i,j);
        else
            per_diff(i,j)= 100* abs_difference(i,j);
        end
    end
end
mesh(per_diff); title('mesh view of percentage difference between solutions');

figure;
[C,h] = contourf(per_diff,13);
```

14

```
%clabel(C,h)
colormap cool
title('% difference between solutions');
xlabel('x'); ylabel('y');

[r,c]=find(abs_difference==max(abs_difference(:)));
fprintf('Maximum difference at node [%d,%d]\n',r,c);


end

%********************
%
%
%********************
function J = get_torsion_constant(a,b)
NUMBER_TERMS = 1000;

fixed_term = pi*b/(2*a);
J = 0;
for n=1:2:NUMBER_TERMS
    J = J + tanh(n*fixed_term)/n^5;
end
J = J * 192*a/(b*pi^5);
J = 16*a^3*b/3 * (1 - J);
end
```

2.1.6 A note on Gaussian Quadrature for one dimension

Review of Gaussian Quadrature method
MAE 207, Computational methods. UCI
Spring 2006

Nasser M. Abbasi

Spring 2006 Compiled on September 4, 2021 at 9:07pm

Contents

1 The problem

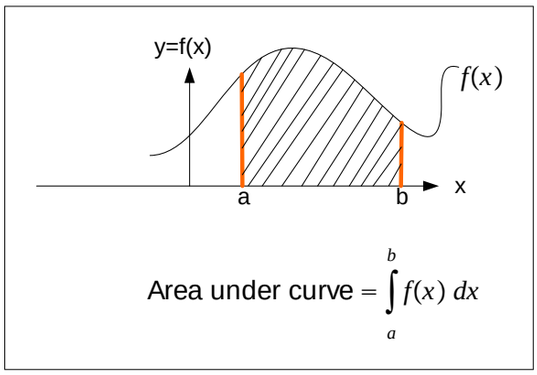


Figure 1: Area under the curve

2

To find a numerical value for the integral of a real valued function of a real variable over a specific range over the real line. This means to evaluate

$$I = \int_a^b f(x)\, dx$$

Geometrically, this integral represents the area under $f(x)$ from a to b

2 Solution

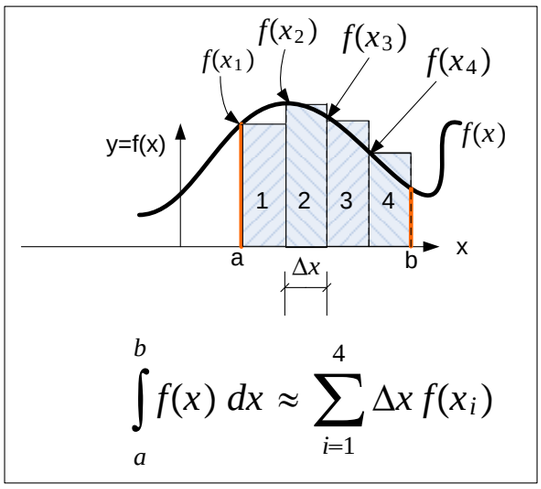


Figure 2: Integrating a function

We can always approximate the area by dividing it in equal width strips and then sum the areas of all the strips.

In general, there will always be an error in the estimate of the area using this method. The error will become smaller the more strips we use (which implies a smaller strip width). Hence we can write

$$\int_a^b f(x)\, dx = \left(\sum_{i=1}^{N} \Delta x\, f(x_i) \right) + E$$

Where E is the error between the actual area and the approximated area using the above method of numerical integration. N above is the number of strips or can also be refereed to as the number of integration points.

Instead of keep referring to the 'width of the strip' all the time, we will call this quantity the weight w_i that we will multiply the value of the function with to obtain the area. Hence the above becomes

$$\int_a^b f(x)\,dx = \left(\sum_{i=1}^{N} w_i\, f\,(x_i)\right) + E$$

Using implied summation on indices the above becomes

$$\int_a^b f(x)\,dx = w_i\, f\,(x_i) + E$$

In the above we divided the range of the integration (the distance between the upper and lower limits of integration) into equal intervals. We can decide to evaluate $f(x_i)$ at the middle of the strip or at the start of the strip or at the end of the strip. In the diagram above we evaluated the $f(x)$ at the left end of the strip.

Our goal is to evaluate this integral such as the error E is minimum and using the smallest number of integration points. In a sense this can be considered an optimization problem with constraints: minimize the error of integration using the smallest possible number of points.

To be able to do this minimization, we need to consider what are the variables involved. We see that there are two degrees of freedom to this problem. One is the width of the strip or w_i. We do not have to use a fixed value of width, we can use different width for different strips if the resulting integral gives better approximation.

The second degree of freedom is the point at which to evaluate $f(x_i)$ associated with each strip. In the example above we choose to evaluate $f(x_i)$ at left end point of the strip. We can choose to select a different x_i point if this will result in a better approximation.

This is the main idea of Gauss Quadrature numerical integration. It is to be able to choose specific values for these two degrees of freedom, the w_i and the x_i.

It turns out that if the function $f(x)$ is a polynomial, then there is an optimal solution. There is an optimal $\{w_i, x_i\}$ for each polynomial of order n.

We can determine these degrees of freedom such that the error E is zero, and with the least possible number of integration points. We are able to tabulate these two degrees of freedom for each polynomial of specific order. In other words, if the function $f(x)$ is a polynomial of order n then we know before the computation starts what these 2 degrees of freedom should be. We know the locations of x_i and we know weight w_i that we need to multiply $f(x_i)$ with to obtain the area with minimum error.

You might ask how can this method of integration know the locations of the integration points x_i beforehand without being given the integration range of the function to integrate? It turns out that we will map $f\,(x)$ into a new known and specific range of integration (from –1 to +1) for the method that we will now discuss.

2.1 Gauss Quadrature

From now on we will assume the function $f(x)$ to be integrated is a polynomial in x of some order n.

Gauss quadrature is optimal when the function is a polynomial

The main starting point is to represent the function $f(x)$ as a combination of linearly independent basis.

Instead of using strips of equal width, we assume the width can vary from one strip to the next. Let us call the width of the strip w_i. Instead of taking the height of the i^{th} strip to be the value of the function at the left edge of the strip, let us also be flexible on the location of the x associated with strip w_i and call the height of the strip w_i as $f(x_i)$ where x_i is to be determined. Hence the above integration becomes

$$\begin{aligned}\int_a^b f(x)\,dx &= \left(\sum_{i=1}^{N} w_i\, f(x_i)\right) + E \\ &\approx \sum_{i=1}^{N} w_i\, f(x_i)\end{aligned}$$

So our goal is to determine w_i and x_i such as the error E is minimized in the above equation. We would really like to find w_i and x_i such that the error is zero with the smallest value for N.

It seems as if this is a very hard problem. We have $2N$ unknown quantities to determine. N different widths, and N associated x points to evaluate the height of each specific strip at. And we only have as an input $f(x)$ and the limits of integration, and we need to determine these $2N$ quantities such that the error in integration is zero.

In other words, the problem is to find w_i, x_i such that

$$I = \int_a^b f(x)\,dx = w_1 f(x_1) + w_2 f(x_2) + \cdots w_N f(x_N) \tag{1}$$

One way to make some progress is to expand $f(x)$ as a series. We can approximate $f(x)$ as convergent power series for example. If $f(x)$ happens to be a polynomial instead, we can represent it exactly using a finite sequence of Legendre polynomials. It is in this second case where this method makes the most sense to use due to the advantages we make from the second representation.

We show both methods below.

Expanding $f(x)$ as convergent power series over the range a, b gives

$$f(x) = a_0 + a_1 x + a_2 x^2 + \cdots a_m x^m + \cdots \tag{2}$$

Substituting (2) into (1) gives

$$I = \int_a^b \left(a_0 + a_1 x + a_2 x^2 + \cdots a_m x^m + \cdots \right) dx = w_1 f(x_1) + w_2 f(x_2) + \cdots w_N f(x_N) \tag{3}$$

But

$$\begin{aligned}\int_a^b \left(a_0 + a_1 x + a_2 x^2 + \cdots a_m x^m + \cdots \right) dx &= \int_a^b a_0 dx + \int_a^b a_1 x\, dx + \int_a^b a_2 x^2\, dx + \cdots + \int_a^b a_m x^m\, dx + \cdots \\ &= a_0 (b-a) + a_1 \frac{\left(b^2 - a^2\right)}{2} + a_2 \frac{\left(b^3 - a^3\right)}{3} + \cdots + a_m \frac{\left(b^{m+1} - a^{m+1}\right)}{m+1} + \cdots\end{aligned}$$

Substituting the above into (3) results in

$$\begin{aligned}a_0 (b-a) + a_1 \frac{\left(b^2 - a^2\right)}{2} + a_2 \frac{\left(b^3 - a^3\right)}{3} + \cdots + a_m \frac{\left(b^{m+1} - a^{m+1}\right)}{m+1} + \cdots \\ = w_1 f(x_1) + w_2 f(x_2) + \cdots + w_N f(x_N)\end{aligned} \tag{4}$$

But from (2) we see that

$$\begin{aligned}f(x_1) &= a_0 + a_1 x_1 + a_2 x_1^2 + \cdots a_m x_1^m + \cdots \\ f(x_2) &= a_0 + a_1 x_2 + a_2 x_2^2 + \cdots a_m x_2^m + \cdots \\ &\cdots \\ f(x_N) &= a_0 + a_1 x_N + a_2 x_N^2 + \cdots a_m x_N^m + \cdots\end{aligned}$$

Substituting the above into (4) gives

$$\begin{aligned}a_0 (b-a) + a_1 \frac{\left(b^2 - a^2\right)}{2} + a_2 \frac{\left(b^3 - a^3\right)}{3} + \cdots + a_m \frac{\left(b^{m+1} - a^{m+1}\right)}{m+1} + \cdots = \\ w_1 \left(a_0 + a_1 x_1 + a_2 x_1^2 + \cdots a_m x_1^m + \cdots \right) \\ + w_2 \left(a_0 + a_1 x_2 + a_2 x_2^2 + \cdots a_m x_2^m + \cdots \right) \\ \cdots \\ + w_N \left(a_0 + a_1 x_N + a_2 x_N^2 + \cdots a_m x_N^m + \cdots \right)\end{aligned}$$

Rearranging gives

$$\begin{aligned}a_0 (b-a) + a_1 \frac{\left(b^2 - a^2\right)}{2} + a_2 \frac{\left(b^3 - a^3\right)}{3} + \cdots + a_m \frac{\left(b^m - a^m\right)}{m} + \cdots = \\ a_0 (w_1 + w_2 + \cdots + w_N) \\ + a_1 (w_1 x_1 + w_2 x_2 + \cdots + w_N x_N) \\ + a_2 \left(w_1 x_1^2 + w_2 x_2^2 + \cdots + w_N x_N^2 \right) \\ \cdots \\ + a_m \left(w_1 x_1^m + w_2 x_2^m + \cdots + w_N x_N^m \right)\end{aligned}$$

Equating coefficients on both sides results in

$$
\begin{aligned}
w_1 + w_2 + \cdots + w_N &= b - a \qquad (5)\\
w_1 x_1 + w_2 x_2 + \cdots + w_N x_N &= \frac{\left(b^2 - a^2\right)}{2}\\
w_1 x_1^2 + w_2 x_2^2 + \cdots + w_N x_N^2 &= \frac{\left(b^3 - a^3\right)}{3}\\
&\cdots\\
w_1 x_1^m + w_2 x_2^m + \cdots + w_N x_N^m &= \frac{(b^m - a^m)}{m}\\
&\cdots
\end{aligned}
$$

Since we have $2N$ unknown quantities to solve for (N weights w_i and N points x_i) we need $2N$ equations. In other words, we need to have $m = 2N$. The above set of simultaneous $2N$ equations can now be solved for the unknown w_i, x_i and this will give us the numerical integration we wanted.

The above assumed that the function $f(x)$ can be represented exactly by the power series expansion with m terms.

We now consider the representation of $f(x)$ as a series of Legendre polynomials. We do this since when $f(x)$ itself is a polynomial. We can represent $f(x)$ exactly by a finite number of Legendre polynomials. But since Legendre polynomials $P_n(x)$ are defined over $[-1, 1]$ we start by transforming $f(x)$ to this new range and then we can expand the mapped $f(x)$ (which we will call $g(\zeta)$) in terms of the Legendre polynomials.

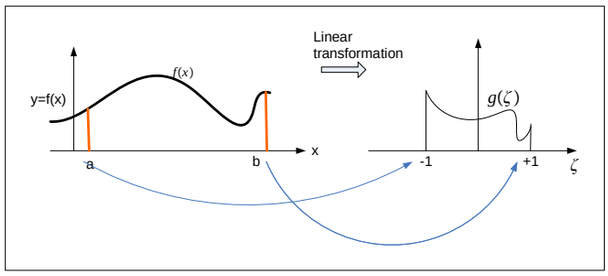


Figure 3: Mapping

An easy way to find this mapping is to align the ranges over each others and take the ratio between as the scale factor. This diagram shows this for a general case where we map $f(x)$ defined over $[a, b]$ to a new range defined over $[c_1, c_2]$

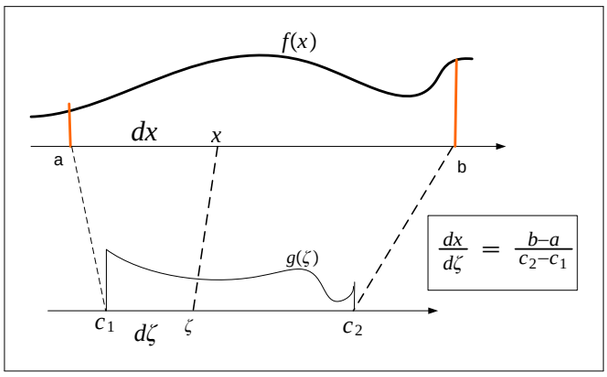


Figure 4: Mapping over new range

We see from the diagram that

$$\zeta = c_1 + d\zeta$$

But $\frac{d\zeta}{dx}$ is the same ratio as $\frac{b-a}{c_2-c_1}$. Hence

$$\frac{dx}{d\zeta} = \frac{b-a}{c_2-c_1} \tag{6}$$

The above is called the Jacobian of the transformation. Now, From the diagram we see that

$$dx = x - a$$

And

$$d\zeta = \zeta - c_1$$

Eq. (6) becomes

$$\frac{x-a}{\zeta - c_1} = \frac{b-a}{c_2-c_1}$$

Hence we obtain that

$$\zeta = \frac{x-a}{b-a}(c_2 - c_1) + c_1$$

And

$$x = \frac{b-a}{c_2-c_1}(\zeta - c_1) + a$$

For the specific case when $c_1 = -1$ and $c_2 = +1$ the above expression becomes

$$\begin{aligned}\zeta &= \frac{x-a}{b-a}(2) - 1 \\ &= \frac{2x - 2a - (b-a)}{b-a} \\ &= \frac{2x - a - b}{b-a}\end{aligned}$$

8

Therefore

$$\begin{aligned} x &= \frac{b-a}{2}(\zeta+1)+a \\ &= \frac{(b-a)\zeta+(a+b)}{2} \end{aligned}$$

Before we proceed further, It will be interesting to see the effect of this transformation on the shape of some functions. Below I plotted some functions under this transformation. The left plots are the original functions plotted over some range, in this case $[4,10]$ and the right side plots show the new shape $(\text{the function } g(\zeta))$ over the new range $[-1,1]$

9

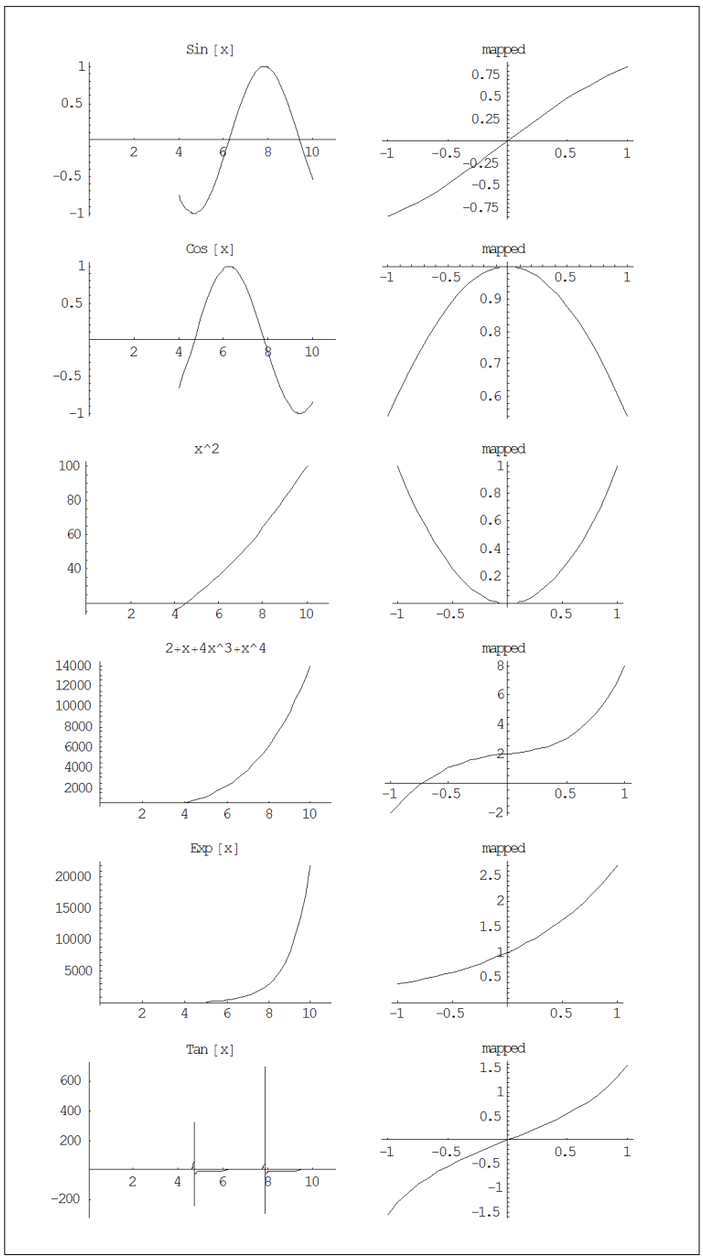


Figure 5: plotted some functions under this transformation

We must remember that in the following analysis, we are integrating now the function $g(\zeta)$ over $[-1,1]$ and not $f(x)$ over $[a,b]$. Hence to obtain the required integral we need to transform back the final result. We will show how to do this below.

We can approximate any function $f(x)$ as a series expansion in terms of weighted sums of a set of basis functions. We do this for example when we use Fourier series expansion. Hence we write

$$f(x) = \sum_{i}^{\infty} a_i \Phi_i(x) \tag{7}$$

We can give an intuitive justification to the above formulation as follows. If we think in terms of vectors. A vector $\mathbf{V}$ in an n-dimensional space is written in terms of its components as follows

$$\begin{aligned}\mathbf{V} &= a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + \cdots a_N\mathbf{e}_N \\ &= \sum_{i}^{N} a_i\mathbf{e}_i\end{aligned}$$

Where $\mathbf{e}_i$ is the basis vectors in that space.

If we now consider a general infinite dimensional vector space where each point in that space is a function, then we see that we can also represent that function as a weighted series of a basis functions just as we did for a normal vector.

There are many sets of basis functions we can choose to represent the function $f(x)$ with. The main requirements for the basis functions is that they are complete (This means they span the whole space) and there is defined an inner product on them.

For our purposes here, we are interested in the class of function $f(x)$ that are polynomials in x. The basis we will select to use are the Legendre basis as explained above. To do this, we transform $f(x)$ to $g(\zeta)$ as shown above and then now the integral becomes

$$\int_a^b f(x)dx = \int_{-1}^{1} f(x(\zeta))\left(\frac{(b-a)}{2}d\zeta\right)$$

This is because we found that $dx = \frac{(b-a)}{2}d\zeta$ from above. If we call $f(x(\zeta))$ as $g(\zeta)$ the integral becomes

