Gauss Legendre Quadrature Formulas over a Tetrahedron

H. T. Rathod,[1] **B. Venkatesudu,**[2] **K. V. Nagaraja**[3]

[1]*Department of Mathematics, Central College Campus, Bangalore University, Bangalore 560 001, India*

[2]*Department of Mathematics, The Oxford College of Engineering, Bangalore 560 068, India*

[3]*Department of Mathematics, Amrita Institute of Technology, Bangalore 560 037, India*

Received 4 July 2004; accepted 22 February 2005
Published online 10 June 2005 in Wiley InterScience (www.interscience.wiley.com).
DOI 10.1002/num.20095

In this article we consider the Gauss Legendre Quadrature method for numerical integration over the standard tetrahedron: $\{(x, y, z)|0 \leq x, y, z \leq 1, x + y + z \leq 1\}$ in the Cartesian three-dimensional (x, y, z) space. The mathematical transformation from the (x, y, z) space to (ξ, η, ζ) space is described to map the standard tetrahedron in (x, y, z) space to a standard 2-cube: $\{(\xi, \eta, \zeta)| - 1 \leq \zeta, \eta, \zeta \leq 1\}$ in the (ξ, η, ζ) space. This overcomes the difficulties associated with the derivation of new weight coefficients and sampling points. The effectiveness of the formulas is demonstrated by applying them to the integration of three nonpolynomial, three polynomial functions and to the evaluation of integrals for element stiffness matrices in linear three-dimensional elasticity. © 2005 Wiley Periodicals, Inc. Numer Methods Partial Differential Eq 22: 197–219, 2006

Keywords: finite element method; Gauss legendre quadrature; numerical integration; tetrahedral finite elements; standard 2-cube; standard tetrahedron

1. INTRODUCTION

The integration theory is extended from real line to the plane and three-dimensional space by the introduction of multiple integrals. Integration is of fundamental importance in both pure and applied mathematics as well as in several areas of science and engineering. Most such integrals cannot be evaluated explicitly or analytically and with many others, it is often faster to integrate them numerically rather than evaluate them exactly using the complicated antiderivatives of the

Correspondence to: H. T. Rathod, Department of Mathematics, Central College Campus, Bangalore University, Bangalore 560 001, India (e-mail: htrathod@yahoo.com)

integrands. However it is not practical to obtain specific integration formulas for all regions of interest. Hence it is desirable to obtain integration formulas over two-simplex, the unit right-angled triangle and the three-simplex, the unit orthogonal tetrahedron which may be used in principle to approximate the plane or three-dimensional region of any physical space.

In recent years, we have been witnessing finite element method (FEM) gaining importance due to the most obvious reason that it can provide solutions to many complicated problems that would be intractable by other numerical techniques [1, 2]. In FEM it may be possible to perform some of the integrations analytically, particularly if constant or linear elements are used to discretise the surface or boundary curve of the given region. However, with higher order elements or for more complex distorted elements the integrals become too complicated for analytical integration and the numerical integration is essential, among various integration schemes, Gauss-Legendre quadrature which can evaluate exactly the $(2n-1)^{\text{th}}$ order polynomial with n-Gaussian points is most commonly used in view of the accuracy and efficiency of calculations [3]. The triangular and tetrahedral elements are very widely used in finite element analysis. The versatility of these elements can be further enhanced by improved numerical integration schemes. Mathematically, the problem can be defined as the evaluation of the following multiple integrals:

$$II = \int_0^1 \int_0^{1-L_1} F(L_1, L_2, L_3)\, dL_2 dL_1, \tag{1}$$

where L_1, L_2, L_3 are the well-known area co-ordinates and

$$III = \int_0^1 \int_0^{1-L_1} \int_0^{1-L_1-L_2} G(L_1, L_2, L_3, L_4)\, dL_3 dL_2 dL_1, \tag{2}$$

where L_1, L_2, L_3, L_4 are the well-known volume co-ordinates.

The basic problem of integrating an arbitrary function of two variables over the surface of the triangle were first given by Hammer et al. [4], and Hammer and Stroud [5, 6]. Cowper [7] provided a table of Gaussian quadrature formulas with symmetrically placed integration points. Lyness and Jespersen [8] made an elaborate study of symmetric quadrature rules by formulating the problem in polar co-ordinates. Lannoy [9] discussed the symmetric 4-point integration formula, which is presented in [7]. Laurie [10] derived a 7-point integration rule and discussed the numerical error in integrating some functions. Laursen and Gellert [11] gave a table of symmetric integration formulas up to a precision of degree ten. Lether [12] and Hillion [13] derived the formulas for triangles as product of one-dimensional Gauss Legendre and Gauss Jacobi quadrature rules. The precision of these formulas is up to degree seven. This is because the zeros and weight coefficients of Gauss Jacobi orthogonal polynomials with weight functions x, x^2, x^3 were available for polynomials of degree up to six only. Even today the zeros and weights for the integral $\int_0^1 x^r f(x)dx$, $r = 1, 2, 3$ are not available beyond a formula of order-eight as documented in Abramowicz and Stegun [14]. Reddy [15] and Reddy and Shippy [16] derived the 3-point, 4-point, 6-point, and 7-point rules of precision 3, 4, 6, and 7, respectively, which gave improved accuracy. Since the precision of all the formulas derived by the authors [4–6] is limited to a precision of degree 10 and it is not likely that the techniques can be extended much further to give a greater accuracy that may be demanded in future, Lague and Baldur [17] proposed product formulas based only on the sampling points and weight coefficients of Gauss Legendre quadrature rules. By the proposed method of [17] this restriction is removed and one can now obtain numerical integration rules of very high degree of precision as the derivation now rely on standard Gauss Legendre Quadrature rules. However, Lague and Baldur [17] have not worked out explicit weight coefficients and sampling points for application to integrals over a triangular surface. Rathod et al. [18] provided this information in a systematic manner in their recent work. For tetrahedral regions, four volume co-ordinates L_1, L_2, L_3, L_4 are involved and we have to compute numerically the integral III stated in eqn. (2). Numerical integration formulas for III with a degree of precision $d = 1, 2, 3$ are listed in Zienkiewicz [1] and these are based on reference [4]. Numerical integration formulas of precision higher than cubic are not available in the current literature and hence we propose here the derivation of higher order formulas for tetrahedral regions.

2. FORMULATION OF INTEGRALS OVER A TETRAHEDRON

The finite element method for three-dimensional problems with tetrahedron element requires the numerical integration of shape functions and their derivatives on a tetrahedron. Since an affine transformation makes it possible to transform any tetrahedron into the three-dimensional tetrahedron T with co-ordinates (0, 0, 0), (1, 0, 0), (0, 1, 0), and (0, 0, 1) in Cartesian three-dimensional space, say (x, y, z), we thus have to consider numerical integration on T. The numerical integration of an arbitrary function f, over the tetrahedron T is given by

$$I = \int\int\int_T f(x, y, z)dxdydz = \int_0^1 dx \int_0^{1-x} dy \int_0^{1-x-y} f(x, y, z)dz$$

$$= \int_0^1 dy \int_0^{1-y} dx \int_0^{1-x-y} f(x, y, z)dz. \quad (3)$$

It is now required to find the value of the integral by a quadrature formula:

$$I = \sum_{m=1}^{N} c_m f(x_m, y_m, z_m), \quad (4)$$

where c_m are the weights associated with the sampling points (x_m, y_m, z_m) and N is the number of pivotal points related to the required precision.

