INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING, VOL. 37, 1027–1038 (1994)

STIFFNESS MATRIX OF THE FOUR-NODE QUADRILATERAL ELEMENT IN CLOSED FORM

D. V. GRIFFITHS

Department of Engineering, Colorado School of Mines, Golden, Colorado 80401, U.S.A.

SUMMARY

The stiffness matrix of a plane four-node quadrilateral finite element is given in closed form. When expressed as a FORTRAN subroutine and compared with the classical method of forming the stiffness matrix using Gaussian integration, the approach gives a CPU time speed-up of the order of 2–3 on a vector machine and of the order of 4–5 on a scalar machine. The technique used to generate the terms of the stiffness matrix made use of a computer algebra system which could clearly be extended to generate the matrices for other elements types.

1. INTRODUCTION

Consider the plane four-node quadrilateral element shown in Figure 1. The stiffness of the element is fully defined by the nodal co-ordinates $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)$ and the elastic properties E and ν. The 8×8 stiffness matrix of the element is customarily obtained using Gauss–Legendre quadrature with two integrating points in each of the two local co-ordinate directions (see e.g. Reference 1). It is well known that this represents the minimum number required to integrate the stiffness terms of this particular element exactly.

Analytical expressions for the fully integrated stiffness matrix of a *rectangular* four-node element have been published by Hacker and Schreyer[2] and analytical integration formulae for linear isoparametric elements by Babu and Pinder[3] and Rathod.[4] This paper describes how the stiffness matrix of a general quadrilateral element can be expressed in closed form by expanding and simplifying the four terms in the numerical integration summation. The computer algebra system Maple[5] was used to help generate the expressions.

Computer Algebra Systems (CAS) have considerable potential in the area of finite element software generation. In particular, Bettess and Bettess[6] and Barbier *et al.*[7] showed how the computer algebra system REDUCE[8] could be used to generate shape functions automatically for any finite element. The systems usually have the additional facility of being able to generate output in the FORTRAN programming language; thus, complex algebraic expressions can be coded without the usual risk of typographical errors.

The ability of CAS to simplify and factorize complex algebraic terms has limitations, however, so some of the expressions produced by the CAS had to be further simplified by hand in order to arrive at a form suitable for publication.

2. FORMULATION

For the present development, we assume that the stress conditions are those of two-dimensional plane elasticity; hence, the stress/strain relationship is given by

$$\boldsymbol{\sigma} = \mathbf{D}\boldsymbol{\varepsilon} \tag{1}$$

CCC 0029–5981/94/061027–12$9.00

Received 18 May 1992
Revised 17 September 1993

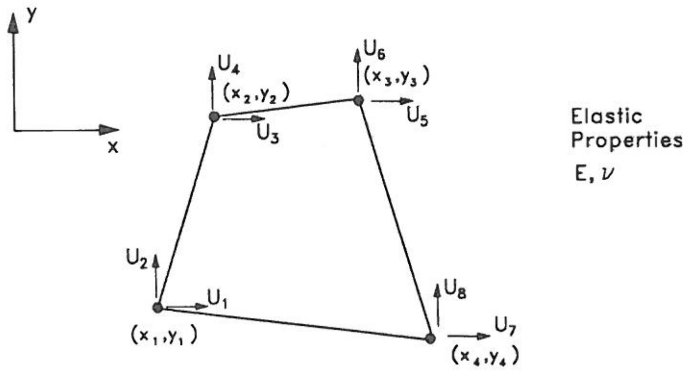

Figure 1. General four-node quadrilateral element

where

$$\boldsymbol{\sigma} = [\sigma_x \;\; \sigma_y \;\; \tau_{xy}]^{\mathrm{T}} \tag{2}$$

$$\boldsymbol{\varepsilon} = [\varepsilon_x \;\; \varepsilon_y \;\; \gamma_{xy}]^{\mathrm{T}} \tag{3}$$

For an isotropic material, the stress/strain $\mathbf{D}$ matrix is

$$\mathbf{D} = \begin{bmatrix} E_1 & E_2 & 0 \\ E_2 & E_1 & 0 \\ 0 & 0 & G \end{bmatrix} \tag{4}$$

where the shear modulus is given by $G = E/(2(1+\nu))$, with E and ν denoting Young's modulus and Poisson's ratio, respectively.