$$\int_a^b f(x)dx = \int_{-1}^{1} \frac{(b-a)}{2} g(\zeta)\ d\zeta$$

Since $\frac{(b-a)}{2}$ is the Jacobian of the transformation, we write the integral as

$$\int_a^b f(x)dx \rightarrow \int_{-1}^{1} J\, g(\zeta)\ d\zeta$$

Since the Jacobian is constant in this case, we will only worry about $\int_{-1}^{1} g(\zeta)\ d\zeta$ and we just need to remember to scale the result at the end by J. This is the integral we will now numerically integrate. Equation (7) is now written as

$$g(\zeta) = \sum_{i}^{\infty} a_i P_i(\zeta)$$

Where $P_i(\zeta)$ is the Legendre polynomial of order i and $g(\zeta)$ is a polynomial in ζ or some order m.

Now we carry the same analysis we did when we expressed $f(x)$ as a power series. The difference now is that the limit of integration is symmetrical and the basis are now the Legendre polynomials instead of the polynomials x^n as the case was in the power series expansion. So now equation (1) becomes

$$I = \int_{-1}^{1} g(\zeta)\ d\zeta = w_1 g(\zeta_1) + w_2 g(\zeta_2) + \cdots + w_N g(\zeta_N) \tag{8}$$

And equation (2) becomes

$$g(\zeta) = a_0 P_0(\zeta) + a_1 P_1(\zeta) + a_2 P_2(\zeta) + \cdots a_m P_m(\zeta) + \cdots \tag{9}$$

Substituting (9) into (8) we get the equivalent of equation (3)

$$\begin{aligned} I &= \int_{-1}^{1} \left(a_0 P_0(\zeta) + a_1 P_1(\zeta) + a_2 P_2(\zeta) + \cdots a_m P_m(\zeta) + \cdots\right)\ d\zeta \\ &= w_1 g(\zeta_1) + w_2 g(\zeta_2) + \cdots + w_N g(\zeta_N) \end{aligned} \tag{10}$$

Therefore

$$\begin{aligned} \int_{-1}^{1} \left(a_0 P_0 + a_1 P_1 + a_2 P_2 + \cdots a_m P_m + \cdots\right)\ d\zeta &= \int_{-1}^{1} a_0 P_0 d\zeta + \int_{-1}^{1} a_1 P_1 d\zeta + \int_{-1}^{1} a_2 P_2\ d\zeta + \cdots + \int_{-1}^{1} a_m P_m d\zeta\ + \cdots \\ &= a_0(2) + a_1 0 + a_2 0 + \cdots + a0 + \cdots \\ &= 2a_0 \end{aligned}$$

The above occurs since the integral of any Legendre polynomial of order greater than zero will be zero over $[-1, 1]$

Substituting the above into (10) we obtain

$$I = 2a_0 = w_1 g(\zeta_1) + w_2 g(\zeta_2) + \cdots + w_N g(\zeta_N) \tag{11}$$

But from (9) we see that

$$\begin{aligned} g(\zeta_1) &= a_0 P_0(\zeta_1) + a_1 P_1(\zeta_1) + a_2 P_2(\zeta_1) + \cdots a_m P_m(\zeta_1) + \cdots \\ g(\zeta_2) &= a_0 P_0(\zeta_2) + a_1 P_1(\zeta_2) + a_2 P_2(\zeta_2) + \cdots a_m P_m(\zeta_2) + \cdots \\ &\cdots \\ g(\zeta_N) &= a_0 P_0(\zeta_N) + a_1 P_1(\zeta_N) + a_2 P_2(\zeta_N) + \cdots a_m P_m(\zeta_N) + \cdots \end{aligned}$$

Substituting the above in (11) gives

$$\begin{aligned}2a_0 = &w_1\left(a_0P_0(\zeta_1) + a_1P_1(\zeta_1) + a_2P_2(\zeta_1) + \cdots a_mP_m(\zeta_1) + \cdots\right) + \\ &+ w_2\left(a_0P_0(\zeta_2) + a_1P_1(\zeta_2) + a_2P_2(\zeta_2) + \cdots a_mP_m(\zeta_2) + \cdots\right) + \\ &\cdots \\ &+ w_N\left(a_0P_0(\zeta_N) + a_1P_1(\zeta_N) + a_2P_2(\zeta_N) + \cdots a_mP_m(\zeta_N) + \cdots\right)\end{aligned}$$

Rearranging results in

$$\begin{aligned}2a_0 = &a_0\left(w_1P_0(\zeta_1) + w_2P_0(\zeta_2) + \cdots + w_NP_0(\zeta_N)\right) \\ &+ a_1\left(w_1P_1(\zeta_1) + w_2P_1(\zeta_2) + \cdots + w_NP_1(\zeta_N)\right) \\ &+ \cdots \\ &+ a_m\left(w_1P_m(\zeta_1) + w_2P_m(\zeta_2) + \cdots + w_NP_m(\zeta_N)\right)\end{aligned}$$

Equating coefficients gives

$$\begin{aligned}2 &= w_1P_0(\zeta_1) + w_2P_0(\zeta_2) + \cdots + w_NP_0(\zeta_N) \\ 0 &= w_1P_1(\zeta_1) + w_2P_1(\zeta_2) + \cdots + w_NP_1(\zeta_N) \\ 0 &= w_1P_2(\zeta_1) + w_2P_2(\zeta_2) + \cdots + w_NP_2(\zeta_N) \\ &\cdots \\ 0 &= w_1P_m(\zeta_1) + w_2P_m(\zeta_2) + \cdots + w_NP_m(\zeta_N)\end{aligned}$$

If we select the points ζ_i to be the roots of P_{i-1} we can write the above as

$$\begin{aligned}2 &= w_1P_0(\zeta_1) + w_2P_0(\zeta_2) + \cdots + w_NP_0(\zeta_N) \\ 0 &= w_1P_1(\zeta_1) + w_2P_1(\zeta_2) + \cdots + w_NP_1(\zeta_N) \\ 0 &= w_1P_2(\zeta_1) + w_2P_2(\zeta_2) + \cdots + w_NP_2(\zeta_N) \\ &\cdots \\ 0 &= w_1P_m(\zeta_1) + w_2P_m(\zeta_2) + \cdots + w_NP_m(\zeta_N)\end{aligned}$$

3 References

1. Mathematical Structural Mechanics help page
2. MIT course 16.20 lecture notes. MIT open course website.
3. Theory of elasticity by S. P. Timoshenko and J. N. Goodier. chapter 10

clearpage

2.2 Documents by Roy Culver (from UCI MAE 207 spring 2006 class)

Local contents

2.2.1 Review of FEM document. Updated May 25,2006

Notes on Application of Finite Element Method to the Solution of the Poisson Equation

Consider the arbitrary two dimensional domain, Ω shown in Figure 1. The Poisson equation for this domain can be written

$$\nabla^2\phi = 1 \quad (1)$$
$$\phi = 0 \quad \text{on } \partial\Omega \quad (2)$$

where we have specified a constant right hand side, and dirichilet boundary conditions.

Applying the weighted residual method, we get:

$$\int_\Omega (\nabla^2 u - 1)\nu \mathrm{d}\Omega = 0 \quad (3)$$

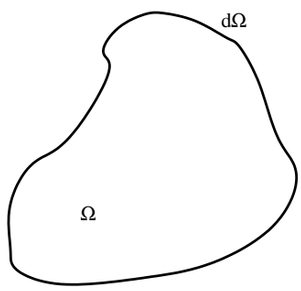


Figure 1: Domain for the Poisson Equation

Now, making use of the divergence theorem to reduce the order of the weighted residual equation we may express the integral as

$$\int_{\partial\Omega} \nu\nabla u \cdot n\partial\Omega - \int_\Omega \nabla u \cdot \nabla\nu \mathrm{d}\Omega - \int_\Omega \nu \mathrm{d}\Omega = 0 \quad (4)$$

Equation 4 is called the symmetric weak form of the Poisson equation. The equation contains three separate integrals. A boundary integral involving the flux of u, the symmetric integral for the integral domain and the load integral. Examining Equation 4 we see that an amissable solution for u and ν only requires a C^0 continuous function. As long as the test function ν satisfies this requirement may used. So, for convenience we may also specify that ν vanishes at the boundary, thus the boundary integral also will vanish.

Due to the arbitrariness of the solution domain, we discretize the computational domain and evaluate the symmetric weak form as the sum of integrals over triangular patches. Figure 2 shows the discretized domain. Since we only require that the test and trial functions are C^0 continuous, we can represent u as a linear function of the nodal values as shown in Figure 2. Here we assume local linear trial and test functions of the form

$$u^j = \sum q_i^j N_i \qquad i = 1, 2, 3$$
$$\nu^j = \sum p_k^j N_k \qquad k = 1, 2, 3$$

Substituting these test and trial function into the second integral of the symmetric

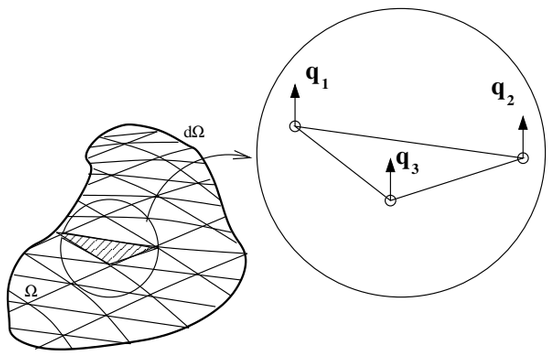


Figure 2: Domain Discretization

weak form we get

$$\begin{aligned}\int_\Omega \nabla u \cdot \nabla \nu \mathrm{d}\Omega &= \sum_{j=1}^{N} \int_\Omega \nabla(q_n^j N_n) \cdot \nabla(p_m^j N_m)\mathrm{d}\Omega \\ &= \sum_{j=1}^{N} \int_\Omega q_n^j N_{n,i} p_m^j N_{m,i}\mathrm{d}\Omega \\ &= \sum_{j=1}^{N} p_m^j K_{m,n}^j q_n^j\end{aligned}$$

where $K_{m,n}^j$, the element stiffness matrix, is given by

$$K_{m,n}^j = \int_{\Omega_j} N_{n,i} N_{m,i}\mathrm{d}\Omega_\mathrm{j} \tag{5}$$

Corrdinate Transformation and Shape Functions

Before we write the shape functions explicitly, we first define a local linear transformation under which the coordinates of each element are mapped into the ξ, η plane. Figure 3 shows graphically what this transformation would look like. To explicitly define this transformation, we represent x and y as linear functions of ξ and η in the form

$$\begin{aligned}x &= a + b\xi + c\eta \\ y &= d + e\xi + g\eta\end{aligned}$$

By evaluating these expressions at all three values of x, y corresponding to values of

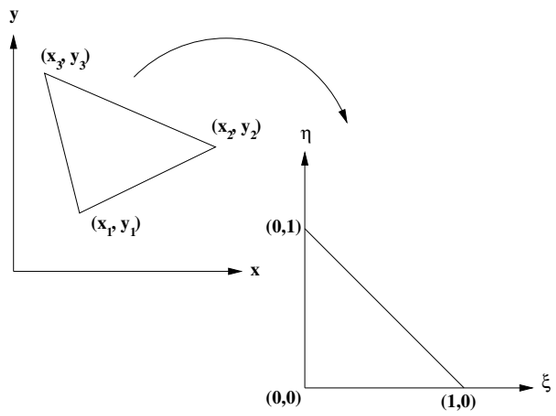


Figure 3: Transformation from Global to Local Coordinate System

ξ, η, we get a system of six equations for a, b, c, d, e, and g.

$$\begin{aligned} x_1 &= a \\ x_2 &= a + b \\ x_3 &= a + c \\ y_1 &= d \\ y_2 &= d + e \\ y_3 &= d + g \end{aligned}$$

Solving these equations we see the correct local tranformation is

$$\begin{array}{rcccccc} x & = & x_1 & + & (x_2 - x_1)\xi & + & (x_3 - x_1)\eta \\ y & = & y_1 & + & (y_2 - y_1)\xi & + & (y_3 - y_1)\eta \end{array} \tag{6}$$

Also, we may evaluate the Jacobian of the transformation to give us the relationship between the two coordinate systems as

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \mathrm{x}}{\partial \xi} & \frac{\partial \mathrm{x}}{\partial \eta} \\ \frac{\partial \mathrm{y}}{\partial \xi} & \frac{\partial \mathrm{y}}{\partial \eta} \end{bmatrix} = \begin{bmatrix} (x_2 - x_1) & (x_3 - x_1) \\ (y_2 - y_1) & (y_3 - y_1) \end{bmatrix} = \begin{bmatrix} \Delta \mathrm{x}_2 & \Delta \mathrm{x}_3 \\ \Delta \mathrm{y}_2 & \Delta \mathrm{y}_3 \end{bmatrix} \tag{7}$$

where we have introduced the notation $(x_2 - x_1) = \Delta \mathrm{x}_2$ for a convenience.

To define the shape functions we note the desired boundary conditions for each shape function as shown in Table 1. Since the global requirement of continuity on the trial function is only C^0 continuity, we may assume a linear form for the local shape function $(N_i = c_1^i + c_2^i \xi + c_3^i \eta)$. Evaluating the boundary conditions we get a system of nine

	(x_1, y_1)	(x_2, y_2)	(x_3, y_3)
N_1	1	0	0
N_2	0	1	0
N_3	0	0	1

Table 1: Boundary Conditions for Shape Functions

equations with nine unknowns.

$$\begin{aligned}
N_1(1) &= 1 &&= c_1^1 \\
N_1(2) &= 0 &&= c_1^1 + c_2^1 \\
N_1(3) &= 0 &&= c_1^1 + c_3^1 \\
N_2(1) &= 0 &&= c_1^2 \\
N_2(2) &= 1 &&= c_1^2 + c_2^2 \\
N_2(3) &= 0 &&= c_1^2 + c_3^2 \\
N_3(1) &= 0 &&= c_1^3 \\
N_3(2) &= 0 &&= c_1^3 + c_2^3 \\
N_3(3) &= 1 &&= c_1^3 + c_3^3
\end{aligned}$$

Solving this system, we arrive at the shape functions in the element local coordinate system as

$$\begin{aligned}
N_1 &= 1 - \xi - \eta \\
N_2 &= \xi \\
N_3 &= \eta
\end{aligned} \tag{8}$$

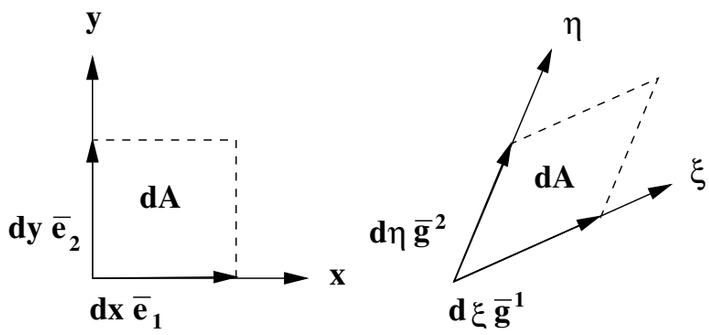
Coordinate System Transformations for Area Integrals

Because of the simplicity of the element local coordinate system, it is desirable to evaluate the stiffness matrix integral in this coordinate system. However, since the original equation is in terms of the standard Cartesian coordinate system, we must properly rewrite the integral in terms of the local coordinates. To do this, we must properly express a differential area element in this new system. In Figure 4(a) we see the differential area element represented in the Cartesian coordinate system. Noting that the area of the parallelagram formed between two non-parallel vectors can found by the norm of the cross product of the two vectors, we express the area element dA in terms of the unit base vectors $\overline{\mathbf{e}}_1$,$\overline{\mathbf{e}}_2$ and differential lengths dx, dy as:

$$\begin{aligned}
\mathrm{dA} &= \|\mathrm{dx}\,\overline{\mathbf{e}}_1 \times \mathrm{dy}\,\overline{\mathbf{e}}_2\| \\
&= \|\overline{\mathbf{e}}_1 \times \overline{\mathbf{e}}_2\|\mathrm{dx}\,\mathrm{dy} \\
&= \|\overline{\mathbf{e}}_3\|\mathrm{dx}\,\mathrm{dy} \\
&= \mathrm{dx}\,\mathrm{dy}
\end{aligned}$$

Similarly, for the element local coordinate system we may represent the differential area element in terms of the covariant base vectors $\overline{\mathbf{g}}^1$,$\overline{\mathbf{g}}^2$ and the differential lengths $\mathrm{d}\xi$, $\mathrm{d}\eta$ as:

$$\begin{aligned}
\mathrm{dA} &= \|\mathrm{d}\xi\,\overline{\mathbf{g}}^1 \times \mathrm{d}\eta\,\overline{\mathbf{g}}^2\| \\
&= \|\overline{\mathbf{g}}^1 \times \overline{\mathbf{g}}^2\|\mathrm{d}\xi\,\mathrm{d}\eta
\end{aligned}$$



(a) Cartesian Coordinate System (b) Element Local Coordinate System

Figure 4: Area Elements in Respective Coordinate Systems

To evaluate the cross product, $\overline{\mathbf{g}}^1 \times \overline{\mathbf{g}}^2$, we first express a vector $\overline{\mathbf{R}}$ in terms of both coordinate systems as:

$$\overline{\mathbf{R}} = \overline{\mathbf{g}}^1\,\xi + \overline{\mathbf{g}}^2\,\eta = \overline{\mathbf{e}}_1\,\mathrm{x} + \overline{\mathbf{e}}_2\,\mathrm{y}$$

By definition the covariant base vectors are:

$$\overline{\mathbf{g}}^1 = \frac{\partial \overline{\mathbf{R}}}{\partial \xi} = \frac{\partial(\overline{\mathbf{e}}_1\,\mathrm{x} + \overline{\mathbf{e}}_2\,\mathrm{y})}{\partial \xi} + \frac{\partial(\overline{\mathbf{e}}_1\,\mathrm{x} + \overline{\mathbf{e}}_2\,\mathrm{y})}{\partial \xi} = \overline{\mathbf{e}}_1\,\frac{\partial \mathrm{x}}{\partial \xi} + \overline{\mathbf{e}}_2\,\frac{\partial \mathrm{y}}{\partial \xi}$$

$$\overline{\mathbf{g}}^2 = \frac{\partial \overline{\mathbf{R}}}{\partial \eta} = \frac{\partial(\overline{\mathbf{e}}_1\,\mathrm{x} + \overline{\mathbf{e}}_2\,\mathrm{y})}{\partial \eta} + \frac{\partial(\overline{\mathbf{e}}_1\,\mathrm{x} + \overline{\mathbf{e}}_2\,\mathrm{y})}{\partial \eta} = \overline{\mathbf{e}}_1\,\frac{\partial \mathrm{x}}{\partial \eta} + \overline{\mathbf{e}}_2\,\frac{\partial \mathrm{y}}{\partial \eta}$$

Now that we have expressed the covariant base vectors in terms of the known cartesian base vectors we may evaluate the cross product. The result is

$$\|\overline{\mathbf{g}}^1 \times \overline{\mathbf{g}}^2\| = \left\| \begin{array}{cc} \frac{\partial \mathrm{x}}{\partial \xi} & \frac{\partial \mathrm{y}}{\partial \xi} \\ \frac{\partial \mathrm{x}}{\partial \eta} & \frac{\partial \mathrm{y}}{\partial \eta} \end{array} \right\| = \frac{\partial \mathrm{x}}{\partial \xi}\,\frac{\partial \mathrm{y}}{\partial \eta} - \frac{\partial \mathrm{x}}{\partial \eta}\,\frac{\partial \mathrm{y}}{\partial \xi} = \mathrm{J}$$

Recognizing the matrix resulting from the cross product $\overline{\mathbf{g}}^1 \times \overline{\mathbf{g}}^2$ as the Jacobian of the transformation, we have used the (J) to denote that the norm of the cross product is also the determinant of the Jacobian matrix. Completing the expression for the differential area element in the element local coordinate system, we have

$$\mathrm{dA} = \mathrm{J}\,\mathrm{d}\xi\,\mathrm{d}\eta$$