The integral I of eqn. (3) can be transformed into an integral over the cube: $\{(u, v, w)|0 \leq u, v, w \leq 1\}$ by the substitution

$$x = u, \; y = (1 - u)v, \; z = (1 - u)(1 - v)w. \quad (5)$$

Then the determinant of the Jacobian and the differential volume are

$$\frac{\partial(x, y, z)}{\partial(u, v, w)} = \begin{vmatrix} \frac{\partial x}{\partial u} & \frac{\partial x}{\partial v} & \frac{\partial x}{\partial w} \\ \frac{\partial y}{\partial u} & \frac{\partial y}{\partial v} & \frac{\partial y}{\partial w} \\ \frac{\partial z}{\partial u} & \frac{\partial z}{\partial v} & \frac{\partial z}{\partial w} \end{vmatrix} = (1-u)^2(1-v) \qquad \text{and}$$

$$dxdydz = \frac{\partial(x, y, z)}{\partial(u, v, w)} \qquad dudvdw = (1-u)^2(1-v)dudvdw. \tag{6}$$

Then on using eqns. (5) and (6) in eqn. (3), we have

$$I = \int_0^1 \left(\int_0^{1-x} \left(\int_0^{1-x-y} f(x, y, z)dz \right) dy \right) dx = \int_0^1 \int_0^1 \int_0^1 f(u, (1-u)v, (1-u)(1-v)w) \times (1-u)^2(1-v)dwdvdu. \tag{7}$$

The integral I of eqn. (7) can be further transformed into an integral over the standard 2-cube: $\{(\xi, \eta, \zeta) | -1 \leq \xi, \eta, \zeta \leq 1\}$ by the substitution

$$u = \frac{(1+\xi)}{2}, \qquad v = \frac{(1+\eta)}{2}, \qquad w = \frac{(1+\zeta)}{2}. \tag{8}$$

Then clearly, the determinant of the Jacobian and the differential volume are

$$\frac{\partial(u, v, w)}{\partial(\xi, \eta, \zeta)} = \frac{1}{8} \qquad \text{and} \qquad dudvdw = \frac{\partial(u, v, w)}{\partial(\xi, \eta, \zeta)} d\xi d\eta d\zeta = \frac{1}{8} d\xi d\eta d\zeta. \tag{9}$$

Now on using eqns. (8) and (9) in eqn. (7), we have

$$I = \int_0^1 \left(\int_0^{1-x} \left(\int_0^{1-x-y} f(x, y, z)dz \right) dy \right) dx$$
$$= \int_{-1}^1 \int_{-1}^1 \int_{-1}^1 f\left(\frac{(1+\xi)}{2}, \frac{(1-\xi)(1+\eta)}{4}, \frac{(1-\xi)(1-\eta)(1+\zeta)}{8} \right) \times \frac{(1-\xi)^2(1-\eta)}{64} d\xi d\eta d\zeta. \tag{10}$$

Equation (10) represents an integral over the standard 2-cube: $\{(\xi, \eta, \zeta) | -1 \leq \xi, \eta, \zeta \leq 1\}$. Efficient quadrature coefficients are readily available in the literature so that any desired accuracy can be obtained [14].

From eqns. (4) and (10), we find that

$$I = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} f\left(\frac{(1+\xi)}{2}, \frac{(1-\xi)(1+\eta)}{4}, \frac{(1-\xi)(1-\eta)(1+\zeta)}{8}\right)$$
$$\times \frac{(1-\xi)^2(1-\eta)}{64}\, d\xi d\eta d\zeta = \sum_{i=1}^{\alpha}\sum_{j=1}^{\beta}\sum_{k=1}^{\gamma} \frac{(1-\xi_i^{(\alpha)})^2(1-\eta_j^{(\beta)})}{64}\, \mathcal{W}_i^{(\alpha)}\mathcal{W}_j^{(\beta)}\mathcal{W}_k^{(\gamma)}$$
$$\times f\left(\frac{(1+\xi_i^{(\alpha)})}{2}, \frac{(1-\xi_i^{(\alpha)})(1+\eta_j^{(\beta)})}{4}, \frac{(1-\xi_i^{(\alpha)})(1-\eta_j^{(\beta)})(1+\zeta_k^{(\gamma)})}{8}\right)$$
$$= \sum_{m=1}^{N=(\alpha\times\beta\times\gamma)} c_m f(x_m, y_m, z_m), \quad (11)$$

where, it is obvious that

$$c_m = \frac{(1-\xi_i^{(\alpha)})^2(1-\eta_j^{(\beta)})}{64}\, \mathcal{W}_i^{(\alpha)}\mathcal{W}_j^{(\beta)}\mathcal{W}_k^{(\gamma)}, \qquad x_m = \frac{(1+\xi_i^{(\alpha)})}{2},$$
$$y_m = \frac{(1-\xi_i^{(\alpha)})(1+\eta_j^{(\beta)})}{4}, \qquad z_m = \frac{(1-\xi_i^{(\alpha)})(1-\eta_j^{(\beta)})(1+\zeta_k^{(\gamma)})}{8}, \quad (12)$$

in which $\xi_i^{(\alpha)}$, $\eta_j^{(\beta)}$, $\zeta_k^{(\gamma)}$ are the sampling points and $\mathcal{W}_i^{(\alpha)}$, $\mathcal{W}_j^{(\beta)}$, $\mathcal{W}_k^{(\gamma)}$ are the corresponding weight coefficients of Gauss Legendre quadrature rules of order α, β, and γ, respectively. In eqn. (10) let us now assume that $f(x, y, z) = x^p y^q z^r$, then we obtain

$$I = \int_0^1\int_0^{1-x}\int_0^{1-x-y} x^p y^q z^r\, dz dy dx$$
$$= \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} \left(\frac{1+\xi}{2}\right)^p \left(\frac{(1-\xi)(1+\eta)}{4}\right)^q \left(\frac{(1-\xi)(1-\eta)(1+\zeta)}{8}\right)^r \left(\frac{(1-\xi)^2(1-\eta)}{64}\right)$$
$$\times\ d\xi d\eta d\zeta = \int_{-1}^{1}\left(\sum_{l=0}^{p+q+r+2} A_l \xi^l\right) d\xi \int_{-1}^{1}\left(\sum_{m=0}^{q+r+1} B_m \eta^m\right) d\eta \int_{-1}^{1}\left(\sum_{p=0}^{r} C_p \zeta^p\right) d\zeta. \quad (13)$$

From eqn. (13), we further infer that appropriate Gauss Legendre quadrature rules have to be applied in variates ξ, η, and ζ to obtain the desired accuracy.

Case 1: Product Formulas with Odd Degree of Precision

Let us now first assume that the degree of precision $d = p + q + r = 2n - 1$, $n = 1, 2, 3 \cdots$; then we find from eqn. (13) that l takes values from 0 to $2n + 1$ and m takes values from 0 to $2n$ and p takes values from 0 to $2n - 1$. This suggests that we have to choose $(n + 1) \times (n + 1) \times n$-order Gauss Legendre quadrature rule in ξ, η, and ζ directions, respectively. This is demonstrated for the monomial X^2Y over the arbitrary linear tetrahedron in XYZ-space. It is seen that in this case $3 \times 3 \times 2$-order Gauss Legendre quadrature is adequate.

