

Elastic stiffness of straight-sided triangular finite elements by analytical and numerical integration

D. V. Griffiths^{*,†}, Jinsong Huang and R. P. Schiermeyer

Colorado School of Mines, Golden, CO, U.S.A.

SUMMARY

Most finite element (FE) software uses numerical integration to compute FE stiffness matrices. In the case of straight-sided isoparametric triangular elements, numerical integration is exact, provided a sufficient number of integration points are used. In this paper, the same integrations are performed analytically in closed form with the help of computer algebra software for 3-, 6-, 10- and 15-noded triangles. The resulting analytical expressions have been converted into Fortran 95 subroutines and compared with the conventional numerical integration methods. The analytical routines produce exactly the same results as their numerical counterparts, but run significantly faster. All Fortran 95 code developed in this study is available on the first author's Web site. Copyright © 2008 John Wiley & Sons, Ltd.

Received 23 October 2007; Revised 18 February 2008; Accepted 19 February 2008

KEY WORDS: finite elements; exact integration; stiffness matrix; triangular elements

1. INTRODUCTION

This paper investigates the analytical integration of the stiffness matrices of straight-sided triangular finite elements (FEs) with the goal of developing more efficient software. An important component of this approach is the use of computer algebra systems (CAS) such as *Derive*, *Maple*, *Macsyma*, *Reduce* and *Mathematica* which have made the manipulation of very large and complicated analytical expressions possible. An important benefit of these systems is their ability to output complicated expressions directly into a high-level computing language such as Fortran, thereby eliminating any possibility of typographical errors.

The potential of using CAS in the development of FE computer codes based on symbolic manipulation has long been recognized. For example, the reader can refer to the early works of

^{*}Correspondence to: D. V. Griffiths, Colorado School of Mines, Golden, CO, U.S.A.

[†]E-mail: d.v.griffiths@mines.edu

Contract/grant sponsor: National Science Foundation; contract/grant number: INT-0106665

Andersen and Noor [1], Korncoff [2], Korncoff and Fenves [3], Jensen and Niordson [4], Babu and Pinder [5] among others.

Yagawa *et al.* [6] reported CPU time savings of 15% using a combined technique involving both numerical and analytical integrations of stiffness in plane elasticity. Kikuchi [7] used the *Reduce* package to obtain explicit formulas for an isoparametric 4-node FE, showing accurate results for distorted elements. Rathod [8] presented analytical integration formulas for a 4-node isoparametric FE and showed that all the integration formulas could be obtained on the basis of four simple integrals. Generation of Fortran code by CAS was discussed by Wang *et al.* [9] and Lawrence *et al.* [10] for triangles, and Shiakolas *et al.* [11] for tetrahedra.

Griffiths [12] presented a general methodology based on coordinate transformations to obtain the semi-analytical closed-form expression for the stiffness matrix of a 4-node elastic plane FE that was subsequently extrapolated to 8-node quadrilaterals by Cardoso [13]. ‘Semi-analytical’ in this context means that CAS were used to reproduce the approximate numerically integrated result symbolically. Similar formulations for 4-node elements applied to steady seepage analysis have also been presented by Smith and Griffiths [14]. Videla *et al.* [15, 16] and Lozada *et al.* [17] used the *Derive* and *Maple* packages to generate closed-form expressions for the integration of 4- and 8-node elastic plane strain FEs and reported significant CPU savings. Amberg *et al.* [18] made extensive use of the *Maple* system to generate complete FE codes in Fortran for applications to research on welding, crystal growth, flow of viscoelastic liquids and more. Chee *et al.* [19] presented an FE development environment based on the *Mathematica* computer software system, which was used to automatically program standard element formulations and develop new elements with novel features. Felippa [20] presented a set of *Mathematica* modules of numerical integration rules.

This work addresses the plane strain stiffness matrix of 3-, 6-, 10- and 15-noded straight-sided triangular FEs. Straight-sided triangular elements lead to relatively simple formulations, due largely to the fact that the determinant of the Jacobian matrix is constant. This means that the stiffness matrix can be integrated exactly using numerical integration. This is not, in general, true of quadrilateral elements, even with straight sides, where the determinant of the Jacobian matrix is a function of position within the element. The resulting expressions for quadrilaterals cannot, in general, be exactly integrated by numerical integration due to the appearance of logarithmic terms.

In the following development, we focus on computing the stiffness matrix of straight-sided triangular elements using analytical integration with the help of *Maple*. The resulting expressions are then coded into Fortran 95 subroutines, can be compared with their numerical counterparts for runtime efficiency.

2. REVIEW OF STIFFNESS FORMULATION

The standard formulation of an FE stiffness matrix (e.g. [21]) is briefly reviewed below.

The stiffness matrix $[\mathbf{k}_m]$ of a planar FE is typically evaluated using the integral

$$[\mathbf{k}_m] = \iint_{\text{area}} [\mathbf{B}]^T [\mathbf{D}] [\mathbf{B}] dx dy \quad (1)$$

where $[\mathbf{B}]$ is the strain–displacement matrix and $[\mathbf{D}]$ is the elastic stress–strain matrix.

This paper discusses the methods of evaluation of Equation (1) for planar straight-sided triangular elements such as those shown in Figure 1.