For plane stress,

$$E_1 = \frac{E}{1-\nu^2}, \quad E_2 = \nu E_1 \tag{5}$$

whereas for plane strain,

$$E_1 = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}, \quad E_2 = \frac{\nu E_1}{(1-\nu)} \tag{6}$$

The element stiffness relationship is given by

$$\mathbf{ku} = \mathbf{f} \tag{7}$$

where the stiffness matrix $\mathbf{k}$ can be written as (see e.g. Reference 9)

$$\mathbf{k} = \int_{V^e} \mathbf{B}^{\mathrm{T}} \mathbf{D} \mathbf{B} \, d(\mathrm{vol}) \tag{8}$$

and the strain/displacement matrix $\mathbf{B}$ is given by

$$\mathbf{B} = \begin{bmatrix} \frac{\partial N_1}{\partial x} & 0 & \frac{\partial N_2}{\partial x} & 0 & \frac{\partial N_3}{\partial x} & 0 & \frac{\partial N_4}{\partial x} & 0 \\ 0 & \frac{\partial N_1}{\partial y} & 0 & \frac{\partial N_2}{\partial y} & 0 & \frac{\partial N_3}{\partial y} & 0 & \frac{\partial N_4}{\partial y} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial y} & \frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial y} & \frac{\partial N_3}{\partial x} & \frac{\partial N_4}{\partial y} & \frac{\partial N_4}{\partial x} \end{bmatrix} \tag{9}$$

It may be noted that this expression for $\mathbf{B}$ assumes that the x- and y-nodal displacements are numbered alternately, thus, with reference to Figure 1, the nodal displacement and force vectors are given by

$$\begin{aligned} \mathbf{u} &= [u_1\ u_2\ u_3\ u_4\ u_5\ u_6\ u_7\ u_8]^{\mathrm{T}} \\ \mathbf{f} &= [f_1\ f_2\ f_3\ f_4\ f_5\ f_6\ f_7\ f_8]^{\mathrm{T}} \end{aligned} \tag{10}$$

The N's are the shape functions in local co-ordinates, which for this four-node quadrilateral element are given as

$$\begin{aligned} N_1 &= \tfrac{1}{4}(1-\xi)(1-\eta) \\ N_2 &= \tfrac{1}{4}(1-\xi)(1+\eta) \\ N_3 &= \tfrac{1}{4}(1+\xi)(1+\eta) \\ N_4 &= \tfrac{1}{4}(1+\xi)(1-\eta) \end{aligned} \tag{11}$$

As the element is isoparametric, the relationship between local and global co-ordinate systems is given by

$$\begin{aligned} x &= N_1x_1 + N_2x_2 + N_3x_3 + N_4x_4 \\ y &= N_1y_1 + N_2y_2 + N_3y_3 + N_4y_4 \end{aligned} \tag{12}$$

The numerically integrated element stiffness matrix can be expressed as

$$\mathbf{k} = \sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}(\det \mathbf{J})_{ij}\, \mathbf{B}^{\mathrm{T}}\mathbf{D}\mathbf{B}_{ij} \tag{13}$$

where the subscripts i and j index the integrating points.

For the two-point formula considered here, $N = 2$, the weighting coefficients w_{ij} all equal unity, and the integrating points are located at $\pm 1/\sqrt{3}$ in local co-ordinates (ξ, η).

The scalar $(\det \mathbf{J})$ is the determinant of the Jacobian matrix, where

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\[2ex] \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} \tag{14}$$

and this, together with the matrix $\mathbf{B}^{\mathrm{T}}\mathbf{D}\mathbf{B}$ is evaluated at each 'Gauss' point in turn.