Case 2: Product Formulas with Even Degree of Precision

Let us now first assume that the degree of precision $d = p + q + r = 2n - 2, n = 1, 2, 3 \cdots$; then we find from eqn. (13) that l takes values from 0 to $2n$ and m takes values from 0 to $2n - 1$, and p takes values from 0 to $2n - 2$. This suggests that we have to choose Gauss Legendre quadrature rule of order $(n + 1) \times n \times n$ in variates ξ, η, and ζ, respectively. This is demonstrated for the monomials X^2Y^2 and X^4Y^4 over the arbitrary linear tetrahedron in XYZ-space. It is seen that in this case $4 \times 3 \times 3$ and $6 \times 5 \times 5$-order Gauss Legendre quadrature rules are adequate to integrate the monomials X^2Y^2 and X^4Y^4, respectively.

Thus we conclude that in both the cases, a precision $d = 2n - 1$ or $d = 2n - 2$ can be obtained by Gauss Legendre quadrature rule of order $(n + 1) \times (n + 1) \times (n + 1)$. We confirm this for the above examples, that is for X^2Y, X^2Y^2 and X^4Y^4 over the arbitrary linear tetrahedron in XYZ-space and it is seen that $3 \times 3 \times 3$, $4 \times 4 \times 4$, and $6 \times 6 \times 6$-order Gauss Legendre quadrature rules are highly suitable.

We note from the above discussion on the product formulas for odd and even degree that we require $(n + 1) \times (n + 1) \times (n)^{\text{th}}$ order or $(n + 1) \times (n) \times (n)^{\text{th}}$ order Gauss Legendre quadrature rules. But in either case one can always obtain the desired accuracy by using $(n + 1) \times (n + 1) \times (n + 1)^{\text{th}}$ order rule. We hope that this serves the general purpose.

The C-program for generating (x_m, y_m, z_m) and (c_m) is given in Appendix A.

3. SOME NUMERICAL RESULTS

We consider some typical integrals with known exact values.

Example 1. Let us consider the following multiple integrals, which are generalized to three-dimensions from Reddy and Shippy [16]:

$$I_1 = \int_0^1 \int_0^{1-x} \int_0^{1-x-y} \sqrt{(x + y + z)}\,dzdydx = 0.142857142857143$$

$$I_2 = \int_0^1 \int_0^{1-x} \int_0^{1-x-y} \frac{dzdydx}{\sqrt{(x + y + z)}} = 0.200000000000000$$

$$I_3 = \int_0^1 \int_0^{1-x} \int_0^{1-x-y} [(1 - x - y)^2 + z^2]^{-1/2} dzdydx = 0.440686793509772.$$

Example 2. We now consider the following multiple integrals from Rathod and Govinda Rao [19, 20]:

$$III_v^{\alpha,\beta,\gamma} = \int\int\int_v X^\alpha Y^\beta Z^\gamma dXdYdZ, \tag{14}$$

where v is the tetrahedron in (X, Y, Z) space with vertices spanning the points $\langle(5, 5, 0), (10, 10, 0), (8, 7, 8), (10, 5, 0)\rangle$.

On using the following transformations:

$$X(x, y, z) = 10 - 5x - 2z, \qquad Y(x, y, z) = 5 + 5y + 2z, \qquad Z(x, y, z) = 8z, \tag{15}$$

we obtain

$$III_v^{\alpha,\beta,\gamma} = \int\int\int_v X^\alpha Y^\beta Z^\gamma dXdYdZ = 200 \int_0^1 \int_0^{1-x} \int_0^{1-x-y} (10 - 5x - 2z)^\alpha \times (5 + 5y + 2z)^\beta \times (8z)^\gamma dzdydx. \tag{16}$$

We have evaluated the above integral for $\alpha = 2, \beta = 1, \gamma = 0$; $\alpha = 2, \beta = 2, \gamma = 0$, and $\alpha = 4, \beta = 4, \gamma = 0$. That is,

$$I_4 = III_v^{2,1,0} = \int\int\int_v X^2 YdXdYdZ = 15721.6666666667$$

$$I_5 = III_v^{2,2,0} = \int\int\int_v X^2 Y^2 dXdYdZ = 109662.063492063$$

$$I_6 = III_v^{4,4,0} = \int\int\int_v X^4 Y^4 dXdYdZ = 426917356.623377.$$

Again from Rathod and Govinda Rao [19, 20], we know that $I_4 = 47165/3$, other integrals were computed in a similar way.

Example 3. Application to Linear Three-Dimensional Elasticity: From the principle of virtual work, the stiffness matrix of an arbitrary element is given by

$$[K]_e = \int_V [B]^T [D][B]dV, \tag{17}$$

where $[D]$ is a material property matrix, $[B]$ is the strain displacement matrix, and the integration is performed over the volume element V in global co-ordinates.

The material property matrix $[D]$ for anisotropic materials following Hearmon [21] and Leknitskii [22] is written as

$$\begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} & D_{15} & D_{16} \\ & D_{22} & D_{23} & D_{24} & D_{25} & D_{26} \\ & & D_{33} & D_{34} & D_{35} & D_{36} \\ & & & D_{44} & D_{45} & D_{46} \\ & Symmetric & & & D_{55} & D_{56} \\ & & & & & D_{66} \end{bmatrix}.$$

The element stiffness matrix of eqn. (17) can be written as

$$[K]_e = t \int_V [B_1 \quad B_2 \quad B_3 \ldots B_{n_e}]^T [D][B_1 \quad B_2 \quad B_3 \ldots B_{n_e}] dV.$$

We can visualize the $[K]_e$ as consisting of $n_e \times n_e$ submatrices, $[K_{i,j}]_e$, signifying the stiffness relationship between nodes i and j, where n_e is the number of nodes in the element. Any such submatrix is given by

$$[K_{i,j}]_e = \int_V [B_i]^T [D][B_j] dV.$$

For a three-dimensional element, the matrix $[B_i]$ has the form

$$[B_i] = \begin{bmatrix} \dfrac{\partial N_i}{\partial X} & 0 & 0 \\ 0 & \dfrac{\partial N_i}{\partial Y} & 0 \\ 0 & 0 & \dfrac{\partial N_i}{\partial Z} \\ 0 & \dfrac{\partial N_i}{\partial Z} & \dfrac{\partial N_i}{\partial Y} \\ \dfrac{\partial N_i}{\partial Z} & 0 & \dfrac{\partial N_i}{\partial X} \\ 0 & \dfrac{\partial N_i}{\partial Y} & \dfrac{\partial N_i}{\partial X} \end{bmatrix},$$

in which N_i are the appropriate element shape functions.

The element stiffness matrix relating nodes i and j for an anisotropic material under three-dimensional loading is given by [23]:

$$[K_{i,j}]_e = \begin{bmatrix} K_{i,j}^{11} & K_{i,j}^{12} & K_{i,j}^{13} \\ K_{i,j}^{21} & K_{i,j}^{22} & K_{i,j}^{23} \\ K_{i,j}^{31} & K_{i,j}^{32} & K_{i,j}^{33} \end{bmatrix}, \tag{18}$$

where $K_{i,j}^{11}, K_{i,j}^{12}, \ldots, K_{i,j}^{33}$ are volume integrals over the domain V, and they are triple integrals of the product of global derivatives of shape functions.