To summarize, we have shown that in order properly evaluate an are integral in the element local coordinate system, we should use Equation 9. This equation arose from reexpressing the differential area in the element local coordinate system in terms of the original cartesian coordinate system.

$$\iint f(\mathrm{x},\mathrm{y})\,\mathrm{dxdy} = \iint f(\xi,\eta)\mathrm{J}\,\mathrm{d}\xi\,\mathrm{d}\eta \tag{9}$$

Evaluating Local Stiffness Matrices

Now that we have defined the shape functions and a transformation from the global coordinate system to an element local system, we return to the local stiffness matrices. Examining Equation 5, and using the Jacobian matrix as in Equaiton 7, we can find the derivatives of the shape functions as

$$\begin{bmatrix} \frac{\partial N_m^j}{\partial x} \\ \frac{\partial N_m^j}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \frac{\partial N_m^j}{\partial x} \\ \frac{\partial N_m^j}{\partial y} \end{bmatrix} [\mathbf{J}] = \begin{bmatrix} \frac{\partial N_m^j}{\partial \xi} \\ \frac{\partial N_m^j}{\partial \eta} \end{bmatrix}$$

thus

$$\begin{bmatrix} \frac{\partial N_m^j}{\partial x} \\ \frac{\partial N_m^j}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial N_m^j}{\partial \xi} \\ \frac{\partial N_m^j}{\partial \eta} \end{bmatrix} [\mathbf{J}]^{-1}$$

or

$$\begin{Bmatrix} \frac{\partial N_n^j}{\partial x} \\ \frac{\partial N_n^j}{\partial y} \end{Bmatrix} = [\mathbf{J}]^{-\mathrm{T}} \begin{Bmatrix} \frac{\partial N_n^j}{\partial \xi} \\ \frac{\partial N_n^j}{\partial \eta} \end{Bmatrix}$$

where $j = 1, 2, ...N$ is the number of the element and $m = 1, 2, 3$ is the number of the corresponding element local shape function. Using the matrix notation developed above, we can now express Equation 5 as

$$K_{m,n}^j = \iint \begin{bmatrix} \frac{\partial N_m^j}{\partial \xi} \\ \frac{\partial N_m^j}{\partial \eta} \end{bmatrix} [\mathbf{J}]^{-1} [\mathbf{J}]^{-\mathrm{T}} \begin{Bmatrix} \frac{\partial N_n^j}{\partial \xi} \\ \frac{\partial N_n^j}{\partial \eta} \end{Bmatrix} \mathrm{J}\, \mathrm{d}\xi\, \mathrm{d}\eta \tag{10}$$

Using Equation 8 we can easily evaluate the local derivatives of the element shape functions. which are:

$$\begin{aligned} \frac{\partial N_1}{\partial \xi} &= -1 & \frac{\partial N_1}{\partial \eta} &= -1 \\ \frac{\partial N_2}{\partial \xi} &= 1 & \frac{\partial N_2}{\partial \eta} &= 0 \\ \frac{\partial N_3}{\partial \xi} &= 0 & \frac{\partial N_3}{\partial \eta} &= 1 \end{aligned} \tag{11}$$

To obtain expressions for $[\mathbf{J}]^{-1}$ and $[\mathbf{J}]^{-\mathrm{T}}$ we start with the Jacobian as in Equation 7 and perform the standard inversion and transpose for a two by two matrix. This yields

$$\begin{aligned} \mathbf{J} &= \begin{bmatrix} \Delta \mathrm{x}_2 & \Delta \mathrm{x}_3 \\ \Delta \mathrm{y}_2 & \Delta \mathrm{y}_3 \end{bmatrix} \\ \mathbf{J}^{-1} &= \frac{1}{\mathrm{J}} \begin{bmatrix} \Delta \mathrm{y}_3 & -\Delta \mathrm{x}_3 \\ -\Delta \mathrm{y}_2 & \Delta \mathrm{x}_2 \end{bmatrix} \\ \mathbf{J}^{-\mathrm{T}} &= \frac{1}{\mathrm{J}} \begin{bmatrix} \Delta \mathrm{y}_3 & -\Delta \mathrm{y}_2 \\ -\Delta \mathrm{x}_3 & \Delta \mathrm{x}_2 \end{bmatrix} \end{aligned}$$

where J is the determinant of the Jacobian given by

$$\mathrm{J} = \Delta\mathrm{x}_2\Delta\mathrm{y}_3 - \Delta\mathrm{x}_3\Delta\mathrm{y}_2.$$

Using these expressions, one can compute the product $\mathbf{J}^{-1}\mathbf{J}^{-\mathrm{T}}$ as

$$\begin{aligned}
\mathbf{J}^{-1}\mathbf{J}^{-\mathrm{T}} &= \frac{1}{\mathrm{J}^2}\begin{bmatrix} \Delta\mathrm{y}_3 & -\Delta\mathrm{x}_3 \\ -\Delta\mathrm{y}_2 & \Delta\mathrm{x}_2 \end{bmatrix}\begin{bmatrix} \Delta\mathrm{y}_3 & -\Delta\mathrm{y}_2 \\ -\Delta\mathrm{x}_3 & \Delta\mathrm{x}_2 \end{bmatrix} \\
&= \frac{1}{\mathrm{J}^2}\begin{bmatrix} \Delta\mathrm{x}_3^2 + \Delta\mathrm{y}_3^2 & -(\Delta\mathrm{x}_2\Delta\mathrm{x}_3 + \Delta\mathrm{y}_2\Delta\mathrm{y}_3) \\ -(\Delta\mathrm{x}_2\Delta\mathrm{x}_3 + \Delta\mathrm{y}_2\Delta\mathrm{y}_3) & \Delta\mathrm{x}_2^2 + \Delta\mathrm{y}_2^2 \end{bmatrix} \\
&= \frac{1}{\mathrm{J}^2}\begin{bmatrix} \lambda_3 & -\lambda_{23} \\ -\lambda_{23} & \lambda_2 \end{bmatrix}
\end{aligned} \tag{12}$$

where the notataion $\lambda_2 = \Delta\mathrm{x}_2^2 + \Delta\mathrm{y}_2^2$ has been introduced for convenience. The element stiffness matrix now becomes

$$K^j_{n,m} = \iint \frac{1}{\mathrm{J}} \begin{bmatrix} \dfrac{\partial N^j_n}{\partial\xi} \\ \dfrac{\partial N^j_n}{\partial\eta} \end{bmatrix} \begin{bmatrix} \lambda_3 & -\lambda_{23} \\ -\lambda_{23} & \lambda_2 \end{bmatrix} \begin{Bmatrix} \dfrac{\partial N^j_m}{\partial\xi} \\ \dfrac{\partial N^j_m}{\partial\eta} \end{Bmatrix} \mathrm{d}\xi\ \mathrm{d}\eta$$

Notice that all the quantities involved in the integral are constant for a given element. Thus, evaluating the integral over a triangular element simply gives us 1/2 times the constant integrand. Using Equation 11 we see that the components of the element stiffness matrix as:

$$\begin{aligned}
\mathrm{J}\ K^j_{1,1} &= \lambda_2 + \lambda_3 - 2\lambda_{23} & \mathrm{J}\ K^j_{1,2} &= \lambda_{23} - \lambda_3 & \mathrm{J}\ K^j_{1,3} &= \lambda_{23} - \lambda_2 \\
\mathrm{J}\ K^j_{2,1} &= \lambda_{23} - \lambda_3 & \mathrm{J}\ K^j_{2,2} &= \lambda_3 & \mathrm{J}\ K^j_{2,3} &= -\lambda_{23} \\
\mathrm{J}\ K^j_{3,1} &= \lambda_{23} - \lambda_2 & \mathrm{J}\ K^j_{3,2} &= -\lambda_{23} & \mathrm{J}\ K^j_{3,3} &= \lambda_2
\end{aligned}$$

with

$$\mathrm{J} = \Delta\mathrm{x}_2\Delta\mathrm{y}_3 - \Delta\mathrm{x}_3\Delta\mathrm{y}_2$$

Assembly of the Global Stiffness Matrix and Load Vector

To assemble the global stiffness matrix we must place componenets of the local stiffness matrices in their appropriate locations in the gobal stiffness matrix. To accomplish this, we must first have a mapping from the local node numbering to the global node numbering. For the case of our simple triangular element, this means that for each local node ($il = 1, 2, 3$) we must have a global node number ($ig \in [1, ...N]$), where N is the total number of nodes. Once we have this mapping, all that remains is to cycle through each element and the local stiffness matrix componenets to the global stiffness matrix component as follows:

$$K(ig, jg) = K(ig, jg) + K^j_{local}(ie, je) \quad \begin{cases} \text{for } ie = 1, 2, 3 \\ \text{for } je = 1, 2, 3 \\ \text{for } j = 1, ...N \end{cases}$$

To assemble the load vector, we first evaluate the integral from Equation 4

$$\begin{aligned}\int_{\Omega} \nu \mathrm{d}\Omega = &\sum_{j=1}^{N} \iint p_k^j N_k^j \mathrm{J}\, \mathrm{d}\xi\, \mathrm{d}\eta \quad \mathrm{k} = 1,2,3 \\ = &\lfloor p_k^j \rfloor \{Q_k^j\}\end{aligned}$$

where

$$Q_k^j = \iint N_k^j \mathrm{J}\, \mathrm{d}\xi\, \mathrm{d}\eta = \frac{1}{6}\,\mathrm{J}$$

Note that integration of each shape function over a triangular element comes out to be the same constant for a given element. Now, assembly of a global load vector follow much the same process as for the stiffness matrix, where the following rule is used:

$$Q(ig) = Q(ig) + Q_{local}^j(ie) \quad \begin{cases} \text{for } ie = 1,2,3 \\ \text{for } j = 1,...N \end{cases}$$

Solution of the FEM System of Equations

Returning to the Symmetric Weak form of the governing equation, we now may express the integrals in the global domain in terms of summations of the integrals in the element local domains. The result is

$$\lfloor p_n \rfloor \left[K_{n,m}\right] \{q_m\} = \lfloor p_k \rfloor \{Q_k\}$$

Here we have left of the element index j to signify these as global vectors and matrices. Since the coefficient vector p^j is arbitrary, we chose it to be a series of linearly independent vectors such that we end up with the following system of equations.

$$\left[K_{n,m}\right] \{q_m\} = \{Q_n\}$$

Application of the Boundary Conditions

The boundary condition is a Dirichilet type, specifically $\phi = 0$ on the boundary. To apply this boundary condition to our system of equations, we must simply modify the global load vector and stiffness matrix to fix the values of ϕ at the boundary nodes. For instance. Suppose global node n is a boundary node. Set all of the coefficients corresponding to the equation for node n in the global stiffness matrix equal to zero except for the diagonal which is set to one. For the load vecotor, we set the nth component also to zero. This

is shown below, and must be repeated for each boundary node.

$$\begin{bmatrix} k_{1,1} & \dots & k_{1,n} & \dots & k_{1,N} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ k_{n,1} & \dots & k_{n,n} & \dots & k_{n,N} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ k_{N,1} & \dots & k_{N,n} & \dots & k_{N,N} \end{bmatrix} \begin{Bmatrix} q_1 \\ \vdots \\ q_n \\ \vdots \\ q_N \end{Bmatrix} = \begin{Bmatrix} Q_1 \\ \vdots \\ Q_n \\ \vdots \\ Q_N \end{Bmatrix}$$

$$\Downarrow$$

$$\begin{bmatrix} k_{1,1} & \dots & k_{1,n} & \dots & k_{1,N} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & \dots & 1 & \dots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ k_{N,1} & \dots & k_{N,n} & \dots & k_{N,N} \end{bmatrix} \begin{Bmatrix} q_1 \\ \vdots \\ q_n \\ \vdots \\ q_N \end{Bmatrix} = \begin{Bmatrix} Q_1 \\ \vdots \\ 0 \\ \vdots \\ Q_N \end{Bmatrix}$$

We may now solve the modified system of equations for q, the nodal values of the solution variable ϕ

$$\{q_m\} = \left[K^*_{n,m}\right]^{-1} \{Q^*_n\}$$

2.2.2 Fortran Code provided by Roy Culver

The following is Fortran code for the FEM problem with needed data files.

Zip file

2.3 Documents by Paul Nylander (from MAE 207 spring 2006 class)

Local contents

New files send on Nov 27,2006. This ZIP send that contains all the material by Paul Nylander.

Zip file

These below are the notebooks individually each converted to PDF

2.3.1 solve Poisson equation using FEM on arbitrary unstructured meshes

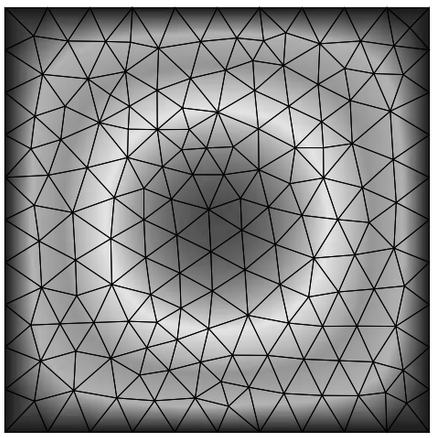
Project.nb 1

Here is some code to solve Poisson's Equation using the Finite Element Method (FEM) on abritrary unstructured meshes:

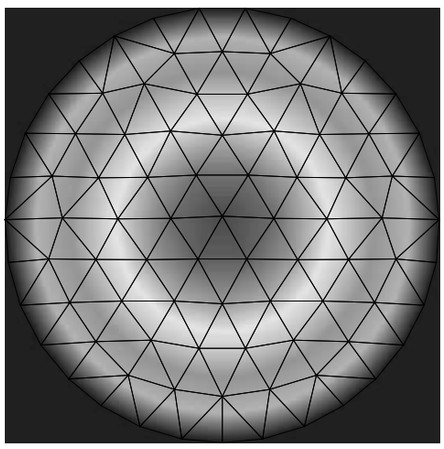
```
ImportMesh[folder_] := Module[{}, nodes = Import[StringJoin[folder, "nodes.txt"], "CSV"];
    elements = Import[StringJoin[folder, "elements.txt"], "CSV"];];
SolvePoissonFEM[] := Module[{}, jmax = Length[elements]; xlist = Map[#[[1]] &, nodes]; ylist = Map[#[[2]] &, nodes];
  {x1, x2} = {Min[xlist], Max[xlist]}; {y1, y2} = {Min[ylist], Max[ylist]}; nxy = Length[nodes]; Q = Table[0, {nxy}];
  Klocal = Map[Module[{}, Do[{Δx_i, Δy_i, void} = nodes[[#[[i]]]] - nodes[[#[[1]]]], {i, 2, 3}]; λ_2 = Δx_2^2 + Δy_2^2;
      λ_23 = Δx_2 Δx_3 + Δy_2 Δy_3; λ_3 = Δx_3^2 + Δy_3^2; d = Δx_2 Δy_3 - Δx_3 Δy_2; Do[Q[[#[[n]]]] += d/6, {n, 1, 3}];
      Chop[1/d ((λ_2 + λ_3 - 2 λ_23   λ_23 - λ_3   λ_23 - λ_2
                 λ_23 - λ_3           λ_3         -λ_23
                 λ_23 - λ_2          -λ_23        λ_2       ))]] &, elements]; Kglobal = Table[0, {nxy}, {nxy}];
  Do[element = elements[[j]]; Do[Kglobal[[element[[m]], element[[n]]]] += Klocal[[j, m, n]], {m, 1, 3}, {n, 1, 3}],
    {j, 1, jmax}]; Do[{x, y, boundary} = nodes[[i]];
    If[boundary == 1, Kglobal[[i]] = Table[If[j == i, 1, 0], {j, 1, nxy}]; Q[[i]] = 0],
    {i, 1, nxy}]; q = Chop[LinearSolve[Kglobal, Q]]];
Interpolate[x_, y_] := Module[{x1, y1}, {x1, y1} = plist[[1]]; Do[{Δx_i, Δy_i} = plist[[i]] - {x1, y1}, {i, 2, 3}];
    {ξ, η} = ((x - x1) {Δy_3, -Δy_2} + (y - y1) {-Δx_3, Δx_2}) / (Δx_2 Δy_3 - Δx_3 Δy_2); q[[element]].{1 - ξ - η, ξ, η}];
PlotSolution[] := Module[{}, r = (y2 - y1) / (x2 - x1); ny = 275; nx = Round[ny / r];
    image = Table[0, {ny}, {nx}]; Do[element = elements[[j]]; xlist = Map[#[[1]] &, nodes[[element]]];
      ylist = Map[#[[2]] &, nodes[[element]]]; plist = Transpose[{xlist, ylist}];
      xIntersect[{{x1_, y1_}, {x2_, y2_}}] := If[y1 == y2, ∞, x1 + (y - y1) (x2 - x1) / (y2 - y1)];
      Do[y = y1 + (y2 - y1) (i - 1) / (ny - 1); jlist = Round[(nx - 1)
            (Select[Map[xIntersect, Partition[plist, 2, 1, 1]], (Min[xlist] ≤ # ≤ Max[xlist]) &] - x1) / (x2 - x1) + 1];
        If[jlist =!= {}, Do[x = x1 + (x2 - x1) (jj - 1) / (nx - 1); image[[i, jj]] = Interpolate[x, y],
          {jj, Max[1, Min[jlist]], Min[nx, Max[jlist]]}]], {i, Max[1, Round[(ny - 1) (Min[ylist] - y1) / (y2 - y1) + 1]],
          Min[ny, Round[(ny - 1) (Max[ylist] - y1) / (y2 - y1) + 1]]}], {j, 1, jmax}];
    ListDensityPlot[image, AspectRatio → Automatic, Mesh → False, Frame → False,
      ColorFunction → (Hue[2 (1 - #) / 3] &), AspectRatio → r,
      ImageSize → {nx, ny}, Epilog → Table[element = elements[[j]];
        Line[Map[({x, y, boundary} = nodes[[#]]; {nx (x - x1) / (x2 - x1), ny (y - y1) / (y2 - y1)}) &,
          Append[element, element[[1]]]], {j, 1, jmax}]]];
folder = "C:/School/Engineering/ME207 - Computer Modeling/Project/";
folder = "";
```

Printed by Mathematica for Students

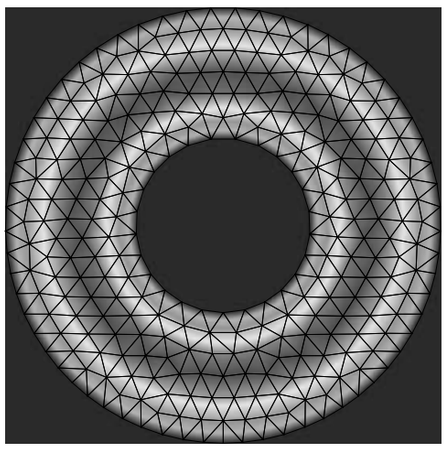
```
ImportMesh[StringJoin[folder, "Square/"]]; SolvePoissonFEM[]; PlotSolution[];
```



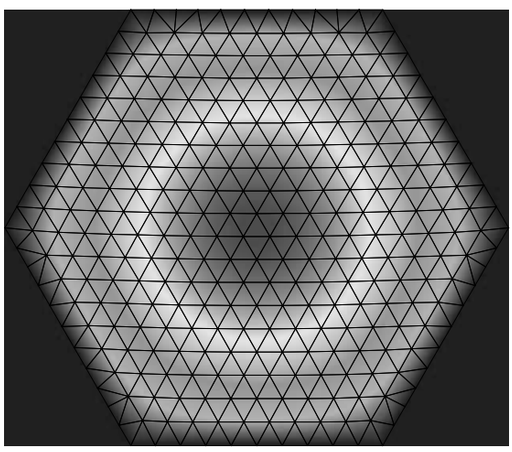
```
ImportMesh[StringJoin[folder, "Circle/"]]; SolvePoissonFEM[]; PlotSolution[];
```



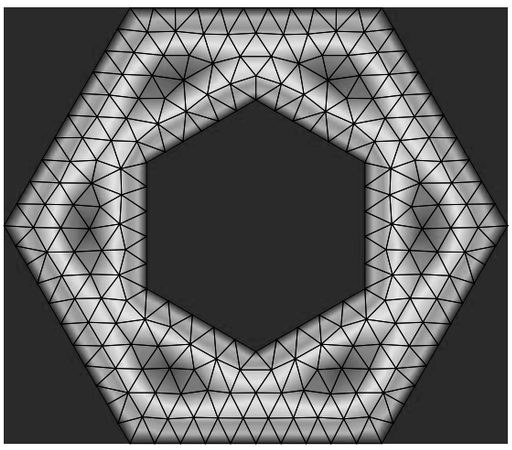
```
ImportMesh[StringJoin[folder, "Donut/"]]; SolvePoissonFEM[]; PlotSolution[];
```



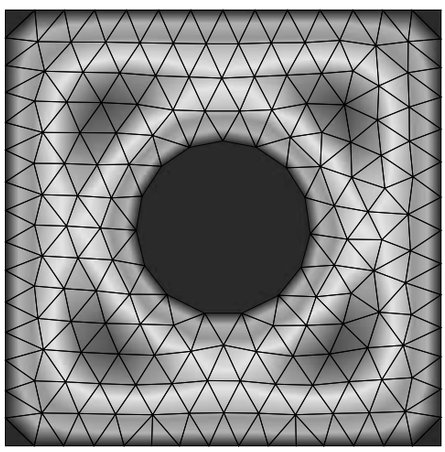
```
ImportMesh[StringJoin[folder, "Hex/"]]; SolvePoissonFEM[]; PlotSolution[];
```



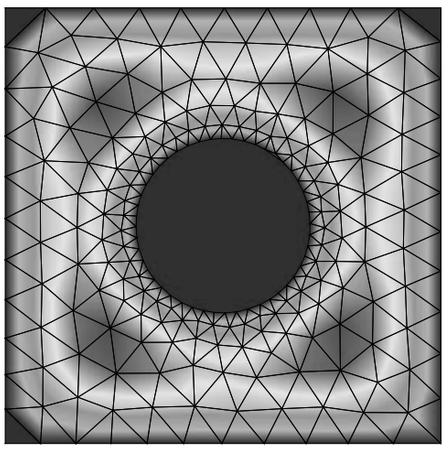
```
ImportMesh[StringJoin[folder, "HexDonut/"]]; SolvePoissonFEM[]; PlotSolution[];
```



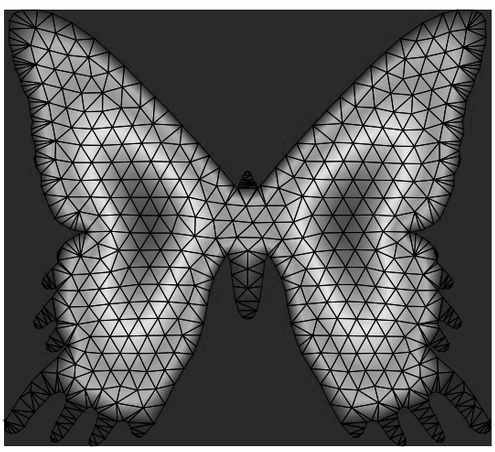
```
ImportMesh[StringJoin[folder, "SquareDonut/"]]; SolvePoissonFEM[]; PlotSolution[];
```