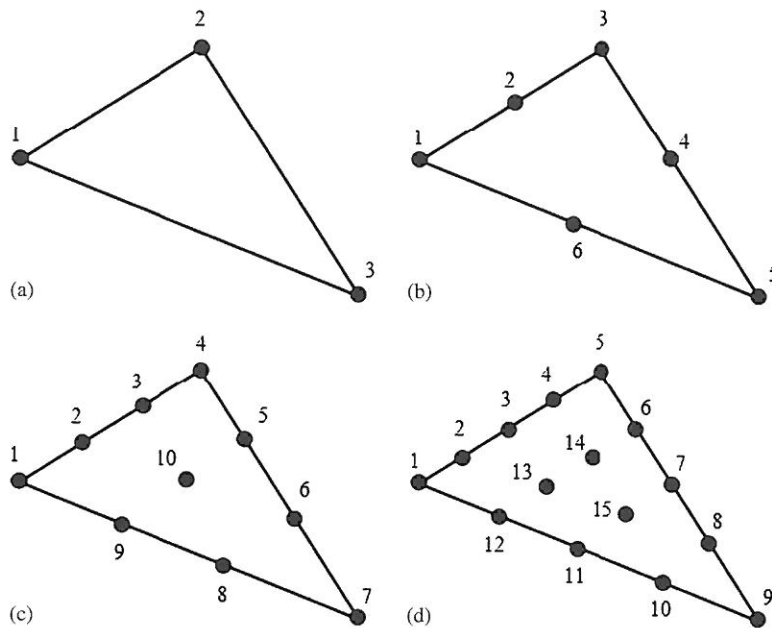


Figure 1. Node numbering adopted for straight-sided triangular elements: (a) 3-node; (b) 6-node; (c) 10-node; and (d) 15-node.

For a typical triangular element such as that shown in Figure 1 with alternating x , y freedom numbering in the same order as the nodal numbering, the $[\mathbf{B}]$ matrix takes the form

$$[\mathbf{B}] = \begin{bmatrix} \frac{\partial N_1}{\partial x} & 0 & \frac{\partial N_2}{\partial x} & 0 & \dots & \frac{\partial N_n}{\partial x} & 0 \\ 0 & \frac{\partial N_1}{\partial y} & 0 & \frac{\partial N_2}{\partial y} & \dots & 0 & \frac{\partial N_n}{\partial y} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial y} & \frac{\partial N_2}{\partial x} & \dots & \frac{\partial N_n}{\partial y} & \frac{\partial N_n}{\partial x} \end{bmatrix} \quad (2)$$

where n is the number of nodes, and N_i , $i = 1, 2, \dots, n$, are the shape functions of the element (see, e.g. [14]).

The $[\mathbf{D}]$ matrix can be expressed as

$$[\mathbf{D}] = \begin{bmatrix} E_1 & E_2 & 0 \\ E_2 & E_1 & 0 \\ 0 & 0 & G \end{bmatrix} \quad (3)$$

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

For plane stress

$$E_1 = \frac{E}{1-\nu^2}, \quad E_2 = \nu E_1 \quad (4)$$

and for plane strain

$$E_1 = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)}, \quad E_2 = \frac{\nu E_1}{(1-\nu)} \quad (5)$$

Shape functions are typically expressed in local coordinates (L_1, L_2) based on the local coordinate system shown in Figure 2; hence, the following standard transformation is used to deliver the shape function derivative with respect to Cartesian coordinates (x, y) as needed by the $[\mathbf{B}]$ matrix in Equation (2)

$$\begin{bmatrix} \frac{\partial N_1}{\partial x} & \frac{\partial N_2}{\partial x} & \frac{\partial N_3}{\partial x} & \dots & \frac{\partial N_n}{\partial x} \\ \frac{\partial N_1}{\partial y} & \frac{\partial N_2}{\partial y} & \frac{\partial N_3}{\partial y} & \dots & \frac{\partial N_n}{\partial y} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial L_1} & \frac{\partial y}{\partial L_1} \\ \frac{\partial x}{\partial L_2} & \frac{\partial y}{\partial L_2} \end{bmatrix}^{-1} \begin{bmatrix} \frac{\partial N_1}{\partial L_1} & \frac{\partial N_2}{\partial L_1} & \frac{\partial N_3}{\partial L_1} & \dots & \frac{\partial N_n}{\partial L_1} \\ \frac{\partial N_1}{\partial L_2} & \frac{\partial N_2}{\partial L_2} & \frac{\partial N_3}{\partial L_2} & \dots & \frac{\partial N_n}{\partial L_2} \end{bmatrix} \quad (6)$$

The Jacobian matrix shown inverted in Equation (6) is readily obtained from the isoparametric relations

$$[\mathbf{J}] = \begin{bmatrix} \frac{\partial x}{\partial L_1} & \frac{\partial y}{\partial L_1} \\ \frac{\partial x}{\partial L_2} & \frac{\partial y}{\partial L_2} \end{bmatrix} = \begin{bmatrix} \frac{\partial N_1}{\partial L_1} & \frac{\partial N_2}{\partial L_1} & \frac{\partial N_3}{\partial L_1} & \dots & \frac{\partial N_n}{\partial L_1} \\ \frac{\partial N_1}{\partial L_2} & \frac{\partial N_2}{\partial L_2} & \frac{\partial N_3}{\partial L_2} & \dots & \frac{\partial N_n}{\partial L_2} \end{bmatrix} \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ \vdots & \vdots \\ x_n & y_n \end{bmatrix} \quad (7)$$

where (x_i, y_i) , $i = 1, 2, \dots, n$, are the Cartesian nodal coordinates (see Figure 1).