2.1. Properties of the stiffness matrix

Before evaluating the terms of the stiffness matrix, some observations can be made about its structure, as this will lead to a reduction in computational effort. The matrix will be symmetrical, so only those terms on and below the main diagonal will need to be evaluated. These terms are given in equation (15):

$$\mathbf{k} = \begin{bmatrix} k_{11} & & & & & & & \\ k_{21} & k_{22} & & & & & & \\ k_{31} & k_{32} & k_{33} & & & & & \\ k_{41} & k_{42} & k_{43} & k_{44} & & & & \\ k_{51} & k_{52} & k_{53} & k_{54} & k_{55} & & & \\ k_{61} & k_{62} & k_{63} & k_{64} & k_{65} & k_{66} & & \\ k_{71} & k_{72} & k_{73} & k_{74} & k_{75} & k_{76} & k_{77} & \\ k_{81} & k_{82} & k_{83} & k_{84} & k_{85} & k_{86} & k_{87} & k_{88} \end{bmatrix} \tag{15}$$

The terms can also split in six different groups as given in Table I. Within each group, the bold term can be considered to be the 'parent' member from which all the others can be obtained by a simple rearrangement of the nodal co-ordinates.

The distribution of the different groups in a matrix layout is given as

$$\mathbf{k} = \begin{bmatrix} \boldsymbol{A} & & & & & & & \\ \boldsymbol{B} & A & & & & & & \\ \boldsymbol{C} & D & A & & & & & \\ \boldsymbol{D} & C & B & A & & & & \\ \boldsymbol{E} & F & C & D & A & & & \\ \boldsymbol{F} & E & D & C & B & A & & \\ C & D & E & F & C & D & A & \\ D & C & F & E & D & C & B & A \end{bmatrix} \tag{16}$$

2.2. *Parent terms of the stiffness matrix*

The parent member of each group from Table I is given in this section, followed by a description of the nodal co-ordinate transformations necessary to retrieve all the other members of that group.

All stiffness terms are of the form

$$k_{ij} = \frac{1}{2}\left\{\frac{A_2(E^*s_1 + Gs_2) + f_1(E^*s_3 + Gs_4)}{3A_2^2 - f_1^2} + \frac{A_2(E^*t_1 + Gt_2) + f_2(E^*t_3 + Gt_4)}{3A_2^2 - f_2^2}\right\} \tag{17}$$

where E^* equals either E_1 or E_2 as indicated in the following sections:

$$\begin{aligned} A_2 &= (x_4 - x_2)(y_3 - y_1) - (x_3 - x_1)(y_4 - y_2) \\ &= \text{twice the area of the element} \end{aligned} \tag{18}$$

Table I. Types of stiffness terms in the four-node element

Group	Description	Terms
A	Diagonals	$\boldsymbol{k_{11}}k_{33}k_{55}k_{77}$ $k_{22}k_{44}k_{66}k_{88}$
B	Orthogonal freedoms at the same node	$\boldsymbol{k_{21}}k_{43}k_{65}k_{87}$
C	Parallel freedoms at adjacent nodes	$\boldsymbol{k_{31}}k_{53}k_{75}k_{71}$ $k_{86}k_{64}k_{42}k_{82}$
D	Orthogonal freedoms at adjacent nodes	$\boldsymbol{k_{41}}k_{63}k_{85}k_{72}$ $k_{32}k_{54}k_{76}k_{81}$
E	Parallel freedoms at opposite nodes	$\boldsymbol{k_{51}}k_{73}k_{84}k_{62}$
F	Orthogonal freedoms at opposite nodes	$\boldsymbol{k_{61}}k_{83}k_{52}k_{74}$

and

$$f_1 = (x_1 + x_3)(y_4 - y_2) - (y_1 + y_3)(x_4 - x_2) - 2(x_2 y_4 - x_4 y_2) \tag{19}$$

$$f_2 = (y_2 + y_4)(x_3 - x_1) - (x_2 + x_4)(y_3 - y_1) - 2(x_3 y_1 - x_1 y_3) \tag{20}$$

The functions $s_1, s_2, s_3, s_4, t_1, t_2, t_3$ and t_4 depend on the nodal co-ordinates, and are given for each of the parent terms in the next six sections.

Having obtained the parent member of each group, the other members of each group from Table I can be obtained using one of the three nodal co-ordinate transformations listed below.

The notation used here is that the symbol $\Leftarrow$ means 'is overwritten by'.