Hence we need to evaluate

$$\int\int\int_V \frac{\partial N_i}{\partial t}\frac{\partial N_j}{\partial s}\, dXdYdZ, \qquad t, s = X, Y, Z. \tag{19}$$

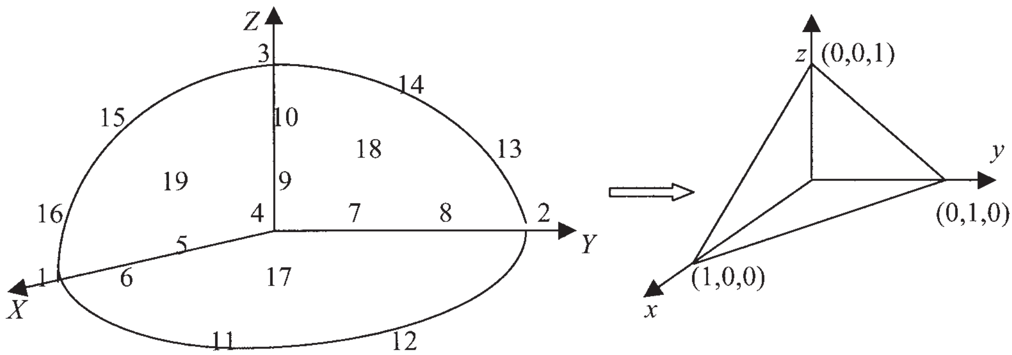

FIG. 1. Cubic tetrahedron T with one curved surface defined by the nodal coordinates $t_i = (x_i, y_i, z_i)$ and its mapping to an orthogonal tetrahedron $\hat{T}$.

We shall evaluate the integrals in eqn. (19), where V is a tetrahedral element T with three plane surfaces and one curved surface (see Fig. 1).

It can be shown that the evaluation of integrals in eqn. (19), now amounts to the evaluation of integrals of the type:

$$IIIpqr = \int\int\int_{\hat{T}} \frac{x^p y^q z^r}{J(x, y, z)}\, dxdydz = \int_0^1 \int_0^{1-x} \int_0^{1-x-y} \frac{x^p y^q z^r}{J(x, y, z)}\, dzdydx, \tag{20}$$

where J is the Jacobian of transformation and $\hat{T}$ is the standard tetrahedron:

$$\{(x, y, z) | 0 \leq x, y, z \leq 1, x + y + z \leq 1\}.$$

The transformation which maps a curved tetrahedron T into an orthogonal tetrahedron $\hat{T}$ is given by

$$t = \sum_{i=1}^{20} N_i(x, y, z) t_i, \qquad t = X, Y, Z \qquad \text{and} \qquad t_i = (X_i, Y_i, Z_i).$$

That is,

$$\begin{aligned} t(x, y, z) = t_4 &+ (t_1 - t_4)x + (t_2 - t_4)y + (t_3 - t_4)z + \tfrac{9}{4}(-t_1 - t_2 + t_{11} + t_{12})xy \\ &+ \tfrac{9}{4}(-t_2 - t_3 + t_{13} + t_{14})yz + \tfrac{9}{4}(-t_1 - t_3 + t_{15} + t_{16})xz + \tfrac{9}{4}[2(t_1 + t_2 + t_3) \\ &- 3(t_{11} + t_{12} + t_{13} + t_{14} + t_{15} + t_{16}) + 12t_{20}]xyz. \end{aligned} \tag{21}$$

For the above nodal data of the Table I:

$$X = x + R(xy + xz) + Sxyz$$

$$Y = y + R(xy + yz) + Sxyz \qquad \text{and}$$

TABLE I. Nodal co-ordinates (X_i, Y_i, Z_i), $i = 1(1)20$ of Example 3.

i	x_i	y_i	z_i
1	1	0	0
2	0	1	0
3	0	0	1
4	0	0	0
5	0.333333333333333	0	0
6	0.666666666666667	0	0
7	0	0.333333333333333	0
8	0	0.666666666666667	0
9	0	0	0.333333333333333
10	0	0	0.666666666666667
11	0.853850937602943	0.520517604269610	0
12	0.520517604269610	0.853850937602943	0
13	0	0.853850937602943	0.520517604269610
14	0	0.520517604269610	0.853850937602943
15	0.520517604269610	0	0.853850937602943
16	0.853850937602943	0	0.520517604269610
17	0.426925468801472	0.426925468801472	0
18	0	0.426925468801472	0.426925468801472
19	0.426925468801472	0	0.426925468801472
20	0.577350269189626	0.577350269189626	0.577350269189626

$$Z = z + R(yz + xz) + Sxyz$$

$$J(x, y, z) = 1 + 2R(x + y + z) + R^2(x^2 + y^2 + z^2) + (2R^2 + S)(xy + xz + yz) + RS(x^2y + xy^2 + x^2z + xz^2 + y^2z + yz^2) + 4R^3xyz + R^2S(x^2yz + xy^2z + xyz^2),$$

where $R = 0.842329219213245$, $S = 1.534481952840430$.

We have tabulated the computed values for I_1, I_2, and I_3 of Example 1, I_4, I_5, and I_6 of Example 2 and integrals of the form

$$IIIpqr = \int_0^1 \int_0^{1-x} \int_0^{1-x-y} \frac{x^p y^q z^r}{J(x, y, z)}\, dzdydx$$

for various values of p, q and r of Example 3 in Tables II–IV, respectively.

The C-program for evaluation of triple integrals in Examples 1, 2, and 3 is given in Appendix B.

4. CONCLUSIONS

In this article, we have derived various order: $n \times n \times n$ ($n = 2(1)10$) numerical integration rules based on classical Gauss Legendre quadrature rules. We have shown how these formulas can be applied to the arbitrary tetrahedral regions, since an affine transformation makes it possible to transform an arbitrary linear tetrahedron in 3-space (X, Y, Z) into an orthogonal tetrahedron T:$\{(x, y, z)|0 \leq x, y, z \leq 1, x + y + z \leq 1\}$ in the 3-space (x, y, z). The derivation

TABLE II. Numerical results for the integrals in Example 1.

Order of integration $(n \times n \times n)$	I_1	I_2	I_3
$(2 \times 2 \times 2)$	0.143127410953799	0.197660776240556	0.440894903222272
$(3 \times 3 \times 3)$	0.142875312759851	0.199583323221218	0.440665600968959
$(4 \times 4 \times 4)$	0.142860037924268	0.199881018522955	0.440687611536256
$(5 \times 5 \times 5)$	0.142857834882224	0.199956079429135	0.440686785349362
$(6 \times 6 \times 6)$	0.142857355360141	0.199980842555110	0.440686792457280
$(7 \times 7 \times 7)$	0.142857220464426	0.199990593978398	0.440686793586073
$(8 \times 8 \times 8)$	0.142857175072350	0.199994953918812	0.440686793507374
$(9 \times 9 \times 9)$	0.142857157618562	0.199997099779896	0.440686793509760
$(10 \times 10 \times 10)$	0.142857150174568	0.199998238575602	0.440686793509776
Exact Result	0.142857142857143	0.200000000000000	0.440686793509772

of the proposed formulas over the orthogonal tetrahedron T is made possible by transforming the tetrahedral region T into a standard 2-cube: $\{(\xi, \eta, \zeta)| -1 \leq \xi, \eta, \zeta \leq 1\}$ and over the 2-cube, the Gauss Legendre quadrature rules of all orders is applicable. It may be noted that a lot of mathematical effort is needed to derive numerical integration rules over the tetrahedral region T and the integration formulas available at this moment in the literature are confined to a precision of cubic order. By the proposed method this restriction is removed and one can now obtain numerical integration rules of very high degree of precision as the derivations proposed here rely on the standard Gauss Legendre quadrature rules. We have also suggested that when the sum of indices $p + q + r$ in the monomial $x^p y^q z^r$ is odd, say equal to $2n - 1$, then the lowest order rule to be chosen is $(n + 1) \times (n + 1) \times n$ and if $p + q + r$ is even, say equal to $2n - 2$, then the lowest order rule to be chosen is $(n + 1) \times n \times n$. The effectiveness of the derived formulas is further demonstrated by applying them to three nonpolynomial, three polynomial functions over the tetrahedral region in 3-space and to the evaluation of integrals for element stiffness matrices in linear three-dimensional elasticity.