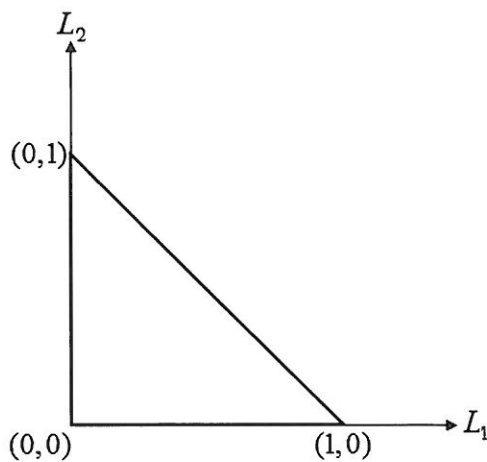


Figure 2. Local coordinate system for triangular elements.

The integration shown in Equation (1) is conveniently performed in the local coordinate space through the following transformation:

$$[\mathbf{k}_m] = \iint_{\text{area}} [\mathbf{B}]^T [\mathbf{D}] [\mathbf{B}] dx dy = \det(\mathbf{J}) \int_0^1 \int_0^{1-L_2} [\mathbf{B}]^T [\mathbf{D}] [\mathbf{B}] dL_1 dL_2 \quad (8)$$

where $\det(\mathbf{J})$ for straight-sided triangles is constant and equal to twice the area A of the element; thus,

$$\det(\mathbf{J}) = 2A \quad (9)$$

If the coordinates of the vertices of a triangular element in a clockwise sense are given by (x_a, y_a) , (x_b, y_b) and (x_c, y_c) , the area is conveniently given by the identity

$$A = \frac{1}{2}(x_a y_b + x_a y_c - x_b y_c + y_a x_b - y_a x_c + y_b x_c) \quad (10)$$

3. NUMERICAL INTEGRATION

Equation (8) is typically evaluated numerically using a summation of the form

$$k_{ij} = 2A \sum_{k=1}^{n_p} w_k f(L_1, L_2)_k \quad (11)$$

where $f(L_1, L_2)$ is the function to be integrated, n_p , is the number of sampling points, and w_k and $(L_1, L_2)_k$ are the weighting coefficients and local sampling point coordinates, respectively.

For straight-sided triangular elements, numerical integration of the stiffness matrix is exact, provided the correct (minimum) number of sampling points is used as indicated in Table I.

The weights and sampling points for these rules have been summarized in Appendix A. The first three integration rules with $n_p = 1, 3$ and 7 can be expressed in the exact form (e.g. [20]), although to the author's knowledge, these quantities for the 12-point rule are available only in the decimal form. Nevertheless, these numerical integration rules provide useful validation tools for the analytical results presented subsequently.

Table I. Minimum number of integrating points needed for exact integration of straight-sided triangular element stiffness matrices.

Element	n_p
3-Node triangle	1
6-Node triangle	3
10-Node triangle	7
15-Node triangle	12

4. ANALYTICAL INTEGRATION

If Equation (8) is expanded term by term, the following integral expression is obtained:

$$k_{ij} = 2A \int_0^1 \int_0^{1-L_2} (E_1(B_{1i}B_{1j} + B_{2i}B_{2j}) + E_2(B_{1i}B_{2j} + B_{2i}B_{1j}) + G B_{3i}B_{3j}) dL_1 dL_2 \quad (12)$$

which in this section will be integrated directly with the help of CAS such as *Maple*. The test example used to demonstrate the methodology is shown in Figure 3. The element is a scalene triangle with nodal coordinates (1.5, 0.0), (2.0, 2.0) and (3.5, 1.0). In all cases, the node at (1.5, 0.0) is assumed to be node 1. Although this paper makes the software that generates the entire stiffness matrices of triangular element available, in order to demonstrate the methodology, the particular term $k_{k,l}$, which couples freedoms k and l as shown by the arrows in Figure 3, will be described in more detail.

With reference to Figure 1, stiffness term $k_{k,l}$ from Figure 3 corresponds to the actual stiffness terms shown in Table II for the triangular elements under consideration.

Appendix B contains a typical *Maple* program for analytical calculation of the plane strain stiffness matrices of a 6-node triangular element. Similar programs were written for the other triangular elements. The program follows the standard FE development described in Equations (1)–(8). The particular term to be computed is set by the row/column values of ii and jj , respectively. The exact results for the test example shown in Figure 3 using the different triangular elements are given in Table III.

The entire stiffness matrix can be developed in the same manner by cycling ii and jj through all the rows and columns. The exact expressions have been cut and pasted into Fortran 95

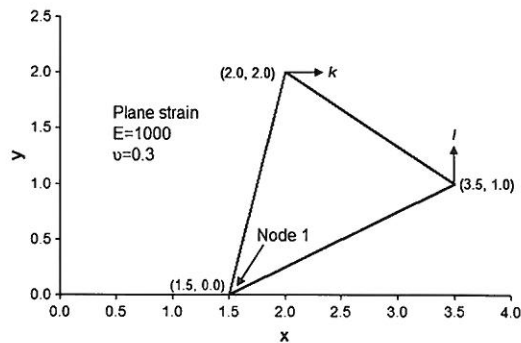


Figure 3. Corner coordinates of test element.