```
ImportMesh[StringJoin[folder, "SquareDonut2/"]]; SolvePoissonFEM[]; PlotSolution[];
```



```
ImportMesh[StringJoin[folder, "Sunset/"]]; SolvePoissonFEM[]; PlotSolution[];
```



2.3.2 Notes on Analytical solution

Analytical Solution.nb 1

Paul Nylander

MAE 207, Methods of Computer Modeling in Engineering and the Sciences
Isotropic Rectangular Beam Torsion

beam: http://scienceworld.wolfram.com/physics/Beam.html

■ **Finite Element Method (FEM) with Triangular Elements**

■ **Roy's notes**

Poisson equation : $\nabla^2\phi = \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} = f \to (\nabla u\, v)_{\partial\Omega} - \int_\Omega (\nabla u \nabla v - f v)\, d\Omega = 0$

coordinates within triangle : ξ_1, ξ_2

linearized solution : $\phi \approx \alpha_1 + \alpha_2 \xi_1 + \alpha_3 \xi_2$

node 1 : $(\xi_1)_{node1} = \xi_{11} \stackrel{?}{=} 0$, $(\xi_2)_{node1} = \xi_{21} \stackrel{?}{=} 0 \to q_1 \equiv \phi_{node1} = \alpha_1 + \alpha_2 \xi_{11} + \alpha_3 \xi_{21} \stackrel{?}{=} \alpha_1$

node 2 : $(\xi_1)_{node2} = \xi_{12} \stackrel{?}{=} 1$, $(\xi_2)_{node2} = \xi_{22} \stackrel{?}{=} 0 \to q_2 \equiv \phi_{node2} = \alpha_1 + \alpha_2 \xi_{12} + \alpha_3 \xi_{22} \stackrel{?}{=} \alpha_1 + \alpha_2$

node 3 : $(\xi_1)_{node3} = \xi_{13} \stackrel{?}{=} 0$, $(\xi_2)_{node3} = \xi_{23} \stackrel{?}{=} 1 \to q_3 \equiv \phi_{node3} = \alpha_1 + \alpha_2 \xi_{13} + \alpha_3 \xi_{23} \stackrel{?}{=} \alpha_1 + \alpha_3$

$u = q_1 N_1 + q_2 N_2 + q_3 N_3 \stackrel{?}{=} \phi$

$$\nabla N_{i?} \cdot \nabla N_k = \begin{pmatrix} \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial x} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_2}{\partial y} & \frac{\partial N_3}{\partial y} \end{pmatrix} \cdot \begin{pmatrix} \partial \frac{*}{\partial} * & \partial \frac{*}{\partial} * \\ \partial \frac{*}{\partial} * & \partial \frac{*}{\partial} * \\ \partial \frac{*}{\partial} * & \partial \frac{*}{\partial} * \end{pmatrix}$$

what is N?

■ **class notes: 5/1/06**