TABLE III. Numerical results for the integrals in Example 2.

Order of integration $(m \times n \times p)$	I_4	I_5	I_6
$(2 \times 2 \times 2)$	15550.9773662551	107484.179240969	387905448.629903
$(3 \times 3 \times 2)$	15721.6666666667	109657.069444444	425750414.081290
$(3 \times 3 \times 3)$	15721.6666666667	109657.491666667	425756672.276488
$(4 \times 3 \times 3)$	15721.6666666667	109662.063492064	426925550.514920
$(4 \times 4 \times 4)$	15721.6666666667	109662.063492063	426910389.218001
$(5 \times 5 \times 5)$	15721.6666666666	109662.063492063	426917352.277076
$(6 \times 5 \times 5)$	15721.6666666666	109662.063492063	426917355.776530
$(6 \times 6 \times 6)$	15721.6666666667	109662.063492063	426917356.623377
$(7 \times 7 \times 7)$	15721.6666666667	109662.063492063	426917356.623377
$(8 \times 8 \times 8)$	15721.6666666667	109662.063492063	426917356.623377
$(9 \times 9 \times 9)$	15721.6666666667	109662.063492063	426917356.623377
$(10 \times 10 \times 10)$	15721.6666666666	109662.063492063	426917356.623377
Exact Result	15721.6666666667	109662.063492063	426917356.623377

TABLE IV. Numerical results for the integrals of the form III_{pqr} in Example 3.

Order of integration ($n \times n \times n$)	*III* 000	*III* 100	*III* 111
$n = 2$	0.056893206226020	0.019567972084673	0.001932171275894
$n = 3$	0.057883923688987	0.018329028115590	0.001545565326832
$n = 4$	0.057917372133509	0.018282559123781	0.001556783748912
$n = 5$	0.057918451760398	0.018281041134523	0.001557150589252
$n = 6$	0.057918486956290	0.018280992451779	0.001557150027720
$n = 7$	0.057918488134594	0.018280990869225	0.001557149986083
$n = 8$	0.057918488175332	0.018280990816424	0.001557149985090
$n = 9$	0.057918488176787	0.018280990814607	0.001557149985072
$n = 10$	0.057918488176841	0.018280990814543	0.001557149985072
Order of integration ($n \times n \times n$)	*III* 210	*III* 120	*III* 211
$n = 2$	0.002280977118494	0.002935667034621	0.000683625093897
$n = 3$	0.002449062446835	0.002492660001219	0.000632893797694
$n = 4$	0.002506905678547	0.002506743162664	0.000675465186414
$n = 5$	0.002507118923834	0.002507107864151	0.000674453537415
$n = 6$	0.002507107129756	0.002507106878406	0.000674419239922
$n = 7$	0.002507106822546	0.002507106818835	0.000674419314707
$n = 8$	0.002507106817092	0.002507106817119	0.000674419320804
$n = 9$	0.002507106817066	0.002507106817072	0.000674419320970
$n = 10$	0.002507106817071	0.002507106817071	0.000674419320974
Order of integration ($n \times n \times n$)	*III* 121	*III* 112	*III* 221
$n = 2$	0.000747255117782	0.000950298117764	0.000265427588870
$n = 3$	0.000668334787305	0.000665491425342	0.000248770797543
$n = 4$	0.000673911629201	0.000673906341778	0.000278273277502
$n = 5$	0.000674438415111	0.000674438335667	0.000277145808604
$n = 6$	0.000674419404613	0.000674419402337	0.000277101001072
$n = 7$	0.000674419320460	0.000674419320388	0.000277102666386
$n = 8$	0.000674419320957	0.000674419320955	0.000277102677707
$n = 9$	0.000674419320974	0.000674419320973	0.000277102677621
$n = 10$	0.000674419320974	0.000674419320974	0.000277102677618
Order of integration ($n \times n \times n$)	*III* 122	*III* 212	*III* 222
$n = 2$	0.000317946210299	0.000337179314418	0.000113202722729
$n = 3$	0.000277247192179	0.000246800679272	0.000093151415277
$n = 4$	0.000276223059541	0.000278299060667	0.000108974517096
$n = 5$	0.000277147560908	0.000277145833478	0.000108556171202
$n = 6$	0.000277102315030	0.000277101001224	0.000108479334138
$n = 7$	0.000277102667895	0.000277102666389	0.000108483396209
$n = 8$	0.000277102677612	0.000277102677707	0.000108483374276
$n = 9$	0.000277102677618	0.000277102677621	0.000108483373159
$n = 10$	0.000277102677618	0.000277102677618	0.000108483373158

TABLE IV. (Continued)

Order of integration ($n \times n \times n$)	*III* 333	*III* 444	*III* 555
$n = 2$	0.000007778591566	0.000000552335150	0.000000039523594
$n = 3$	0.000008025783249	0.000000828490405	0.000000094386236
$n = 4$	0.000011177364873	0.000001361820624	0.000000180717334
$n = 5$	0.000010628962559	0.000001220861950	0.000000154148399
$n = 6$	0.000010581796273	0.000001218909820	0.000000155227980
$n = 7$	0.000010595556709	0.000001228317566	0.000000158248961
$n = 8$	0.000010594735134	0.000001226843341	0.000000157286986
$n = 9$	0.000010594741935	0.000001226910060	0.000000157385336
$n = 10$	0.000010594742820	0.000001226912833	0.000000157386610
Order of integration ($n \times n \times n$)	*III* 666	*III* 777	*III* 888
$n = 2$	0.000000002837192	0.000000000204153	0.000000000014724
$n = 3$	0.000000011384017	0.000000001421674	0.000000000181514
$n = 4$	0.000000025130475	0.000000003592354	0.000000000522274
$n = 5$	0.000000020754760	0.000000002927201	0.000000000427270
$n = 6$	0.000000021139420	0.000000003018783	0.000000000446422
$n = 7$	0.000000021914165	0.000000003198955	0.000000000486168
$n = 8$	0.000000021549517	0.000000003090776	0.000000000458104
$n = 9$	0.000000021615030	0.000000003118096	0.000000000466993
$n = 10$	0.000000021613296	0.000000003116020	0.000000000465889

APPENDIX A
C-program for Generating Sampling Points (x_m, y_m, z_m) and Weight Coefficients (c_m)

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
main()
{
int i,j,k,n,m;
double xm,ym,zm,cm,A[20],W[20];
clrscr();
printf("Enter the value of n= ");
scanf("%d",&n);
printf("Enter the values of A's in order");
for(i=1;i<=n;i++)
scanf("%lf",&A[i]);
printf("Enter the values of W's in order");
for(i=1;i<=n;i++)
scanf("%lf",&W[i]);
for(i=1;i<=n;i++)
```