Type 1	
Transformation	Terms affected
$(x_1, y_1) \Leftarrow (x_2, y_2)$	f_1, f_2
$(x_2, y_2) \Leftarrow (x_3, y_3)$	s_1, s_2, s_3, s_4
$(x_3, y_3) \Leftarrow (x_4, y_4)$	t_1, t_2, t_3, t_4
$(x_4, y_4) \Leftarrow (x_1, y_1)$	

Type 2	
Transformation	Terms affected
$(x_1, y_1) \Leftarrow (y_3, x_3)$	f_1, f_2
$(x_2, y_2) \Leftarrow (y_2, x_2)$	s_1, s_2, s_3, s_4
$(x_3, y_3) \Leftarrow (y_1, x_1)$	t_1, t_2, t_3, t_4
$(x_4, y_4) \Leftarrow (y_4, x_4)$	

Type 3	
Transformation	Terms affected
$(x_1, y_1) \Leftarrow (y_1, x_1)$	
$(x_2, y_2) \Leftarrow (y_2, x_2)$	s_1, s_2, s_3, s_4
$(x_3, y_3) \Leftarrow (y_3, x_3)$	t_1, t_2, t_3, t_4
$(x_4, y_4) \Leftarrow (y_4, x_4)$	

It should be noted that the transformations always apply to the 's' and 't' functions, however, transformation types 1 and 2 also apply to the 'f' functions. None of the transformations should be applied to A_2, which relates to the constant area of the element.

2.2.1. Group A—k_{11}

$$E^* = E_1 \tag{21}$$

$$s_1 = 2(y_4 - y_2)^2 \tag{22}$$

$$s_2 = 2(x_4 - x_2)^2 \tag{23}$$

$$s_3 = -s_1/2 \tag{24}$$

$$s_4 = -s_2/2 \tag{25}$$

$$t_1 = (y_2 - y_3)^2 + (y_3 - y_4)^2 + (y_4 - y_2)^2 \tag{26}$$

$$t_2 = (x_2 - x_3)^2 + (x_3 - x_4)^2 + (x_4 - x_2)^2 \tag{27}$$

$$t_3 = (y_4 - y_3)^2 - (y_3 - y_2)^2 \tag{28}$$

$$t_4 = (x_4 - x_3)^2 - (x_3 - x_2)^2 \tag{29}$$

To compute	From	Use transformation type
k_{33}	k_{11}	1
k_{55}	k_{33}	1
k_{77}	k_{55}	1
k_{88}	k_{77}	2
k_{22}	k_{88}	1
k_{44}	k_{22}	1
k_{66}	k_{44}	1

2.2.2. Group B—k_{21}

$$E^* = E_2 \tag{30}$$

$$s_1 = 2(x_2 - x_4)(y_4 - y_2) \tag{31}$$

$$s_2 = s_1 \tag{32}$$

$$s_3 = -s_1/2 \tag{33}$$

$$s_4 = s_3 \tag{34}$$

$$t_1 = x_2(y_4 - 2y_2 + y_3) + x_3(y_2 - 2y_3 + y_4) + x_4(y_2 - 2y_4 + y_3) \tag{35}$$

$$t_2 = t_1 \tag{36}$$

$$t_3 = x_2(y_2 - y_3) + x_3(y_4 - y_2) + x_4(y_3 - y_4) \tag{37}$$

$$t_4 = t_3 \tag{38}$$

To compute	From	Use transformation type
k_{43}	k_{21}	1
k_{65}	k_{43}	1
k_{87}	k_{65}	1

2.2.3. Group C—k_{31}

$$E^* = E_1 \tag{39}$$

$$s_1 = (y_4 - y_2)(2y_1 - y_3 - y_4) \tag{40}$$

$$s_2 = (x_4 - x_2)(2x_1 - x_3 - x_4) \tag{41}$$

$$s_3 = (y_4 - y_2)(y_4 - y_1) \tag{42}$$

$$s_4 = (x_4 - x_2)(x_4 - x_1) \tag{43}$$

$$t_1 = (y_3 - y_1)(2y_2 - y_3 - y_4) \tag{44}$$

$$t_2 = (x_3 - x_1)(2x_2 - x_3 - x_4) \tag{45}$$

$$t_3 = (y_3 - y_1)(y_3 - y_2) \tag{46}$$

$$t_4 = (x_3 - x_1)(x_3 - x_2) \tag{47}$$

To compute	From	Use transformation type
k_{53}	k_{31}	1
k_{75}	k_{53}	1
k_{71}	k_{75}	1
k_{86}	k_{71}	2
k_{82}	k_{86}	1
k_{42}	k_{82}	1
k_{64}	k_{42}	1