Table II. Stiffness term corresponding to $k_{k,l}$ for different triangular elements.

Element	Term
3-Node triangle	$k_{3,6}$
6-Node triangle	$k_{5,10}$
10-Node triangle	$k_{7,14}$
15-Node triangle	$k_{9,18}$

Table III. Stiffness term corresponding to $k_{k,l}$ in the test example for different element types.

Element	Term	Exact	Decimal
3-Node triangle	$k_{3,6}$	$\frac{23750}{273}$	260.9890110
6-Node triangle	$k_{5,10}$	$-\frac{23750}{273}$	-86.9963370
10-Node triangle	$k_{7,14}$	$\frac{2375}{52}$	45.6730769
15-Node triangle	$k_{9,18}$	$-\frac{508250}{17199}$	-29.5511367

subroutines which are available on the first author's Web site at www.mines.edu/~vgriffit/analytical.

5. OBSERVATIONS ON THE EXACT EXPRESSIONS

In addition to stiffness matrix symmetry, the analytical evaluation of stiffness terms revealed common factors, zero terms and repetitions. All these features could be exploited in the Fortran coding to secure greater efficiency.

5.1. Common factors

All the terms in the stiffness matrices contain the common factor

$$c_f = \frac{E}{2A(1+\nu)(1-2\nu)} \quad (13)$$

5.2. Zero and repeated terms

The 6- and 10-node element stiffness matrices contain terms that are always equal to zero. The 6-, 10- and 15-node elements contain repeated terms that are always equal. These patterns are independent of the nodal coordinates or elastic properties. The zero and repeated terms are indicated in Appendices C–E, where zero entries indicate zero terms and integer entries indicate repeated terms that are always equal. For example, for the 6-node element, term $k_{1,7}=0$, as indicated by the '0', and $k_{1,4}=k_{3,6}$, as indicated by the two entries of the integer '2'. This notation is applied only to the upper triangle; however, symmetry is assumed.

6. COMPARISON OF NUMERICAL AND ANALYTICAL COMPUTER TIMINGS

The final section of this paper presents timing comparisons for the generation of triangular element stiffness matrices by both the analytically generated code and the conventional numerical integration approach. The numerical integrations used the standard number of sampling points summarized in Table I. Stiffness matrices generated by the numerical and analytical approaches were always identical within the limitations of machine accuracy.

In all cases, the stiffness matrix of the test element was computed 10 million times so that the runs would take long enough to be measurable and reproducible. Timings were performed on

Table IV. Computer platforms used for timing comparisons.

Platform	CPU	L2 cache	FSB	RAM
A	Genuine Intel(R) CPU T2500 @ 2.00 GHz	2M	667 MHz	2G DDR2-667 MHz
B	Intel® Core™2 Duo CPU T7300 @2.00 GHz	4M	800 MHz	2G DDR2-667 MHz
C	Pentium® 4 CPU 3.2 GHz	1M	800 MHz	1G DDR-400 MHz
D	Intel® Core™2 Duo CPU E6700 @2.66 GHz	4M	1066 MHz	4G DDR2-667 MHz
E	Intel® Core™ 2 Duo CPU E6600 @2.40 GHz	4M	1066 MHz	2G DDR2-667 MHz

Table V. Timing results of the stiffness matrix generation for the test element on different platforms.

Platform	Element	Numerical	Analytical	Speedup
A	3-Node	62	7	9
	6-Node	333	22	15
	10-Node	1377	60	23
	15-Node	4114	188	22
B	3-Node	70	6	12
	6-Node	307	19	16
	10-Node	1191	51	23
	15-Node	3439	155	22
C	3-Node	64	6	11
	6-Node	338	20	17
	10-Node	1435	55	26
	15-Node	4239	155	27
D	3-Node	56	5	11
	6-Node	247	16	15
	10-Node	957	49	20
	15-Node	2754	158	17
E	3-Node	46	6	8
	6-Node	230	17	14
	10-Node	942	54	17
	15-Node	2854	172	17

Numbers are rounded to the nearest whole number and based on 10 million repetitions. Time unit is seconds.

several different platforms as listed in Table IV with timing results and speedup ratio as shown in Table V. Numbers have been rounded to the nearest whole number and it was noted that repeated runtimes varied by less than 1%. The tables indicate quite significant efficiency gains, with a speedup ratio of up to 27 for the 15-node triangle stiffness matrix generation.