```
{for(j=1;j<=n;j++)
{for(k=1;k<=n;k++)
{
xm=(1.0+A[i])/2.0; ym=(1.0-A[i])*(1.0+A[j])/4.0;
zm=(1.0-A[i])*(1.0-A[j])*(1.0+A[k])/8.0;
cm=(1.0-A[i])*(1.0-A[i])*(1.0-A[j])*W[i]*W[j]*W[k]/64.0;
printf("%0.15lf  %0.15lf  %0.15lf  %0.15lf\n",xm,ym,zm,cm);
}}}
getch();
}
```

The sample output for n=2,3,4 and 5 obtained from the above program is given below:

x_m	y_m	z_m	c_m
	n=2		
0.211324865405187	0.166666666666667	0.131445855765802	0.061320326520293
0.211324865405187	0.166666666666667	0.490562612162344	0.061320326520293
0.211324865405187	0.622008467928146	0.035220810900864	0.016430731970725
0.211324865405187	0.622008467928146	0.131445855765802	0.016430731970725
0.788675134594813	0.044658198738520	0.035220810900864	0.004402601362608
0.788675134594813	0.044658198738520	0.131445855765802	0.004402601362608
0.788675134594813	0.166666666666667	0.009437387837656	0.001179673479707
0.788675134594813	0.166666666666667	0.035220810900864	0.001179673479707
	n=3		
0.112701665379259	0.100000000000000	0.088729833462074	0.014972747367084
0.112701665379259	0.100000000000000	0.698568501158667	0.014972747367084
0.112701665379259	0.100000000000000	0.393649167310371	0.023956395787334
0.112701665379259	0.443649167310371	0.050000000000000	0.013499628508586
0.112701665379259	0.443649167310371	0.393649167310371	0.013499628508586
0.112701665379259	0.443649167310371	0.221824583655185	0.021599405613738
0.112701665379259	0.787298334620741	0.011270166537926	0.001901788268649
0.112701665379259	0.787298334620741	0.088729833462074	0.001901788268649
0.112701665379259	0.787298334620741	0.050000000000000	0.003042861229838
0.500000000000000	0.056350832689629	0.050000000000000	0.007607153074595
0.500000000000000	0.056350832689629	0.393649167310371	0.007607153074595
0.500000000000000	0.056350832689629	0.221824583655185	0.012171444919352
0.500000000000000	0.250000000000000	0.028175416344815	0.006858710562414
0.500000000000000	0.250000000000000	0.221824583655185	0.006858710562414
0.500000000000000	0.250000000000000	0.125000000000000	0.010973936899863
0.500000000000000	0.443649167310371	0.006350832689629	0.000966235128423
0.500000000000000	0.443649167310371	0.050000000000000	0.000966235128423
0.500000000000000	0.443649167310371	0.028175416344815	0.001545976205477
0.887298334620741	0.012701665379258	0.011270166537926	0.000241558782106
0.887298334620741	0.012701665379258	0.088729833462074	0.000241558782106
0.887298334620741	0.012701665379258	0.050000000000000	0.000386494051369
0.887298334620741	0.056350832689629	0.006350832689629	0.000217792616242
0.887298334620741	0.056350832689629	0.050000000000000	0.000217792616242

0.887298334620741	0.056350832689629	0.028175416344815	0.000348468185988
0.887298334620741	0.100000000000000	0.001431498841332	0.000030681988197
0.887298334620741	0.100000000000000	0.011270166537926	0.000030681988197
0.887298334620741	0.100000000000000	0.006350832689629	0.000049091181116

n=4

0.069431844202974	0.064611063213548	0.805832094644763	0.004239832934111
0.069431844202974	0.064611063213548	0.060124997938716	0.004239832934111
0.069431844202974	0.064611063213548	0.580183044309859	0.007948679008092
0.069431844202974	0.064611063213548	0.285774048273620	0.007948679008092
0.069431844202974	0.307096311531159	0.580183044309859	0.005722890433136
0.069431844202974	0.307096311531159	0.043288799956008	0.005722890433136
0.069431844202974	0.307096311531159	0.417720226262576	0.010729059318706
0.069431844202974	0.307096311531159	0.205751618003291	0.010729059318706
0.069431844202974	0.623471844265867	0.285774048273620	0.002818857915521
0.069431844202974	0.623471844265867	0.021322263257539	0.002818857915521
0.069431844202974	0.623471844265867	0.205751618003291	0.005284688592238
0.069431844202974	0.623471844265867	0.101344693527868	0.005284688592238
0.069431844202974	0.865957092583479	0.060124997938716	0.000316343749669
0.069431844202974	0.865957092583479	0.004486065274832	0.000316343749669
0.069431844202974	0.865957092583479	0.043288799956008	0.000593069340565
0.069431844202974	0.865957092583479	0.021322263257539	0.000593069340565
0.330009478207572	0.046518677526561	0.580183044309859	0.004120367029080
0.330009478207572	0.046518677526561	0.043288799956008	0.004120367029080
0.330009478207572	0.046518677526561	0.417720226262576	0.007724708831375
0.330009478207572	0.046518677526561	0.205751618003291	0.007724708831375
0.330009478207572	0.221103222500738	0.417720226262576	0.005561636370626
0.330009478207572	0.221103222500738	0.031167073029114	0.005561636370626
0.330009478207572	0.221103222500738	0.300750235878433	0.010426746279122
0.330009478207572	0.221103222500738	0.148137063413257	0.010426746279122
0.330009478207572	0.448887299291690	0.205751618003291	0.002739430867978
0.330009478207572	0.448887299291690	0.015351604497447	0.002739430867978
0.330009478207572	0.448887299291690	0.148137063413257	0.005135781756688
0.330009478207572	0.448887299291690	0.072966159087481	0.005135781756688
0.330009478207572	0.623471844265867	0.043288799956008	0.000307430121953
0.330009478207572	0.623471844265867	0.003229877570553	0.000307430121953
0.330009478207572	0.623471844265867	0.031167073029114	0.000576358407229
0.330009478207572	0.623471844265867	0.015351604497447	0.000576358407229
0.669990521792428	0.022913166676413	0.285774048273620	0.000999657923009
0.669990521792428	0.022913166676413	0.021322263257539	0.000999657923009
0.669990521792428	0.022913166676413	0.205751618003291	0.001874121002261
0.669990521792428	0.022913166676413	0.101344693527868	0.001874121002261
0.669990521792428	0.108906255706834	0.205751618003291	0.001349329762022
0.669990521792428	0.108906255706834	0.015351604497447	0.001349329762022
0.669990521792428	0.108906255706834	0.148137063413257	0.002529672588768
0.669990521792428	0.108906255706834	0.072966159087481	0.002529672588768
0.669990521792428	0.221103222500738	0.101344693527868	0.000664623746473
0.669990521792428	0.221103222500738	0.007561562178966	0.000664623746473
0.669990521792428	0.221103222500738	0.072966159087481	0.001246011553748
0.669990521792428	0.221103222500738	0.035940096619353	0.001246011553748
0.669990521792428	0.307096311531159	0.021322263257539	0.000074586791665

0.669990521792428	0.307096311531159	0.001590903418873	0.000074586791665
0.669990521792428	0.307096311531159	0.015351604497447	0.000139832506234
0.669990521792428	0.307096311531159	0.007561562178966	0.000139832506234
0.930568155797026	0.004820780989426	0.060124997938716	0.000023603139442
0.930568155797026	0.004820780989426	0.004486065274832	0.000023603139442
0.930568155797026	0.004820780989426	0.043288799956008	0.000044250276349
0.930568155797026	0.004820780989426	0.021322263257539	0.000044250276349