```
Solve[{0 == b x1 + c x2 + d x3, 1 == b x1 + c x2 + d x3, 0 == b x1 + c x2 + d x3}, {x1, x2, x3}]
```

■ **Analytical Solution, Nasser's notes**

Pandlet

$\boxed{\nabla^2\phi = -2\,G\,K}$, angle at twist : $\frac{d\alpha}{dt}$, $\frac{T}{G\,J}$

$$\phi^{(\alpha,\eta)} = \frac{32\,G\,a^3}{\pi^3} \sum_{n\ odd}^{\infty} \frac{1}{n^3} (-1)^{\frac{n-1}{2}} \left(1 - \frac{\text{Cosh}[b]}{\text{Cosh}[b]}\right)$$

Fig1

twist rate : $\frac{d\alpha}{dz} = k\,z$, rectangular beam dimensions : $a \times b$, applied torque : T

modulus of rigidity : $G = \frac{\tau}{\gamma} = \frac{E}{2\,(1+\nu)}$, Young's modulus : E,

Poisson's ratio : ν, shearing stress : τ, shearing strain : γ

Printed by Mathematica for Students

torsion constant : $J = \frac{16}{3} a^3 b \left(1 - \frac{196\,a}{b\,\pi^5} \sum_{n=1,3,5\ldots}^{\infty} \frac{1}{n^5} \text{Tanh}\left[\frac{n\,\pi\,b}{2\,a}\right]\right) = 2.18498$

Prandtl stress function : Φ, Poisson equation : $\nabla^2 \Phi = \frac{\partial^2 \Phi}{\partial x^2} + \frac{\partial^2 \Phi}{\partial y^2} = -2\,G\,K$, $\Phi_{\text{boundary}} = 0$

Timoshenko : $\Phi = \frac{32\,G\,k\,a}{\pi^3} \sum_{n=1,3,5\ldots}^{\infty} \frac{1}{n^3} (-1)^{\frac{n-1}{2}} \left(1 - \frac{\text{Cosh}\left[\frac{n\,\pi\,y}{2\,a}\right]}{\text{Cosh}\left[\frac{n\,\pi\,b}{2\,a}\right]}\right) \text{Cos}\left[\frac{n\,\pi\,x}{2\,a}\right]$,

linear twist : $k = \frac{T}{G\,J}$

shear stress tensor field : $\tau_{yz} = -\frac{\partial \Phi}{\partial x}$, $\tau_{xz} = \frac{\partial \Phi}{\partial y}$, $\tau_{yx} = 0$, $\sigma_x = \sigma_y = \sigma_z = 0$

principle stresses : $\{\sigma_1, \sigma_2, \sigma_3\} = \text{Eigenvalues}\left[\begin{pmatrix} \sigma_{xx} & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \sigma_{yy} & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_{zz} \end{pmatrix}\right]$

Von Mises stress : $\sigma_v = \sqrt{((\sigma_1 - \sigma_2)^2 + (\sigma_2 - \sigma_3)^2 + (\sigma_3 - \sigma_1)^2)/2} = (3\,(\tau_{xz}^2 + \tau_{yz}^2))^{1/2}$

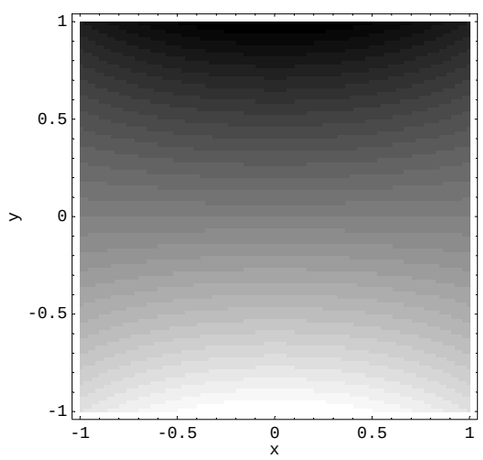
strain tensor field (material constitutive relation)

$\epsilon_x = \frac{1}{E}(\sigma_x - \nu\,(\sigma_y + \sigma_z)) = 0$, $\epsilon_y = \frac{1}{E}(\sigma_y - \nu\,(\sigma_x + \sigma_z)) = 0$, $\epsilon_z = \frac{1}{E}(\sigma_z - \nu\,(\sigma_x + \sigma_y)) = 0$

$\gamma_{xy} = \frac{\tau_{xy}}{G} = 0$, $\gamma_{yz} = \frac{\tau_{yz}}{G}$, $\gamma_{xz} = \frac{\tau_{xz}}{G}$

angle of twist : $\alpha = k\,z$

```
a = b = 2; T = G = 1; J = 2.1849801331564294`; k = T/(G J);
Φ = (32 G k a)/π^3 Sum[1/n^3 (-1)^((n-1)/2) (1 - Cosh[n π y/(2 a)]/Cosh[n π b/(2 a)]) Cos[n π x/(2 a)], {n, 1, 10, 2}]; τyz = -∂x Φ; τxz = ∂y Φ;
DensityPlot[τxz, {x, -a/2, a/2}, {y, -a/2, a/2},
  PlotPoints → 100, Mesh → False, FrameLabel → {"x", "y"}];
```



Printed by Mathematica for Students

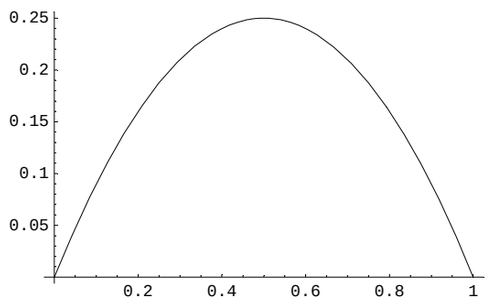
```
(* http://www.efunda.com/formulae/solid_mechanics/mat_mechanics/strain.cfm *)
```

$$\gamma_{yz} \equiv \frac{\partial w}{\partial y} + \frac{\partial v}{\partial z}, \quad \gamma_{xz} \equiv \frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}, \quad \gamma_{xy} \equiv \frac{\partial v}{\partial x} + \frac{\partial u}{\partial y}$$