7. COMMENTS ON THE EFFICIENCY OF THE NUMERICAL CODE

The speedup ratios shown in Table V are impressive, although no effort was put into ‘tuning’ either of the codes for maximum efficiency. This is particularly true of the numerical integration approach that used a traditional algorithm taken straight from the published F95 codes of Smith and Griffiths [14]. Although the numerical coding in this reference is very compact (about 10 lines, see Appendix F), a great deal of work is hidden behind the intrinsic Fortran 95 and

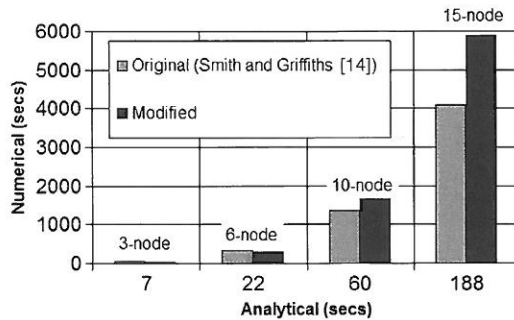


Figure 4. Comparison of timings of original and modified numerical codes.

user-supplied functions, particularly `MATMUL`. It may also be noted that the conventional numerical code calculates all the terms of the stiffness matrix as if they were independent and does not take account of symmetry. The analytical code (over 1700 lines for the 15-node triangle) computes terms in the upper triangle using the long but relatively simple arithmetic operations generated by the CAS, and then copies over the symmetric terms. Appendix G shows the Fortran for a typical term from the stiffness matrix of the 15-node triangle.

In order to address this potential source of numerical inefficiency, the code shown in Appendix F was rewritten to eliminate `MATMUL`.

Firstly, the `MATMUL` terms used to generate the $[B]$ matrix from Equation (1) were replaced by the source code that evaluated the matrix products using conventional DO-loops. This made virtually no difference to the numerical runtimes, and if anything ran slightly slower than the version including `MATMUL`.

Secondly, the calls to `MATMUL` and `TRANPOSE` used to create the matrix product $[B]^T[D][B]$ in a single line as shown in Appendix F were replaced by a much longer version that generated each term of this matrix explicitly. This approach avoided the use of both `MATMUL` and DO-loops and a typical term of this longer version for a 15-node triangle is shown in Appendix H. It should be noted that symmetric terms are not computed twice and are simply copied over as in the analytical version. Figure 4 shows the comparison between the original numerical version [14] and a 'modified' version incorporating both the changes mentioned above. The modified version ran slightly faster for lower-order elements, but more importantly, it ran significantly slower for higher-order elements. In all cases the benefits of the analytical approach were still very significant and the speedup ratios of Table V are still valid.

8. CONCLUDING REMARKS

Most proprietary FE software uses numerical integration to compute FE elastic stiffness matrices. Although the numerical approach is convenient and generally accurate, it is not always necessary and analytical formulations can give identical solutions in a fraction of time. This paper has described the use of CAS to exactly integrate the stiffness matrices of straight-sided triangular FEs. The methodology has been applied to 3-, 6-, 10- and 15-noded elements. Although the resulting expressions can be algebraically long, they give identical results to those obtained by numerical

integration, but with significantly faster CPU times. The numerical and analytical approaches were compared on a variety of computer platforms. The greatest improvement observed in this study resulted in a speedup ratio of about 27 for a 15-noded triangle. Attempts were made to optimize the numerical code by removing the intrinsic function `MATMUL` and calls to `DO`-loops. Little improvement was observed, and for higher-order elements the modified numerical code ran slower than the original version.

A key point of this study, however, is that numerical integration is not necessary for elastic triangular straight-sided FEs unless some numerical device such as ‘reduced integration’ is preferred. Even then, CAS offer much potential for developing software that runs more efficiently.

Software downloads. All the programs used in this study can be downloaded from the first author’s Web site at www.mines.edu/~vgriffit/analytical.

APPENDIX A: WEIGHTING COEFFICIENTS AND SAMPLING POINTS FOR TRIANGLES

Number of nodes (n)	Number of sampling points (n_p)	Weights (w_k)	Sampling points $(L_1, L_2)_k$
3	1	$\frac{1}{2}$	$(\frac{1}{3}, \frac{1}{3})$
6	3	$\frac{1}{6}$	$(\frac{1}{2}, \frac{1}{2})$
		$\frac{1}{6}$	$(\frac{1}{2}, 0)$
		$\frac{1}{6}$	$(0, \frac{1}{2})$
10	7	$\frac{9}{80}$	$(\frac{1}{3}, \frac{1}{3})$
		$(155 - \sqrt{15})/2400$	$((9 + 2\sqrt{15})/21, (6 - \sqrt{15})/21)$
		$(155 - \sqrt{15})/2400$	$((6 - \sqrt{15})/21, (9 + 2\sqrt{15})/21)$
		$(155 - \sqrt{15})/2400$	$((6 - \sqrt{15})/21, (6 - \sqrt{15})/21)$
		$(155 - \sqrt{15})/2400$	$((6 + \sqrt{15})/21, (9 - 2\sqrt{15})/21)$
		$(155 - \sqrt{15})/2400$	$((9 - 2\sqrt{15})/21, (6 + \sqrt{15})/21)$
15	12	0.025422453185103	$((6 + \sqrt{15})/21, (6 + \sqrt{15})/21)$
		0.873821971016995	$(0.873821971016995, 0.063089014491502)$
		0.063089014491502	$(0.063089014491502, 0.873821971016995)$
		0.063089014491502	$(0.063089014491502, 0.063089014491502)$
		0.058393137863190	$(0.501426509658179, 0.249286745170910)$
		0.058393137863190	$(0.249286745170910, 0.501426509658179)$
		0.058393137863190	$(0.249286745170910, 0.249286745170910)$
		0.041425537809187	$(0.053145049844817, 0.310352451033784)$
		0.041425537809187	$(0.310352451033784, 0.053145049844817)$
		0.041425537809187	$(0.053145049844817, 0.636502499121398)$
		0.041425537809187	$(0.310352451033784, 0.636502499121398)$
		0.041425537809187	$(0.636502499121398, 0.053145049844817)$
0.041425537809187	$(0.636502499121398, 0.310352451033784)$		