2.2.4. Group D—k_{41}

$$E^* = E_2 \tag{48}$$

$$s_1 = (x_3 - x_1)(y_4 - y_2) + (x_4 - x_1)(y_4 - y_2) \tag{49}$$

$$s_2 = (y_3 - y_1)(x_4 - x_2) + (y_4 - y_1)(x_4 - x_2) \quad (50)$$

$$s_3 = (x_4 - x_1)(y_2 - y_4) \quad (51)$$

$$s_4 = (y_4 - y_1)(x_2 - x_4) \quad (52)$$

$$t_1 = (x_3 - x_1)(y_4 - y_2) + (x_3 - x_1)(y_3 - y_2) \quad (53)$$

$$t_2 = (y_3 - y_1)(x_4 - x_2) + (y_3 - y_1)(x_3 - x_2) \quad (54)$$

$$t_3 = (x_3 - x_1)(y_2 - y_3) \quad (55)$$

$$t_4 = (y_3 - y_1)(x_2 - x_3) \quad (56)$$

To compute	From	Use transformation type
k_{63}	k_{41}	1
k_{85}	k_{63}	1
k_{72}	k_{85}	1
k_{32}	k_{41}	3
k_{54}	k_{32}	1
k_{76}	k_{54}	1
k_{81}	k_{76}	1

2.2.5. Group E—k_{51}

$$E^* = E_1 \quad (57)$$

$$s_1 = -(y_4 - y_2)^2 \quad (58)$$

$$s_2 = -(x_4 - x_2)^2 \quad (59)$$

$$s_3 = 0 \quad (60)$$

$$s_4 = 0 \quad (61)$$

$$t_1 = (y_3 + y_1)(y_4 + y_2) - 2(y_4 - y_2)^2 - 2(y_1 y_3 + y_2 y_4) \quad (62)$$

$$t_2 = (x_3 + x_1)(x_4 + x_2) - 2(x_4 - x_2)^2 - 2(x_1 x_3 + x_2 x_4) \quad (63)$$

$$t_3 = (y_4 - y_2)(y_1 - y_2 + y_3 - y_4) \quad (64)$$

$$t_4 = (x_4 - x_2)(x_1 - x_2 + x_3 - x_4) \quad (65)$$

To compute	From	Use transformation type
k_{73}	k_{51}	1
k_{84}	k_{73}	2
k_{62}	k_{84}	1

2.2.6. Group F—k_{61}

$$E^* = E_2 \tag{66}$$

$$s_1 = (x_4 - x_2)(y_4 - y_2) \tag{67}$$

$$s_2 = s_1 \tag{68}$$

$$s_3 = 0 \tag{69}$$

$$s_4 = 0 \tag{70}$$

$$t_1 = (x_4 - x_2)(y_4 - y_2) + (x_2 - x_1)(y_2 - y_3) + (x_4 - x_1)(y_4 - y_3) \tag{71}$$

$$t_2 = (y_4 - y_2)(x_4 - x_2) + (y_2 - y_1)(x_2 - x_3) + (y_4 - y_1)(x_4 - x_3) \tag{72}$$

$$t_3 = (x_2 - x_1)(y_3 - y_2) + (x_4 - x_1)(y_4 - y_3) \tag{73}$$

$$t_4 = (y_2 - y_1)(x_3 - x_2) + (y_4 - y_1)(x_4 - x_3) \tag{74}$$

To compute	From	Use transformation type
k_{83}	k_{61}	1
k_{52}	k_{61}	3
k_{74}	k_{52}	1

2.3. Subroutine timings

The expressions described in the previous section were entered into a FORTRAN subroutine. This enabled an efficiency comparison to be made with more conventional methods of forming the stiffness matrix which use the Gaussian quadrature formulation given in equation (13). Comparisons were performed on two machines; the scalar Amdahl 5890-300 and the Amdahl Vector Processor VP1100, both installed at the University of Manchester Computing Centre (MCC).