```
x = y = 0; (1/2 ((x/(a/2))^2 + (y/(b/2))^2))^(1/2)
```

```
0
```

```
Plot[g (1 - g), {g, 0, 1}];
```



Printed by Mathematica for Students

```
PlotColor[hue_] :=
  RGBColor @@ If[hue == 1, {1, 0, 0}, Module[{x = Mod[4 hue, 1]}, Switch[Mod[Floor[4 hue], 4],
      0, {0, x, 1}, 1, {0, 1, 1 - x}, 2, {x, 1, 0}, 3, {1, 1 - x, 0}]]];
a = 2.0; b = 2.0; L = 10.0; G = 1; dx = a / 8; dy = b / 8; dz = L / 40; J = 2.1849801331564294`;
Clear[x, y, z, T]; k = T / (G J);
Φ = (32 G k a / π^3) Sum[(1 / n^3) (-1)^((n-1)/2) (1 - Cosh[n π y / (2 a)] / Cosh[n π b / (2 a)]) Cos[n π x / (2 a)], {n, 1, 10, 2}];
τyz = -∂x Φ; τxz = ∂y Φ; γyz = τyz / G; γxz = τxz / G;
Rotate[{x_, y_, z_}, θ_] := {x Cos[θ] - y Sin[θ], x Sin[θ] + y Cos[θ], z};
animation = {}; scale2 = 10.0; Tmax = scale2; dt = 1.0 / 360;
Prepare[T1_, z1_] := Module[{}, α = k z1 / scale2 /. T → T1;
   scale = (x^2 + y^2) / ((x + γxz / 2)^2 + (y + γyz / 2)^2)^(1/2) /. {x → a / 2, y → b / 2}];
Calculate[T1_, {x1_, y1_, z1_}] :=
  {Rotate[{scale (x + γxz / 2), scale (y + γyz / 2), z1 + scale (x γxz + y γyz) / 2}, α -
      ArcTan[(1 + 0.09576080145730925` T) / (1 - 0.09332536922126035` T)] - π / 4], (3 (τxz^2 + τyz^2))^(1/2)} /. {x → x1, y → y1, T → T1};
Do[slices = {}; Show[Graphics3D[{EdgeForm[], Table[T = Tmax (1 - Cos[2 π t]) / 2;
      Prepare[T, z]; slice1 = slice2; slice2 = Map[Calculate[T, {#[[1]], #[[2]], z}] &,
        Flatten[{Table[{x, -b / 2}, {x, -a / 2, a / 2 - dx, dx}], Table[{a / 2, y},
           {y, -b / 2, b / 2 - dy, dy}], Table[{x, b / 2}, {x, a / 2, dx - a / 2, -dx}], Table[
           {-a / 2, y}, {y, b / 2, dy - b / 2, -dy}]}, 1]]; slices = Append[slices, slice2];
      n = Length[slice2]; If[z == 0, Polygon[Map[#[[1]] &, slice2]], Table[
        quad = {slice1[[i]], slice2[[i]], slice2[[Mod[i, n] + 1]], slice1[[Mod[i, n] + 1]]};
        {SurfaceColor[PlotColor[(Plus @@ Map[#[[2]] &, quad] / 4) / 4.632030522701348]],
         Polygon[Map[#[[1]] &, quad]]}, {i, 1, n}]], {z, 0, L, dz}]},
    ViewPoint → {-1, 1, 1}, ViewVertical → {0, 1, 0}, Axes → True,
    AxesLabel → {"x", "y", "z"},
    PlotRange → {{-a, a}, {-b, b}, {0, L}}]];
  animation = Append[animation, slices],
  {t,
   0,
   0.5,
   dt}];
```

```
Zero[x_, n_] := Module[{str = ToString[x]},
   StringJoin @@ Append[Table["0", {n - StringLength[str]}], str]];
POVFormat[x_Real] := ToString[x, CForm]; POVFormat[x_Integer] := ToString[x];
POVFormat[{{x_, y_, z_}, σ_}] := StringJoin["<", POVFormat[x],
   ",", POVFormat[y], ",", POVFormat[z], ",", POVFormat[σ], ">"];
POVFormat[xlist_List] := StringJoin["{", StringJoin @@
    Map[POVFormat[#] <> "," &, Drop[xlist, -1]], POVFormat[Last[xlist]], "}"];
TakeLoop[list_, n_] := Take[Join[list, list], n]; nslice = Length[animation[[1]]];
WriteVar[name_, x_] := WriteString[wf, StringJoin["#declare ", name, "=array[",
    ToString[Length[x]], "][", ToString[Length[x[[1]]]], "] ", POVFormat[x], ";\n"]];
Do[T = Tmax (1 - Cos[2 π t]) / 2; frame = Round[t / dt + 1]; slices = animation[[frame]]; wf =
   OpenWrite[StringJoin["C:/Files/PovRay/Physics/Torsion/", Zero[frame, 3], ".inc"]];
  Do[WriteVar[StringJoin["wall", ToString[iwall]],
    Table[TakeLoop[slices[[islice]], 8 {iwall - 1, iwall} + 1], {islice, 1, nslice}]],
   {iwall, 1, 4}]; Prepare[T, 0]; WriteVar["cap1",
   Table[Calculate[T, {x, y, 0}], {y, -b / 2, b / 2, dy}, {x, -a / 2, a / 2, dx}]];
  Prepare[T, L]; WriteVar["cap2", Table[Calculate[T, {x, y, L}],
    {y, -b / 2, b / 2, dy}, {x, -a / 2, a / 2, dx}]];
  Close[wf], {t, 0, 0.5, dt}];

CopyScaleExport["C:/Files/Animated Gifs/Torsion", 140, "*.bmp"];
CopyScaleExport["C:/Files/Animated Gifs/Torsion", 70, "*.bmp"];

Export["C:/Files/Animated Gifs/Torsion/Torsion.gif", Map[Import[#[[1]]] &, Partition[
    FileNames["*.bmp", "C:/Files/Animated Gifs/Torsion"], 1, 1]], ConversionOptions →
   {"AnimationDisplayTime" → 0, "Loop" → True, "GlobalColorReduction" → True}];
```

Printed by Mathematica for Students

2.3.3 Notes on FVM

FVM.nb 1

Finite Volume Method (FVM)

■ **Notes**

```
N := a + b ξ + c η + d ξ η + e ξ^2 + f η^2 + g ξ^2 η + h ξ η^2 + i ξ^2 η^2;
x := x1 + (x2 - x1) ξ + (x3 - x1) η; y := y1 + (y2 - y1) ξ + (y3 - y1) η; J = ( ∂ξ x  ∂η x ; ∂ξ y  ∂η y );
Inverse[J].( ∂ξ N ; ∂η N ) // MatrixForm // FullSimplify

Clear[N]; N[ξ_, η_] := a + b ξ + c η + d ξ η + e ξ^2 + f η^2 + g ξ^2 η + h ξ η^2 + i ξ^2 η^2;

dNdx[ξ_, η_] :=
  ((-y1 + y3) (b + η (d + h η) + 2 (e + η (g + i η)) ξ) + (x1 - x3) (c + 2 f η + ξ (d + g ξ + 2 η (h + i ξ))))
    / (x3 (y1 - y2) + x1 (y2 - y3) + x2 (-y1 + y3));
dNdy[ξ_, η_] :=
  ((y1 - y2) (b + η (d + h η) + 2 (e + η (g + i η)) ξ) + (-x1 + x2) (c + 2 f η + ξ (d + g ξ + 2 η (h + i ξ))))
    / (x3 (y1 - y2) + x1 (y2 - y3) + x2 (-y1 + y3));

Solve[{N[0, 0] == q1, dNdx[0, 0] == q4, dNdy[0, 0] == q7, N[1, 0] == q2, dNdx[1, 0] == q5,
  dNdy[1, 0] == q8, N[0, 1] == q3, dNdx[0, 1] == q6, dNdy[0, 1] == q9}, {a, b, c, d, e, f, g, h, i}]

{}
```

$$N = a + b\,\xi + c\,\eta + d\,\xi\,\eta + e\,\xi^2 + f\,\eta^2 + g\,\xi^2\,\eta + h\,\xi\,\eta^2 + i\,\xi^2\,\eta^2$$

$$a = q_1$$

$$b = (-x1 + x2)\,q_4 + (-x1 + x3)\,q_7$$

$$c = (-y1 + y2)\,q_4 + (-y1 + y3)\,q_7$$

$$d = d$$

$$e = -q_1 + q_2 + (x1 - x2)\,q_4 + (x1 - x3)\,q_7$$

$$f = -q_1 + q_3 + (y1 - y2)\,q_4 + (y1 - y3)\,q_7$$

$$g = -d + \frac{1}{x1 - x3}\Big((-2\,(y1 - y3)\,q_1 + 2\,(y1 - y3)\,q_2 - (x2\,y1 + x3\,y1 - x3\,y2 - x2\,y3 + x1\,(-2\,y1 + y2 + y3))\,q_4 + (x3\,y1 + x1\,y2 - x3\,y2 - x1\,y3 + x2\,(-y1 + y3))\,q_5) + 2\,(y1 - y3)\,q_7\Big)$$

$$h = -d + \frac{1}{y1 - y3}\Big(-2\,(x1 - x3)\,q_1 + 2\,(x1 - x3)\,q_3 - (x2\,y1 + x3\,y1 - x3\,y2 - x2\,y3 + x1\,(-2\,y1 + y2 + y3))\,q_4 + (-x3\,y1 - x1\,y2 + x3\,y2 + x2\,(y1 - y3) + x1\,y3)\,q_6) + 2\,(x1 - x3)\,q_7\Big)$$

$$i = i$$

---

pg. 805

shape functions : $N_1 = \xi 3 = 1 - \xi - \eta$, $N_2 = \xi 1 = \xi$, $N_3 = \xi 2 = \eta$

Printed by Mathematica for Students

```
u := a ξ1 + b ξ2 + c ξ3 + d ξ1 ξ2 + e ξ2 ξ3 + f ξ3 ξ1 +
   g (ξ1^2 ξ2 + ξ1 ξ2 ξ3 (3 (1 - μ3) ξ1 - (1 + 3 μ3) ξ2 + (1 + 3 μ3) ξ3) / 2) +
   h (ξ2^2 ξ3 + ξ1 ξ2 ξ3 (3 (1 - μ1) ξ2 - (1 + 3 μ1) ξ3 + (1 + 3 μ1) ξ1) / 2) +
   i (ξ3^3 ξ1 + ξ1 ξ2 ξ3 (3 (1 - μ2) ξ3 - (1 + 3 μ2) ξ1 + (1 + 3 μ2) ξ2) / 2);
μ1 = (ℓk^2 - ℓj^2) / ℓi^2
```

see Patankar, 1985
Malagasekara

- **http://www.amazon.com/gp/offer-listing/0582218845/sr=8-1/qid=1157653744/ref=sr_1_1/002-2175139-6340059?ie=UTF8&s=books**
  zdhan@care.eng.uci.edu

2.3.4 FEM notes. Triangle and quad element

■ **Notes: http://bessie.che.uc.edu/tlb/teach/undergrad/chem381f97/math/node7.html**

soln = $\int_0^1 \int_0^1 \{\partial_\xi u, \partial_\eta u\}\, J^{-T} J^{-1} \begin{pmatrix} \partial_\xi v \\ \partial_\eta v \end{pmatrix} \|J\|\, d\xi\, d\eta$

local stiffness matrix : $K^j_{mn} = \int_0^1 \int_0^1 \{\partial_\xi N_m, \partial_\eta N_m\}\, J^{-T} J^{-1} \begin{pmatrix} \partial_\xi N_n \\ \partial_\eta N_n \end{pmatrix} \|J\|\, d\xi\, d\eta$

■ **Triangle Element**

shape functions : $N_1 = 1 - \xi - \eta,\ N_2 = \xi,\ N_3 = \eta$

trial and test function : $u = \sum_{i=1}^{3} q_i N_i,\ v = \sum_{i=1}^{3} p_k N_k \to \frac{\partial u}{\partial \xi} = -q_1 + q_2,\ \frac{\partial v}{\partial \xi} = -p_1 + p_2,\ \frac{\partial u}{\partial \eta} = -q_1 + q_3,\ \frac{\partial v}{\partial \eta} = -p_1 + p_3$

$\vec{x} = \vec{x}_1 + (\vec{x}_2 - \vec{x}_1)\,\xi + (\vec{x}_3 - \vec{x}_1)\,\eta \to x = x_1 + (x_2 - x_1)\,\xi + (x_3 - x_1)\,\eta,\ y = y_1 + (y_2 - y_1)\,\xi + (y_3 - y_1)\,\eta$

$\Delta x_2 \equiv x_2 - x_1,\ \Delta y_2 \equiv y_2 - y_1,\ \Delta x_3 \equiv x_3 - x_1,\ \Delta y_3 \equiv y_3 - y_1$

Jacobian : $J = \begin{pmatrix} \partial_\xi x & \partial_\eta x \\ \partial_\xi y & \partial_\eta y \end{pmatrix} = \begin{pmatrix} \Delta x_2 & \Delta x_3 \\ \Delta y_2 & \Delta y_3 \end{pmatrix},\ \{dx, dy\} = J\,\{d\xi, d\eta\}$

eigenvalues : $\lambda_2 \equiv \Delta x_2^2 + \Delta y_2^2,\ \lambda_{23} \equiv \Delta x_2\,\Delta x_3 + \Delta y_2\,\Delta y_3,\ \lambda_3 \equiv \Delta x_3^2 + \Delta y_3^2$

$K^j_{mn} = \frac{1}{\Delta x_2\,\Delta y_3 - \Delta x_3\,\Delta y_2} \begin{pmatrix} \lambda_2 + \lambda_3 - 2\lambda_{23} & \lambda_{23} - \lambda_3 & \lambda_{23} - \lambda_2 \\ \lambda_{23} - \lambda_3 & \lambda_3 & -\lambda_{23} \\ \lambda_{23} - \lambda_2 & -\lambda_{23} & \lambda_2 \end{pmatrix}$

■ **Quad Element**

shape functions : $N_1 = (\eta - 1)\,(\xi - 1),\ N_2 = \xi\,(1 - \eta),\ N_3 = \eta\,(1 - \xi),\ N_4 = \xi\,\eta$

$\vec{x} = \vec{x}_1 + (\vec{x}_2 - \vec{x}_1)\,\xi + (\vec{x}_3 - \vec{x}_1)\,\eta + (\vec{x}_1 - \vec{x}_2 - \vec{x}_3 + \vec{x}_4)\,\xi\,\eta$

$\Delta x_2 \equiv x_2 - x_1,\ \Delta y_2 \equiv y_2 - y_1,\ \Delta x_3 \equiv x_3 - x_1,\ \Delta y_3 \equiv y_3 - y_1,\ \Delta x_a \equiv x_1 - x_2 - x_3 + x_4,\ \Delta y_a \equiv y_1 - y_2 - y_3 + y_4$

Jacobian : $J = \begin{pmatrix} \partial_\xi x & \partial_\eta x \\ \partial_\xi y & \partial_\eta y \end{pmatrix} = \begin{pmatrix} \Delta x_2 + \eta\,\Delta x_a & \Delta x_3 + \xi\,\Delta x_a \\ \Delta y_2 + \eta\,\Delta y_a & \Delta y_3 + \xi\,\Delta y_a \end{pmatrix}$

eigenvalues : $\lambda_2 \equiv \Delta x_2^2 + \Delta y_2^2,\ \lambda_{23} \equiv \Delta x_2\,\Delta x_3 + \Delta y_2\,\Delta y_3,\ \lambda_3 \equiv \Delta x_3^2 + \Delta y_3^2$

$K^j_{mn} = x$

■ **Program**

Calculate the x, y coordinates of the nodes :