APPENDIX B: MAPLE PROGRAM FOR 6-NODE TRIANGLE STIFFNESS

```

> restart: with(linalg): nod:=6:
> der:=Matrix(2,nod): fun:=Matrix(nod,1):
> coord:=Matrix(nod,2): bee:=Matrix(3,2*nod):
> E1:= E*(1- nu)/(1+ nu)/(1-2* nu): E2:=E1* nu/(1- nu): G:= E/2/(1+ nu):
> coord[1,1]:=x1: coord[1,2]:=y1:
> coord[2,1]:=(x1+x3)/2: coord[2,2]:=(y1+y3)/2:
> coord[3,1]:=x3: coord[3,2]:=y3:
> coord[4,1]:=(x3+x5)/2: coord[4,2]:=(y3+y5)/2:
> coord[5,1]:=x5: coord[5,2]:=y5:
> coord[6,1]:=(x5+x1)/2: coord[6,2]:=(y5+y1)/2:
> fun[1,1]:=(2*L1-1)*L1: fun[2,1]:=4*(1-L1-L2)*L1:
> fun[3,1]:=(2*(1-L1-L2)-1)*(1-L1-L2): fun[4,1]:=4*L2*(1-L1-L2):
> fun[5,1]:=(2*L2-1)*L2: fun[6,1]:=4*L1*L2:
> for kk to nod do
>   der[1,kk]:=diff(fun[kk,1],L1):
>   der[2,kk]:=diff(fun[kk,1],L2):
> end do:
> jac:=multiply(der,coord): detJ:=det(jac):
> deriv:=simplify(multiply(inverse(jac),der)):
> for kk to nod do
>   bee[1,2*kk-1]:=deriv[1,kk]: bee[2,2*kk]:=deriv[2,kk]:
>   bee[3,2*kk-1]:=deriv[2,kk]: bee[3,2*kk]:=deriv[1,kk]:
> end do:
> ii:=5: jj:=10:
> kint:=detJ*simplify(E1*(bee[1,ii]*bee[1,jj]+bee[2,ii]*bee[2,jj])+
>   E2*(bee[1,ii]*bee[2,jj]+bee[2,ii]*bee[1,jj])+
>   G*bee[3,ii]*bee[3,jj]):
> km[ii,jj]:=simplify(int(int(kint,L1=0..1-L2),L2=0..1));
> km[ii,jj]:=subs({x1=3/2,x3=2,x5=7/2,y1=0,y3=2,y5=1,
E=1000,nu=3/10},km[ii,jj]);
> km[ii,jj]:=evalf(%);

km[5, 10] := - (-2 nu x1 y5 + 2 nu x3 y5 - 2 nu y1 x3 + y1 x1- x1 y3 +
2 x1 y3 nu - y1 x5 + 2 y1 x5 nu + y3 x5- 2 y3 x5 nu) E/(12(x1 y3 - x1
y5 + x3 y5 - y1 x3 + y1 x5 - y3 x5) (-1 + 2 nu) (1 + nu))

-23750
km[5, 10] := -----
273

km[5, 10] := -86.99633700

```

APPENDIX C: REPEATED AND ZERO TERMS IN 6-NODE ELEMENT STIFFNESS MATRIX

	1	2	3	4	5	6	7	8	9	10	11	12
1			1	2			0	0			3	4
2			5	6			0	0			7	8
3			9	10	1	2		11	0	0		12
4				13	5	6	11		0	0	12	
5							14	15			0	0
6							16	17			0	0
7							9	10	14	15		18
8								13	16	17	18	
9											3	7
10											4	8
11											9	10
12												13

APPENDIX D: REPEATED AND ZERO TERMS IN 10-NODE ELEMENT STIFFNESS MATRIX

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
1									1	2	1	2								0	0
2									2	3	2	3								0	0
3			4	5					6	7	6	7	8	9	10	11		12	13	14	
4				15					7	16	7	16	9	17	11	18	12		14	19	
5					4	5				20	6	7	8	9	10	11	10	11	21	22	
6						15					7	16	9	17	11	18	11	18	22	23	
7															24	25	24	25	0	0	
8															25	26	25	26	0	0	
9									4	5					27	28	27	28	29	30	
10										15					28	31	28	31	30	32	
11											4	5				33	27	28	13	14	
12												15				33		28	31	14	19
13																				0	0
14																				0	0
15															4	5				21	22
16																15				22	23
17																	4	5	29	30	
18																		15	30	32	
19																					
20																					

APPENDIX E: REPEATED TERMS IN 15-NODE ELEMENT STIFFNESS MATRIX

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1											1	2	1	2	1
2											2	5	2	5	2
3			7	8								9		10	
4				20							9		10		11
5												23		24	
6											23		24		25
7							32	33				34		35	
8								43			34		35		36
9															
10															
11											32	33			
12												43			
13															
14															
15															64