The first test involved calculation of the element stiffness matrix of a general quadrilateral element. Tables II and III show the CPU time used by the scalar and vector machines, respectively. The *explicit* approach refers to the formulation given in this paper, and the *numerical* approach to the more conventional numerical integration technique. The speed-up ratio is also given in all cases. As a further check on the timings, the matrix was evaluated repeatedly up to 10 000 times to avoid any misleading result based on the inclusion of CPU time occupied by other parts of the code.

It is clear from Tables II and III that the *explicit* approach for forming the element stiffness matrix tends to give a speed-up factor of 4·9 and 3·0 on the scalar and vector machines, respectively.

The second test involved calculation of the global stiffness matrix of a mesh of quadrilateral element. Tables IV and V show the CPU time required by the scalar and vector machines, respectively. In each case, three levels of mesh refinement were considered over the rectangular block shown in Figure 2. As the mesh was refined, the speed-up factors tended to 4·0 and 2·2 for the scalar and vector machines, respectively. Although still significant, the speed-up factors in this

Table II. Element stiffness matrix formulation (scalar machine)

Number of evaluations	CPU time (s) Explicit	CPU time (s) Numerical	Ratio
1	$0{\cdot}5160 \times 10^{-3}$	$0{\cdot}1922 \times 10^{-2}$	3·7
10	$0{\cdot}4020 \times 10^{-2}$	$0{\cdot}1873 \times 10^{-1}$	4·6
100	$0{\cdot}3868 \times 10^{-1}$	$0{\cdot}1867 \times 10^{0}$	4·8
1000	$0{\cdot}3800 \times 10^{0}$	$0{\cdot}1852 \times 10^{1}$	4·9
10 000	$0{\cdot}3799 \times 10^{1}$	$0{\cdot}1854 \times 10^{2}$	4·9

Table III. Element stiffness matrix formulation (vector machine)

Number of evaluations	CPU time (s) Explicit	CPU time (s) Numerical	Ratio
1	$0{\cdot}2596 \times 10^{-3}$	$0{\cdot}5049 \times 10^{-3}$	1·9
10	$0{\cdot}1562 \times 10^{-2}$	$0{\cdot}4452 \times 10^{-2}$	2·9
100	$0{\cdot}1465 \times 10^{-1}$	$0{\cdot}4403 \times 10^{-1}$	3·0
1000	$0{\cdot}1451 \times 10^{0}$	$0{\cdot}4378 \times 10^{0}$	3·0
10 000	$0{\cdot}1455 \times 10^{1}$	$0{\cdot}4365 \times 10^{1}$	3·0

Table IV. Global stiffness matrix formulation (scalar machine)

Number of elements in x-direction	Number of elements in y-direction	CPU Time (s) Explicit	CPU Time (s) Numerical	Ratio
3	2	$0{\cdot}3012 \times 10^{-2}$	$0{\cdot}1186 \times 10^{-1}$	3·9
15	20	$0{\cdot}7476 \times 10^{-1}$	$0{\cdot}3006 \times 10^{0}$	4·0
30	20	$0{\cdot}2980 \times 10^{0}$	$0{\cdot}1203 \times 10^{1}$	4·0

Table V. Global stiffness matrix formulation (vector machine)

Number of elements in x-direction	Number of elements in y-direction	CPU Time (s) Explicit	CPU Time (s) Numerical	Ratio
3	2	$0{\cdot}1084 \times 10^{-1}$	$0{\cdot}1250 \times 10^{-1}$	1·2
15	20	$0{\cdot}4415 \times 10^{-1}$	$0{\cdot}8959 \times 10^{-1}$	2·0
30	20	$0{\cdot}1502 \times 10^{0}$	$0{\cdot}3293 \times 10^{0}$	2·2

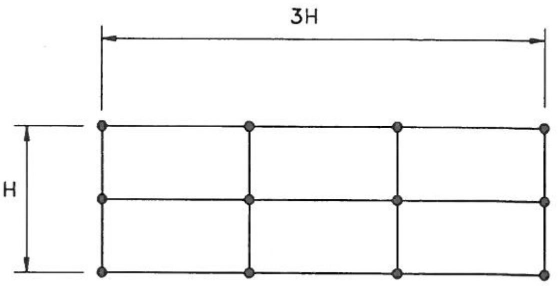

Figure 2. Rectangular mesh of elements

case were less than those required to produce the element matrices alone due to the overheads associated with the assembly process.