```
a = b = 2.0; nx = 7; ny = 5;
nodes = Chop[Flatten[Table[{x, y}, {y, -b/2, b/2, b/(ny - 1)}, {x, -a/2, a/2, a/(nx - 1)}], 1]]
```

```
{{-1., -1.}, {-0.666667, -1.}, {-0.333333, -1.}, {0, -1.}, {0.333333, -1.}, {0.666667, -1.}, {1., -1.},
  {-1., -0.5}, {-0.666667, -0.5}, {-0.333333, -0.5}, {0, -0.5}, {0.333333, -0.5}, {0.666667, -0.5},
  {1., -0.5}, {-1., 0}, {-0.666667, 0}, {-0.333333, 0}, {0, 0}, {0.333333, 0}, {0.666667, 0}, {1., 0},
  {-1., 0.5}, {-0.666667, 0.5}, {-0.333333, 0.5}, {0, 0.5}, {0.333333, 0.5}, {0.666667, 0.5}, {1., 0.5},
  {-1., 1.}, {-0.666667, 1.}, {-0.333333, 1.}, {0, 1.}, {0.333333, 1.}, {0.666667, 1.}, {1., 1.}}
```

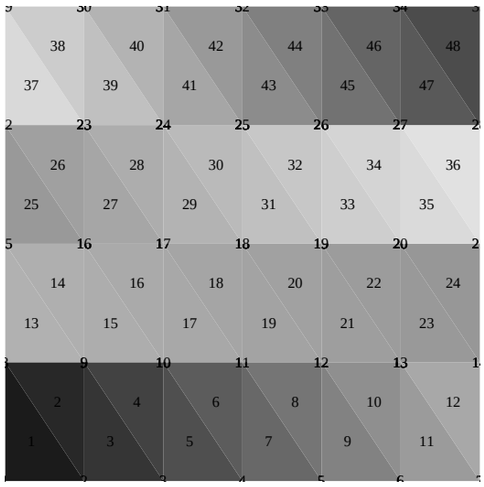
Calculate the indices of nodes of the triangle elements :

```
elements = Flatten[Table[{{i, i + 1, i + nx}, {i + 1, i + nx + 1, i + nx}} + (j - 1) nx, {j, 1, ny - 1}, {i, 1, nx - 1}], 2]
```

```
{{1, 2, 8}, {2, 9, 8}, {2, 3, 9}, {3, 10, 9}, {3, 4, 10}, {4, 11, 10}, {4, 5, 11}, {5, 12, 11}, {5, 6, 12}, {6, 13, 12}, {6, 7, 13},
  {7, 14, 13}, {8, 9, 15}, {9, 16, 15}, {9, 10, 16}, {10, 17, 16}, {10, 11, 17}, {11, 18, 17}, {11, 12, 18}, {12, 19, 18}, {12, 13, 19},
  {13, 20, 19}, {13, 14, 20}, {14, 21, 20}, {15, 16, 22}, {16, 23, 22}, {16, 17, 23}, {17, 24, 23}, {17, 18, 24}, {18, 25, 24},
  {18, 19, 25}, {19, 26, 25}, {19, 20, 26}, {20, 27, 26}, {20, 21, 27}, {21, 28, 27}, {22, 23, 29}, {23, 30, 29}, {23, 24, 30},
  {24, 31, 30}, {24, 25, 31}, {25, 32, 31}, {25, 26, 32}, {26, 33, 32}, {26, 27, 33}, {27, 34, 33}, {27, 28, 34}, {28, 35, 34}}
```

Here is a picture to visualize the triangle elements :

```
jmax = Length[elements]; $DefaultFont = {"Times-Roman", 9};
PlotColor[hue_] := Hue[2 (1 - Min[1, Max[0, hue]]) / 3];
Show[Graphics[Table[plist = Map[nodes[[#]] &, elements[[j]]]; {PlotColor[(j - 1) / (jmax - 1)], Polygon[plist],
        RGBColor[0, 0, 0], Text[j, Plus @@ plist / 3], Table[Text[elements[[j, i]], plist[[i]], {i, 1, 3}]}, {j, 1, jmax, 1}]],
  AspectRatio → Automatic, PlotRange → {{-1, 1} a / 2, {-1, 1} b / 2}];
```



Calculate the local stiffness matrices :

my calculations (does this agree with Roy?) : $\lambda_2 = \Delta x_2^2 + \Delta y_2^2$, $\lambda_{23} = \Delta x_2\,\Delta y_2 + \Delta x_3\,\Delta y_3$, $\lambda_3 = \Delta x_3^2 + \Delta y_3^2$, $\lambda_a = \Delta x_3^2 - \Delta y_2^2$

$$K_{local} = \frac{1}{\Delta x_2\,\Delta y_3 - \Delta x_3\,\Delta y_2} \begin{pmatrix} (\Delta x_2 - \Delta y_2)^2 + (\Delta x_3 - \Delta y_3)^2 & \Delta x_2\,\Delta y_2 - \Delta y_2^2 + \Delta x_3\,\Delta y_3 - \Delta y_3^2 & -\Delta x_2^2 - \Delta x_3^2 + \Delta x_2\,\Delta y_2 + \Delta x_3\,\Delta y_3 \\ \Delta x_2\,\Delta y_2 - \Delta y_2^2 + \Delta x_3\,\Delta y_3 - \Delta y_3^2 & \Delta y_2^2 + \Delta y_3^2 & -\Delta x_2\,\Delta y_2 - \Delta x_3\,\Delta y_3 \\ -\Delta x_2^2 - \Delta x_3^2 + \Delta x_2\,\Delta y_2 + \Delta x_3\,\Delta y_3 & -\Delta x_2\,\Delta y_2 - \Delta x_3\,\Delta y_3 & \Delta x_2^2 + \Delta x_3^2 \end{pmatrix}$$

$$= \frac{1}{\Delta x_2\,\Delta y_3 - \Delta x_3\,\Delta y_2} \begin{pmatrix} \lambda_2 + \lambda_3 - 2\,\lambda_{23} & \lambda_{23} - \lambda_3 + \lambda_a & \lambda_{23} - \lambda_2 - \lambda_a \\ \lambda_{23} - \lambda_3 + \lambda_a & \lambda_3 - \lambda_a & -\lambda_{23} \\ \lambda_{23} - \lambda_2 - \lambda_a & -\lambda_{23} & \lambda_2 + \lambda_a \end{pmatrix}$$

```
Klocal =
  Map[Module[{}, Do[{Δx_i, Δy_i} = nodes[[#[[i]]]] − nodes[[#[[1]]]], {i, 2, 3}]; λ2 = Δx2^2 + Δy2^2; λ23 = Δx2 Δx3 + Δy2 Δy3;
      λ3 = Δx3^2 + Δy3^2; d = Δx2 Δy3 − Δx3 Δy2; Chop[1/d ( λ2 + λ3 − 2 λ23   λ23 − λ3   λ23 − λ2
                                                            λ23 − λ3          λ3         −λ23
                                                            λ23 − λ2          −λ23       λ2       )]] &, elements];

Map[
  MatrixForm,
  Klocal]
```

$$\left\{\begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix},\right.$$

$$\begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix},$$

$$\begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix},$$

$$\begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix},$$

$$\begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix},$$

$$\begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix},$$

$$\begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix},$$

$$\begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix},$$

$$\left\{\begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix},\right.$$
$$\begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix},$$
$$\begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix},$$
$$\begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix},$$
$$\begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix},$$
$$\begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix},$$
$$\left.\begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}, \begin{pmatrix} 2.16667 & -1.5 & -0.666667 \\ -1.5 & 1.5 & 0 \\ -0.666667 & 0 & 0.666667 \end{pmatrix}, \begin{pmatrix} 0.666667 & -0.666667 & 0 \\ -0.666667 & 2.16667 & -1.5 \\ 0 & -1.5 & 1.5 \end{pmatrix}\right\}$$

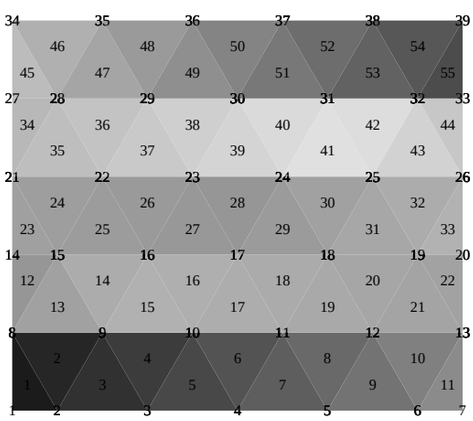
Assemble global stiffness matrix :

```
nxy = Length[nodes]; Kglobal = Table[0, {nxy}, {nxy}];
Do[element = elements⟦j⟧;
  Do[Kglobal⟦element⟦m⟧, element⟦n⟧⟧ += Klocal⟦j, m, n⟧, {m, 1, 3}, {n, 1, 3}], {j, 1, jmax}];
Kglobal // MatrixForm
```

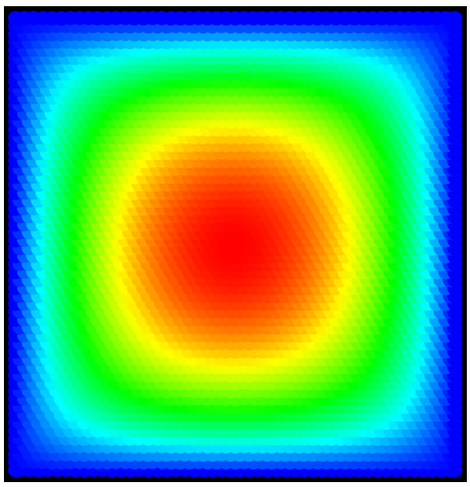
2.16667	−1.5	0	0	0	0	0	−0.666667	0	0	0
−1.5	4.33333	−1.5	0	0	0	0	0	−1.33333	0	0
0	−1.5	4.33333	−1.5	0	0	0	0	0	−1.33333	0
0	0	−1.5	4.33333	−1.5	0	0	0	0	0	−1.33
0	0	0	−1.5	4.33333	−1.5	0	0	0	0	0
0	0	0	0	−1.5	4.33333	−1.5	0	0	0	0
0	0	0	0	0	−1.5	2.16667	0	0	0	0
−0.666667	0	0	0	0	0	0	4.33333	−3.	0	0
0	−1.33333	0	0	0	0	0	−3.	8.66667	−3.	0
0	0	−1.33333	0	0	0	0	0	−3.	8.66667	−3
0	0	0	−1.33333	0	0	0	0	0	−3.	8.66
0	0	0	0	−1.33333	0	0	0	0	0	−3
0	0	0	0	0	−1.33333	0	0	0	0	0
0	0	0	0	0	0	−0.666667	0	0	0	0
0	0	0	0	0	0	0	−0.666667	0	0	0
0	0	0	0	0	0	0	0	−1.33333	0	0
0	0	0	0	0	0	0	0	0	−1.33333	0
0	0	0	0	0	0	0	0	0	0	−1.33
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

■ Honey Comb

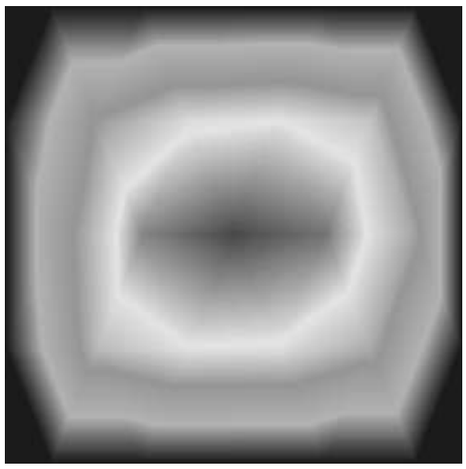
```
nx = 50; ny = Round[2 nx Tan[π / 6]]; dx = 1.0 / (nx − 2);
nodes = Flatten[Table[Table[{Min[1, Max[−1, x]], dx (j − 0.5 (ny + 1)) / Tan[π / 6]},
          {x, −(1 + dx Mod[j, 2]), 1 + dx, 2 dx}], {j, 1, ny}], 1];
elements = Flatten[Table[Table[i + (j − 1) nx − Floor[j / 2] + Join[If[Mod[j, 2] == 1 || i > 1, {{0, 1, nx}}, {}],
              If[Mod[j, 2] == 0 || i < nx − 1, {{1, nx + 1, nx}}, {}]], {i, 1, nx − 1}], {j, 1, ny − 1}], 2];
jmax = Length[elements]; PlotColor[hue_] := Hue[2 (1 − Min[1, Max[0, hue]]) / 3];

$DefaultFont = {"Times−Roman", 9};
Show[Graphics[Table[plist = Map[nodes⟦#⟧ &, elements⟦j⟧];
        {PlotColor[(j − 1) / (jmax − 1)], Polygon[plist], RGBColor[0, 0, 0], Text[j, Plus @@ plist / 3],
          Table[Text[elements⟦j, i⟧, plist⟦i⟧], {i, 1, 3}]}, {j, 1, jmax, 1}]], AspectRatio → Automatic, PlotRange → All];
```