0.930568155797026	0.022913166676413	0.043288799956008	0.000031859316866
0.930568155797026	0.022913166676413	0.003229877570553	0.000031859316866
0.930568155797026	0.022913166676413	0.031167073029114	0.000059728646651
0.930568155797026	0.022913166676413	0.015351604497447	0.000059728646651
0.930568155797026	0.046518677526561	0.021322263257539	0.000015692575033
0.930568155797026	0.046518677526561	0.001590903418873	0.000015692575033
0.930568155797026	0.046518677526561	0.015351604497447	0.000029419848303
0.930568155797026	0.046518677526561	0.007561562178966	0.000029419848303
0.930568155797026	0.064611063213548	0.004486065274832	0.000001761084871
0.930568155797026	0.064611063213548	0.000334715714594	0.000001761084871
0.930568155797026	0.064611063213548	0.003229877570553	0.000003301615550
0.930568155797026	0.064611063213548	0.001590903418873	0.000003301615550

n=5

0.046910077030668	0.044709521703645	0.454190200632844	0.003455952678910
0.046910077030668	0.219940124839679	0.698757684624372	0.002346734777065
0.046910077030668	0.219940124839679	0.034392113505281	0.002346734777065
0.046910077030668	0.219940124839679	0.563964232066324	0.004740764418080
0.046910077030668	0.219940124839679	0.169185566063329	0.004740764418080
0.046910077030668	0.219940124839679	0.366574899064827	0.005634781968811
0.046910077030668	0.476544961484666	0.454190200632844	0.001813025505580
0.046910077030668	0.476544961484666	0.022354760851822	0.001813025505580
0.046910077030668	0.476544961484666	0.366574899064827	0.003662589777903
0.046910077030668	0.476544961484666	0.109970062419839	0.003662589777903
0.046910077030668	0.476544961484666	0.238272480742333	0.004353284200533
0.046910077030668	0.733149798129653	0.209622716641315	0.000704005022098
0.046910077030668	0.733149798129653	0.010317408198364	0.000704005022098
0.046910077030668	0.733149798129653	0.169185566063329	0.001422198192795
0.046910077030668	0.733149798129653	0.050754558776350	0.001422198192795
0.046910077030668	0.733149798129653	0.109970062419839	0.001690397586996
0.046910077030668	0.908380401265687	0.042612194596523	0.000070841369555
0.046910077030668	0.908380401265687	0.002097327107122	0.000070841369555
0.046910077030668	0.908380401265687	0.034392113505281	0.000143110438979
0.046910077030668	0.908380401265687	0.010317408198364	0.000143110438979
0.046910077030668	0.908380401265687	0.022354760851822	0.000170098332251
0.230765344947159	0.036084856923188	0.698757684624372	0.001894039243550
0.230765344947159	0.036084856923188	0.034392113505281	0.001894039243550
0.230765344947159	0.036084856923188	0.563964232066324	0.003826249962299
0.230765344947159	0.036084856923188	0.169185566063329	0.003826249962299
0.230765344947159	0.036084856923188	0.366574899064827	0.004547807567383
0.230765344947159	0.177512700518577	0.563964232066324	0.003088149395941
0.230765344947159	0.177512700518577	0.027757722467940	0.003088149395941
0.230765344947159	0.177512700518577	0.455173033583358	0.006238535737859
0.230765344947159	0.177512700518577	0.136548920950906	0.006238535737859

0.230765344947159	0.177512700518577	0.295860977267132	0.007415004329977
0.230765344947159	0.384617327526421	0.366574899064827	0.002385822920682
0.230765344947159	0.384617327526421	0.018042428461594	0.002385822920682
0.230765344947159	0.384617327526421	0.295860977267132	0.004819728467296
0.230765344947159	0.384617327526421	0.088756350259289	0.004819728467296
0.230765344947159	0.384617327526421	0.192308663763210	0.005728637128330
0.230765344947159	0.591721954534264	0.169185566063329	0.000926424538886
0.230765344947159	0.591721954534264	0.008327134455248	0.000926424538886
0.230765344947159	0.591721954534264	0.136548920950906	0.001871519752855
0.230765344947159	0.591721954534264	0.040963779567671	0.001871519752855
0.230765344947159	0.591721954534264	0.088756350259289	0.002224452604614
0.230765344947159	0.733149798129653	0.034392113505281	0.000093222606464
0.230765344947159	0.733149798129653	0.001692743417907	0.000093222606464
0.230765344947159	0.733149798129653	0.027757722467940	0.000188323972528
0.230765344947159	0.733149798129653	0.008327134455248	0.000188323972528
0.230765344947159	0.733149798129653	0.018042428461594	0.000223838273981
0.500000000000000	0.023455038515334	0.454190200632844	0.000951130350813
0.500000000000000	0.023455038515334	0.022354760851822	0.000951130350813
0.500000000000000	0.023455038515334	0.366574899064827	0.001921429284706
0.500000000000000	0.023455038515334	0.109970062419839	0.001921429284706
0.500000000000000	0.023455038515334	0.238272480742333	0.002283774120164
0.500000000000000	0.115382672473579	0.366574899064827	0.001550777064587
0.500000000000000	0.115382672473579	0.018042428461594	0.001550777064587
0.500000000000000	0.115382672473579	0.295860977267132	0.003132807678149
0.500000000000000	0.115382672473579	0.088756350259289	0.003132807678149
0.500000000000000	0.115382672473579	0.192308663763210	0.003723595323417
0.500000000000000	0.250000000000000	0.238272480742333	0.001198089532334
0.500000000000000	0.250000000000000	0.011727519257667	0.001198089532334
0.500000000000000	0.250000000000000	0.192308663763210	0.002420324733784
0.500000000000000	0.250000000000000	0.057691336236790	0.002420324733784
0.500000000000000	0.250000000000000	0.125000000000000	0.002876751714678
0.500000000000000	0.384617327526421	0.109970062419839	0.000465222935413
0.500000000000000	0.384617327526421	0.005412610053740	0.000465222935413
0.500000000000000	0.384617327526421	0.088756350259289	0.000939821730277
0.500000000000000	0.384617327526421	0.026626322214291	0.000939821730277
0.500000000000000	0.384617327526421	0.057691336236790	0.001117054144152
0.500000000000000	0.476544961484666	0.022354760851822	0.000046813628964
0.500000000000000	0.476544961484666	0.001100277663512	0.000046813628964
0.500000000000000	0.476544961484666	0.018042428461594	0.000094570715294
0.500000000000000	0.476544961484666	0.005412610053740	0.000094570715294
0.500000000000000	0.476544961484666	0.011727519257667	0.000112404944503
0.769234655052841	0.010825220107480	0.209622716641315	0.000170456069101
0.769234655052841	0.010825220107480	0.010317408198364	0.000170456069101
0.769234655052841	0.010825220107480	0.169185566063329	0.000344347420568
0.769234655052841	0.010825220107480	0.050754558776350	0.000344347420568
0.769234655052841	0.010825220107480	0.109970062419839	0.000409284761972
0.769234655052841	0.053252644428581	0.169185566063329	0.000277921277830
0.769234655052841	0.053252644428581	0.008327134455248	0.000277921277830
0.769234655052841	0.053252644428581	0.136548920950906	0.000561443635575
0.769234655052841	0.053252644428581	0.040963779567671	0.000561443635575