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	2									3	4	3	4	3	4
2	5									4	6	4	6	4	6
3	11	12	13		14		15		16		17	18	19	18	19
4		13	21	14		15		16		17		19	22	19	22
5	25	12	13		26		27		28		29		30		31
6		13	21	26		27		28		29		30		31	
7	36	12	13		37		38		39	40	41		42	40	41
8		13	21	37		38		39		41	44	42		41	44
9				45	46	45	46	45	46	47	48	47	48	47	48
10				46	49	46	49	46	49	48	50	48	50	48	50
11					51		52		53	54	55		56	54	55
12				51		52		53		55	57	56		55	57
13					58		59		60		61		62		63
14				58		59		60		61		62		63	
15	65				66		67		68	69	70	69	70		71

	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
16	72			66		67		68		70	73	70	73	71	
17										74	75	74	75	74	75
18										75	76	75	76	75	76
19				64	65					77	78	77	78		79
20					72					78	80	78	80	79	
21											81		82		83
22										81		82		83	
23								7	8		84	85	86	85	86
24									20	84		86	87	86	87
25										88	89		90		91
26											92	90		91	
27												88	89		93
28													92	93	
29														88	89
30															92

APPENDIX F: EXTRACT OF THE CODING FROM THE NUMERICAL
INTEGRATION SUBROUTINE

```

km=zero
int_pts_1: DO i=1,nip
  CALL shape_der (der,points,i)
  jac=MATMUL (der,coord)
  det=determinant (jac)
  CALL invert (jac)
  deriv=MATMUL (jac,der)
  CALL beemat (bee,deriv)
  km=km+MATMUL (MATMUL (TRANPOSE (bee) ,dee) ,bee) *det*weights (i)
END DO int_pts_1

```

Notation: *i*, DO-loop counter; *nip*, number of integrating points; *det*, determinant of the Jacobian matrix; *km*, element stiffness matrix; *der*, matrix holding derivatives of shape functions w.r.t. local coordinates; *points*, matrix holding sampling points and weighting coefficients; *jac*, Jacobian matrix (later holds the inverse of the Jacobian matrix); *coord*, matrix holding element nodal coordinates; *deriv*, matrix holding derivatives of shape functions w.r.t. global coordinates; *bee*, strain–displacement matrix; *dee*, stress–strain matrix; *weights*, matrix holding numerical integration weighting coefficients.

APPENDIX G: EXTRACT OF THE CODING FROM THE EXACTLY INTEGRATED
15-NODE ELEMENT STIFFNESS MATRIX SUBROUTINE

```

cf=ym/(x1*y5-x1*y9+x5*y9-y1*x5+y1*x9-y5*x9)/(-d1+d2*pr)/(d1+pr)
!-----
km(1,1)=-d47/d252*(-d2*y5**2+d4*y5*y9-d2*y9**2+d2*pr*y5**2-      &
d4*pr*y5*y9+d2*pr*y9**2-x5**2+d2*x5**2*pr+d2*x5*x9-d4*x5*x9*pr-x9**2+  &
d2*x9**2*pr)*cf
km(1,2)=-d47/d252*(x5-x9)*(y5-y9)*cf
km(1,3)=d4/d945*(-d92*x5*x1+d184*y9*y1+d92*x9*x1-d184*y5*y1+d30*y5**2+      &
d124*y5*y9-d154*y9**2-d30*pr*y5**2+d154*pr*y9**2+d15*x5**2+d62*x5*x9-      &
d77*x9**2-d30*x5**2*pr+d154*x9**2*pr-d124*pr*y5*y9+d184*pr*y5*y1-      &
d184*pr*y9*y1-d124*x5*x9*pr-d184*x9*x1*pr+d184*x5*x1*pr)*cf
km(1,4)=d4/d945*(d184*pr*x5*y9-d184*pr*y5*x9+d92*y1*x5-d92*y1*x9+      &
d15*y5*x9-d77*x5*y9-d15*y5*x5+d184*x9*y1*pr-d184*x5*y1*pr+      &
d184*pr*y5*x1-d184*pr*y9*x1+d77*y9*x9)*cf
km(1,5)=-(-d201*x5*x1+d402*y9*y1+d201*x9*x1-d402*y5*y1-d40*y5**2+      &
d482*y5*y9-d442*y9**2+d40*pr*y5**2+d442*pr*y9**2-d20*x5**2+d241*x5*x9-      &
d221*x9**2+d40*x5**2*pr+d442*x9**2*pr-d482*pr*y5*y9+d402*pr*y5*y1-      &
d402*pr*y9*y1-d482*x5*x9*pr-d402*x9*x1*pr+d402*x5*x1*pr)/d945*cf
.
.
km(2,1)=km(1,2)
km(3,1)=km(1,3)
km(4,1)=km(1,4)
km(5,1)=km(1,5)

```

Notation: cf, factor common to all terms; x1, y1, global coordinates of node 1 (see Figure 1); x5, y5, global coordinates of node 1 (see Figure 1); x9, y9, global coordinates of node 1 (see Figure 1); ym, Young's modulus; pr, Poisson's ratio; d252, the real number 252.0, etc.; km(1, 2), the stiffness term $k_{1,2}$, etc.