```
Klocal =
  Map[Module[{}, Do[{Δx_i, Δy_i} = nodes[[#[[i]]]] - nodes[[#[[1]]]], {i, 2, 3}]; λ2 = Δx2^2 + Δy2^2; λ23 = Δx2 Δx3 + Δy2 Δy3;
      λ3 = Δx3^2 + Δy3^2; d = Δx2 Δy3 - Δx3 Δy2; Chop[1/d {{λ2 + λ3 - 2 λ23, λ23 - λ3, λ23 - λ2},
      {λ23 - λ3, λ3, -λ23}, {λ23 - λ2, -λ23, λ2}}]] &, elements];
nxy = Length[nodes]; Kglobal = Table[0, {nxy}, {nxy}];
Do[element = elements[[j]]; Do[Kglobal[[element[[m]], element[[n]]]] += Klocal[[j, m, n]], {m, 1, 3}, {n, 1, 3}], {j, 1, jmax}];
Q = Table[0, {nxy}]; Map[Module[{}, Do[{Δx_i, Δy_i} = nodes[[#[[i]]]] - nodes[[#[[1]]]], {i, 2, 3}];
      Do[Q[[#[[n]]]] += (Δx2 Δy3 - Δx3 Δy2)/6, {n, 1, 3}]] &, elements];
ymax = Max[Map[#[[2]] &, nodes]];
Do[{x, y} = nodes[[i]];
    If[Abs[x] == 1 || Abs[y] == ymax, Kglobal[[i]] = Table[If[j == i, 1, 0], {j, 1, nxy}]; Q[[i]] = 0], {i, 1, nxy}];
q = LinearSolve[Kglobal, Q]; q1 = Min[q]; q2 = Max[q];
Show[Graphics[{PointSize[1.5 dx], Table[{PlotColor[(q[[i]] - q1)/(q2 - q1)], Point[nodes[[i]]]}, {i, 1, nxy}]},
      AspectRatio → Automatic, Background → RGBColor[0, 0, 0]]];
```



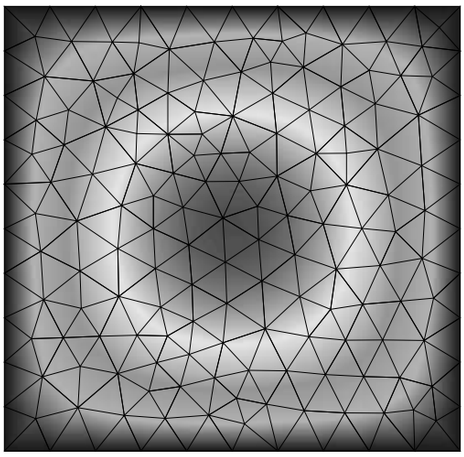
```
(* Interpolated plot, runtime: 2.3 minutes *)
Interpolate[x_, y_] :=
    Module[{j = 1, ξ = 2, η = 0}, While[! (0 ≤ ξ ≤ 1 && 0 ≤ η ≤ 1 && ξ + η ≤ 1), element = elements[[j]];
        {x1, y1} = nodes[[element[[1]]]]; Do[{Δx_i, Δy_i} = nodes[[element[[i]]]] - {x1, y1}, {i, 2, 3}]; d = Δx_2 Δy_3 - Δx_3 Δy_2;
        {ξ, η} = ((x - x1) {Δy_3, -Δy_2} + (y - y1) {-Δx_3, Δx_2}) / d; j++]; q[[element]].{1 - ξ - η, ξ, η}];
ymax = 0.5 dx (ny - 1) / Tan[π / 6];
DensityPlot[Interpolate[x, y], {x, -1, 1}, {y, -ymax, 0.999 ymax},
    PlotPoints → 275, Mesh → False, Frame → False, ColorFunction → PlotColor];
```



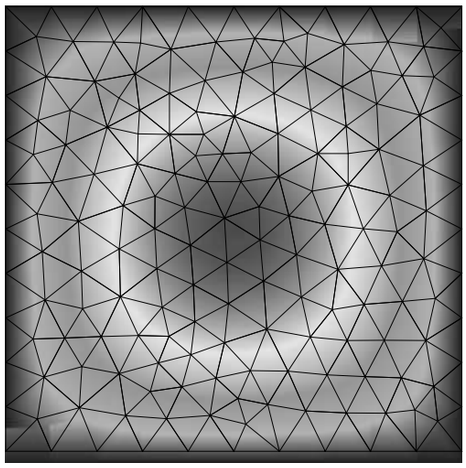
■ Unstructured Grid

```
nodes =
  Transpose[Import["C:/School/Engineering/ME207 - Computer Modeling/Unstructured Grid/mesh_p.txt", "Table"]];
elements = Transpose[Round[Delete[
        Import["C:/School/Engineering/ME207 - Computer Modeling/Unstructured Grid/mesh_t.txt", "Table"], 4]]];
jmax = Length[elements]; xlist = Map[#[[1]] &, nodes]; ylist = Map[#[[2]] &, nodes];
{x1, x2} = {Min[xlist], Max[xlist]}; {y1, y2} = {Min[ylist], Max[ylist]};
nxy = Length[nodes]; Q = Table[0, {nxy}];
Klocal = Map[Module[{}, Do[{Δx_i, Δy_i} = nodes[[#[[i]]]] - nodes[[#[[1]]]], {i, 2, 3}]; λ_2 = Δx_2^2 + Δy_2^2;
        λ_23 = Δx_2 Δx_3 + Δy_2 Δy_3; λ_3 = Δx_3^2 + Δy_3^2; d = Δx_2 Δy_3 - Δx_3 Δy_2; Do[Q[[#[[n]]]] += d / 6, {n, 1, 3}];
        Chop[1/d {{λ_2 + λ_3 - 2 λ_23, λ_23 - λ_3, λ_23 - λ_2}, {λ_23 - λ_3, λ_3, -λ_23}, {λ_23 - λ_2, -λ_23, λ_2}}]] &, elements]; nxy = Length[nodes];
Kglobal = Table[0, {nxy}, {nxy}];
Do[element = elements[[j]];
    Do[Kglobal[[element[[m]], element[[n]]]] += Klocal[[j, m, n]], {m, 1, 3}, {n, 1, 3}], {j, 1, jmax}];
Do[{x, y} = nodes[[i]]; If[x == x1 || x == x2 || y == y1 || y == y2, Kglobal[[i]] = Table[If[j == i, 1, 0], {j, 1, nxy}]; Q[[i]] = 0],
  {i, 1, nxy}]; q = Chop[LinearSolve[Kglobal, Q]];
```

```
(* 17 minutes *)
Interpolate[x_, y_] :=
    Module[{j = 1, ξ = 2, η = 0, x1, y1}, While[! (0 ≤ ξ ≤ 1 && 0 ≤ η ≤ 1 && ξ + η ≤ 1), element = elements[[j]];
        {x1, y1} = nodes[[element[[1]]]]; Do[{Δx_i, Δy_i} = nodes[[element[[i]]]] - {x1, y1}, {i, 2, 3}]; d = Δx_2 Δy_3 - Δx_3 Δy_2;
        {ξ, η} = ((x - x1) {Δy_3, -Δy_2} + (y - y1) {-Δx_3, Δx_2}) / d; j++]; q[[element]].{1 - ξ - η, ξ, η}];
DensityPlot[Interpolate[x, y], {x, x1, x2}, {y, y1, y2}, PlotPoints → 275, AspectRatio → Automatic,
    Mesh → False, Frame → False, ColorFunction → (Hue[2 (1 - #) / 3] &),
    Epilog → Table[{Line[Map[nodes[[#]] &, elements[[j]]]]}, {j, 1, jmax}]];
```



```
n = 275; image = Table[0, {n}, {n}];
Interpolate[x_, y_] := Module[{x1, y1}, {x1, y1} = plist[[1]]; Do[{Δx_i, Δy_i} = plist[[i]] - {x1, y1}, {i, 2, 3}];
      {ξ, η} = ((x - x1) {Δy_3, -Δy_2} + (y - y1) {-Δx_3, Δx_2}) / (Δx_2 Δy_3 - Δx_3 Δy_2); q[[element]].{1 - ξ - η, ξ, η}];
Do[element = elements[[j]]; plist = nodes[[element]]; xlist = Map[#[[1]] &, plist]; ylist = Map[#[[2]] &, plist];
    xIntersect[{{x1_, y1_}, {x2_, y2_}}] := x1 + If[y1 == y2, 0, (y - y1) (x2 - x1) / (y2 - y1)];
    Do[y = y1 + (y2 - y1) (i - 1) / (n - 1); jlist =
        Floor[n (Select[Map[xIntersect, Partition[plist, 2, 1, 1]], (Min[xlist] ≤ # ≤ Max[xlist]) &] - x1) / (x2 - x1)] + 1;
      If[jlist =!= {}, Do[x = x1 + (x2 - x1) (jj - 1) / (n - 1);
          image[[Max[1, Min[n, i]], Max[1, Min[n, jj]]]] = Interpolate[x, y], {jj, Min[jlist], Max[jlist]}]],
      {i, Floor[n (Min[ylist] - y1) / (y2 - y1)] + 1, Floor[n (Max[ylist] - y1) / (y2 - y1)] + 1}], {j, 1, jmax}];
ListDensityPlot[image, AspectRatio → Automatic, Mesh → False, Frame → False, ColorFunction → (Hue[2 (1 - #) / 3] &),
    Epilog → Table[{Line[Map[n (nodes[[#]] + 1) / 2 &, elements[[j]]]]}, {j, 1, jmax}]];
```



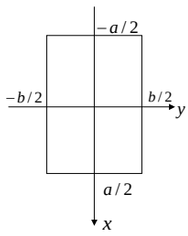
Printed by Mathematica for Students

clearpage

2.4 Documents by Wei Gao (from MAE 207 Fall 2006 class)

Notes on Torsion problem for cross section. Send to class mailing list on Nov 27,2006. Torsion and FEM.

Torsion of Rectangular Bars



$$\nabla^2\phi = -2G\alpha$$

$$\phi\big|_\Gamma = 0$$

Series Solution:

$$\phi = G\alpha\left(\frac{b^2}{4} - y^2\right) - \frac{8G\alpha b^2}{\pi}\sum_{n=0}^{\infty}\frac{(-1)^n}{(2n+1)^3}\frac{\cosh\lambda_n x}{\cosh(\lambda_n a/2)}\cos\lambda_n y$$

where $\lambda_n = (2n+1)\frac{\pi}{b}$

Finite Element Program

MATLAB program which solves the 2D Poisson equation with Dirichlet boundary conditions:

$$\nabla^2\phi + f(x,y) = 0$$

$$\phi\big|_\Gamma = 0$$

http://www.csit.fsu.edu/~burkardt/m_src/fem2d_poisson/fem2d_poisson.html

Notice: There is a minor error in “nodes_plot.m”

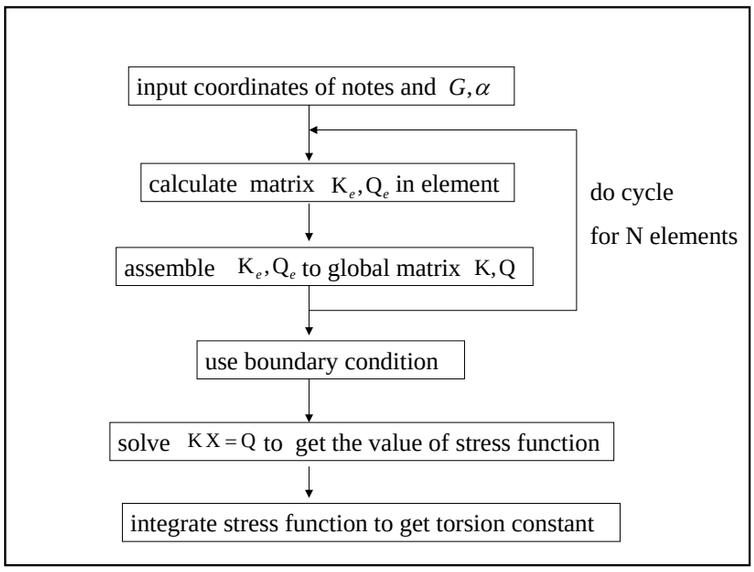
1

How to use this program?

"rhs.m": modify the function $f(x, y)$

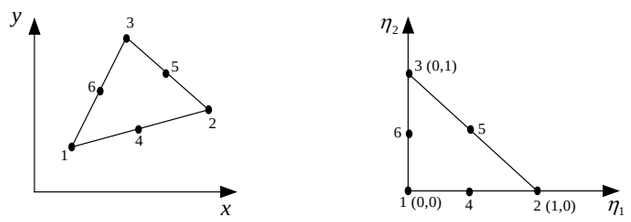
"exact.m": input exact solution if it exists

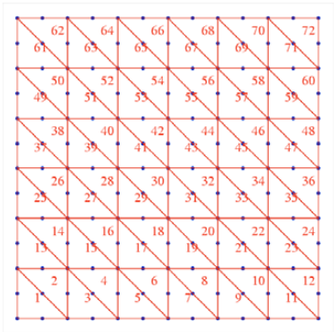
Run "fem2d_poisson.m" , get the results!



2

Quadrate Triangle Element



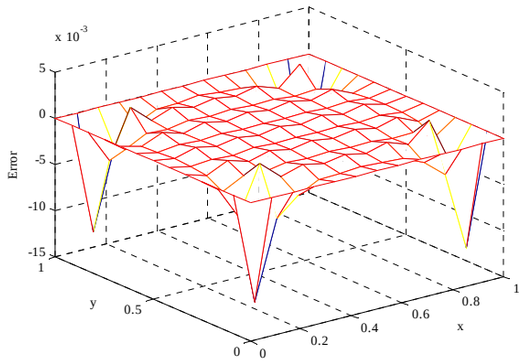


$a = b = 1mm$

$\alpha = 0.001$

$G = 10^8 MPa$

Error of the value of stress function



$$error = \frac{\phi_{series} - \phi_{fem}}{\phi_{series}}$$

Larger error in corner nodes?

4

Torsion Constant

$$D_t = \frac{M_t}{\alpha} = \frac{2\iint_S \phi dxdy}{\alpha}$$

$D_t^{series} = 1.4058e+007$

$D_t^{fem} = 1.3828e+007$

$error = 1.64\%$

Notice: we do the integration in rectangular element, but it is better to do it in triangular element.

5