The main reason for the improved performance is that the code based on the *explicit* approach is comprised entirely of assignment statements which process very quickly whereas the more conventional *numerical* approach uses a number of loops which carry more overhead. The vector machine can take some advantage of the vector and matrix calculations in the *numerical* formulation; however, the improvement observed in the *explicit* approach is still significant.

The *numerical* formulation is undoubtedly more elegant, and considerably shorter in terms of the number of lines of code. In the context of a precompiled library of subroutines called by a main program, however, the elegance of the code is arguably of secondary importance to the processing speed.

2.4 Concluding remarks

The stiffness matrix of a plane four-node quadrilateral finite element has been expressed in closed form. The algebraic expressions were obtained with the help of a computer algebra system, and based on the summation of the numerical integration terms that would be required to integrate the element stiffness exactly.

A comparison of the processing speed of the *explicit* and *numerical* approaches indicated a speed-up of between two and four on the vector and scalar machines, respectively, in an example of global stiffness matrix assembly.

It should be noted that no attempt was made in the present work to optimize either the *explicit* or the *numerical* approaches under comparison. Undoubtedly, further improvements could be made to both algorithms; however, it is believed that the *explicit* approach will always occupy less CPU time due to the fact that it is comprised entirely of assignment statements.

Although the exactly integrated four-node element is not widely used, the techniques used to generate the stiffness terms could easily be extrapolated to other types of element matrix (e.g. mass), and higher-order elements. Work is presently under way to obtain expressions for matrices of more 'popular' elements, such as the four-node element with selective reduced integration and the eight-node element with uniform reduced integration.

Both the FORTRAN subroutine which produces the four-node element stiffness matrix and the Maple source code used to generate the expressions are available from the author on request.

ACKNOWLEDGEMENTS

The work described in this paper was supported in part by an SERC grant (No. GR/G55921) on 'Development of efficient algorithms for 3-D analysis of non-linear soils'.

REFERENCES

1. I. M. Smith and D. V. Griffiths, *Programming the Finite Element Method*, 2nd edn, Wiley, Chichester, New York, 1988.
2. W. L. Hacker and H. L. Schreyer, 'Eigenvalue analysis of compatible and incompatible rectangular four-node quadrilateral elements', *Int. j. numer. methods eng.*, **28**, 687–703 (1989).
3. D. Babu and G. F. Pinder, 'Analytical integration formulae for linear isoparametric finite elements', *Int. j. Numer. methods eng.*, **20**, 1153–1166 (1984).
4. H. D. Rathod. 'Some analytical integration formulae for a four node isoparametric element', *Comput. Struct.*, **30**, 1101–1109 (1988).
5. D. Harper, *An Introduction to Maple*, Computer Algebra Support Project, University of Liverpool, 1989; Maple is available from WATCOM Publications Ltd., Waterloo, Ontario, Canada N2L 3X2.
6. P. Bettess and J. A. Bettess, 'Automatic generation of shape function routines', in G. N. Pande and J. Middleton (eds.), *Proc. Int. Conf. Num. Methods in Eng.: Theory and Applications*, Vol. 1, Martinus Nuhoff, 1987.
7. C. Barbier, P. J. Clark, P. Bettess and J. A. Bettess, 'Automatic generation of shape functions for finite element analysis using REDUCE', *Eng. Comput.* (Swansea, Wales), **7**(4), 349–358 (1990).
8. G. Rayna, *REDUCE—Software for Algebraic Computation*, Springer, Berlin, 1987.
9. O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method*, Vol. 1, 4th edn, Mc-Graw Hill, London, New York, 1989.