APPENDIX H: TYPICAL TERM OF THE MATRIX $[\mathbf{B}]^T[\mathbf{D}][\mathbf{B}]$ FOR A 15-NODE TRIANGLE
IN LONG FORM AVOIDING THE USE OF MATMUL, TRANSPOSE OR DO-LOOPS

```

btdb(2,1)=(bee(1,2)*dee(1,1)+bee(2,2)*dee(2,1)+bee(3,2)*dee(3,1))*bee(1,1)+
(bee(1,2)*dee(1,2)+bee(2,2)*dee(2,2)+bee(3,2)*dee(3,2))*bee(2,1)+
(bee(1,2)*dee(1,3)+bee(2,2)*dee(2,3)+bee(3,2)*dee(3,3))*bee(3,1)
btdb(1,2)=btdb(2,1)

```

Notation: btdb, matrix formed by the product $[\mathbf{B}]^T[\mathbf{D}][\mathbf{B}]$ (see Equation (1)); bee, $[\mathbf{B}]$ matrix (strain-displacement); dee, $[\mathbf{D}]$ matrix (stress-strain).

ACKNOWLEDGEMENTS

The authors acknowledge the support provided by the National Science Foundation by grant number INT-0106665 from the Americas Program.

REFERENCES

1. Andersen CM, Noor AK. Use of symbolic manipulation in development of 2-dimensional finite elements. *SIAM Review* 1974; **16**(12):115–116.
2. Korncoff AR. Finite element stiffness matrix generation via symbolic manipulation. In *Proceedings of the Symposium on Application of Computer Methods in Engineering*, Wellford C (ed.), vol. II, 1977; 779–785.
3. Korncoff AR, Fenves SJ. Symbolic generation of finite element stiffness matrices. *Computers and Structures* 1979; **10**:119–124.
4. Jensen J, Niordson F. Symbolic and algebraic manipulation languages and their application in mechanics. In *Structural Mechanics Software Series*, Perrone N, Pilkey W (eds). University Press of Virginia: Charlottesville, VA, 1977.
5. Babu DK, Pinder GF. Analytical integration formulae for linear isoparametric finite elements. *International Journal for Numerical Methods in Engineering* 1984; **20**(6):1153–1161.
6. Yagawa G, Ye GW, Yoshimura S. A numerical integration scheme for finite element method based on symbolic manipulation. *International Journal for Numerical Methods in Engineering* 1990; **29**:1539–1549.
7. Kikuchi M. Application of the symbolic mathematics system to the finite element program. *Computational Mechanics* 1989; **5**:41–47.
8. Rathod HD. Some Analytical integration formulae for a four node isoparametric element. *Computers and Structures* 1988; **30**(5):1101–1109.
9. Wang PS, Chang TYP, van Hulzen JA. Code generation and optimization for finite element analysis. *International Symposium on Symbolic and Algebraic Computation, EUROSAM 84*, Cambridge, U.K., 1984; 237–241.
10. Lawrence KL, Nambiar RV, Bergmann B. Closed form stiffness matrices and error estimators for plane hierarchical triangular elements. *International Journal for Numerical Methods in Engineering* 1991; **31**(5):879–894.
11. Shiakolas PS, Nambiar RV, Lawrence KL, Rogers WA. Closed-form stiffness matrices for the linear strain and quadratic strain tetrahedron finite elements. *Computers and Structures* 1992; **5**(2):237–242.
12. Griffiths DV. Stiffness matrix of the four-node quadrilateral element in closed form. *International Journal for Numerical Methods in Engineering* 1994; **37**(6):1027–1038.
13. Cardoso J-P. Generation of finite element matrices using computer algebra. *M.Sc. Thesis*, School of Engineering, University of Manchester, 1994.
14. Smith IM, Griffiths DV. *Programming the Finite Element Method* (4th edn). Wiley: Chichester, U.K., 2004.
15. Videla L, Aparicio N, Cerrolaza M. Explicit integration of the stiffness matrix of a four-noded-plane elasticity finite element. *Communications in Numerical Methods in Engineering* 1996; **12**:731–743.
16. Videla L, Baloa T, Griffiths DV, Cerrolaza M. Exact integration of the stiffness matrix of an 8-node plane elastic finite element by symbolic computation. *Numerical Methods for Partial Differential Equations* 2008; **24**(1):249–261.
17. Lozada IJ, Osorio JC, Griffiths DV, Cerrolaza M. Semi-analytical integration of the 8-node plane element stiffness matrix using symbolic computation. *Numerical Methods for Partial Differential Equations* 2006; **22**(2):296–316.
18. Amberg G, Tonhardt R, Winkler C. Finite element simulations using symbolic computing. *Mathematics and Computers in Simulation* 1999; **49**:257–274.
19. Chee KY, Boyle JT, Mackenzie D. A computer algebra based finite element environment. *Advances in Engineering Software* 2001; **32**:913–925.
20. Felippa CA. A compendium of FEM integration formulas for symbolic work. *Engineering Computations* 2004; **21**:867–890.
21. Zienkiewicz OC. *The Finite Element Method* (3rd edn). McGraw-Hill: Englewood Cliffs, NJ, 1977.