0.769234655052841	0.053252644428581	0.088756350259289	0.000667321173390

0.769234655052841	0.115382672473579	0.109970062419839	0.000214714662336
0.769234655052841	0.115382672473579	0.005412610053740	0.000214714662336
0.769234655052841	0.115382672473579	0.088756350259289	0.000433756571553
0.769234655052841	0.115382672473579	0.026626322214291	0.000433756571553
0.769234655052841	0.115382672473579	0.057691336236790	0.000515554769800
0.769234655052841	0.177512700518577	0.050754558776350	0.000083374558238
0.769234655052841	0.177512700518577	0.002498085652232	0.000083374558238
0.769234655052841	0.177512700518577	0.040963779567671	0.000168429403669
0.769234655052841	0.177512700518577	0.012288864860910	0.000168429403669
0.769234655052841	0.177512700518577	0.026626322214291	0.000200191969715
0.769234655052841	0.219940124839679	0.010317408198364	0.000008389667270
0.769234655052841	0.219940124839679	0.000507811909116	0.000008389667270
0.769234655052841	0.219940124839679	0.008327134455248	0.000016948415501
0.769234655052841	0.219940124839679	0.002498085652232	0.000016948415501
0.769234655052841	0.219940124839679	0.005412610053740	0.000020144562700
0.953089922969332	0.002200555327023	0.042612194596523	0.000003486737214
0.953089922969332	0.002200555327023	0.002097327107122	0.000003486737214
0.953089922969332	0.002200555327023	0.034392113505281	0.000007043744304
0.953089922969332	0.002200555327023	0.010317408198364	0.000007043744304
0.953089922969332	0.002200555327023	0.022354760851822	0.000008372059841
0.953089922969332	0.010825220107480	0.034392113505281	0.000005684974827
0.953089922969332	0.010825220107480	0.001692743417907	0.000005684974827
0.953089922969332	0.010825220107480	0.027757722467940	0.000011484521661
0.953089922969332	0.010825220107480	0.008327134455248	0.000011484521661
0.953089922969332	0.010825220107480	0.018042428461594	0.000013650282922
0.953089922969332	0.023455038515334	0.022354760851822	0.000004392061882
0.953089922969332	0.023455038515334	0.001100277663512	0.000004392061882
0.953089922969332	0.023455038515334	0.018042428461594	0.000008872639079
0.953089922969332	0.023455038515334	0.005412610053740	0.000008872639079
0.953089922969332	0.023455038515334	0.011727519257667	0.000010545849211
0.953089922969332	0.036084856923188	0.010317408198364	0.000001705455115
0.953089922969332	0.036084856923188	0.000507811909116	0.000001705455115
0.953089922969332	0.036084856923188	0.008327134455248	0.000003445281079
0.953089922969332	0.036084856923188	0.002498085652232	0.000003445281079
0.953089922969332	0.036084856923188	0.005412610053740	0.000004094995235
0.953089922969332	0.044709521703645	0.002097327107122	0.000000171613514
0.953089922969332	0.044709521703645	0.000103228219901	0.000000171613514
0.953089922969332	0.044709521703645	0.001692743417907	0.000000346685638
0.953089922969332	0.044709521703645	0.000507811909116	0.000000346685638
0.953089922969332	0.044709521703645	0.001100277663512	0.000000412063922

APPENDIX B
C-program for the Evaluation of Triple Integrals in Examples 1, 2, and 3

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
main()
{
```

```
int i,j,k,n;
double x,y,z,c,A[20],W[20],I1,I2,I3,I4,I5,I6,III000,III100,III010,III001,III111,III210,
    III120,III211,III121,III112,III221,III122,III212,III222,III333,III444,III555,III666,
    III777,III888,S1=0.0,S2=0.0,S3=0.0,S4=0.0,S5=0.0,S6=0.0,S7=0.0, S8=0.0,S9=0.0,
    S10=0.0, S11=0.0, S12=0.0,S13=0.0,S14=0.0,S15=0.0,S16=0.0, S17=0.0,S18=0.0,
    S19=0.0,S20=0.0, S21=0.0,S22=0.0,S23=0.0, S24=0.0,S25=0.0,S26=0.0;
clrscr();
printf("Enter the value of n= ");
scanf("%d",&n);
printf("Enter the values of A's in order");
for(i=1;i<=n;i++)
scanf("%lf",&A[i]);
printf("Enter the values of W's in order");
for(i=1;i<=n;i++)
scanf("%lf",&W[i]);
for(i=1;i<=n;i++)
{for(j=1;j<=n;j++)
{for(k=1;k<=n;k++) {
x=(1.0+A[i])/2.0;  y=(1.0-A[i])*(1.0+A[j])/4.0;
z=(1.0-A[i])*(1.0-A[j])*(1.0+A[k])/8.0;
c=(1.0-A[i])*(1.0-A[i])*(1.0-A[j])*W[i]*W[j]*W[k]/64.0;
I1=c*sqrt(x+y+z);
S1=S1+I1;
I2=c*1.0/sqrt(x+y+z);
S2=S2+I2;
I3=c*1.0/sqrt((1.0-x-y)*(1.0-x-y)+z*z);
S3=S3+I3;
I4=200.0*c*(pow(10.0-5.0*x-2.0*z,2)*(5.0+5.0*y+2.0*z));
S4=S4+I4;
I5=200.0*c*(pow(10.0-5.0*x-2.0*z,2)*pow(5.0+5.0*y+2.0*z,2));
S5=S5+I5;
I6=200.0*c*(pow(10.0-5.0*x-2.0*z,4)*pow(5.0+5.0*y+2.0*z,4));
S6=S6+I6;
III000=c*(1.0/(1+1.684658438426490*(x+y+z)+.7095185135403395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z
+x*z*z+y*y*z+y*z*z) +2.390592702111290*x*y*z+1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S7=S7+III000;
III100=c*((x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z)/ (1+1.684658438426490*(x+y+z)+.7095185135403395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z) +1.292538985232890*(x*x*y+x*y*y
+x*x*z+x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z +1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S8=S8+III100;
III111=c*((x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043*x*
  y*z)*(y+.842329219213245*x*y+.842329219213245*y*z+1.53448195284043*x*y*z)
```

```
*(z+.842329219213245*x*z+.842329219213245*y*z+1.534481952840430*x*y*z)/
(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z) +2.953518979922
*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+x*z*z+y*y*z+y*z*z)
+2.390592702111290*x*y*z+1.088743354233900*(x*x*y*z+x*y*y*z+x*y*z*z)));
S9=S9+III111;
III210=c*(pow(x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z,2)*(y+.842329219213245*x*y+.842329219213245*y*z+1.53448195284043
*x*y*z)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220 *(x*y+x*z+y*z) +1.292538985232890*(x*x*y+x*y*y+x*x*z
+x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z+1.088743354233900*(x*x*y*z+
x*y*y*z+x*y*z*z)));
S10=S10+III210;
III120=c*((x+.842329219213245*x*y+.842329219213245*x*z+1.534481952840430*x
*y*z)*pow(y+.842329219213245*x*y+.842329219213245*y*z+1.534481952840430
*x*y*z,2)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z
+x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z+1.088743354233900*(x*x*y*z
+x*y*y*z+x*y*z*z)));
S11=S11+III120;
III211=c*(pow(x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z, 2) *(y+.842329219213245*x*y+.842329219213245*y*z+1.534481952840430
*x*y*z)*(z+.842329219213245*y*z+.842329219213245*x*z+1.534481952840430
*x*y*z)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.9535189799 21220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+
x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z+1.088743354233900*(x*x*y*z
+x*y*y*z+x*y*z*z)));
S12=S12+III211;
III121=c*((x+.842329219213245*x*y+.842329219213245*x*z+1.534481952840430*x
*y*z)*pow(y+.842329219213245*x*y+.842329219213245*y*z+1.534481952840430
*x*y*z,2)*(z+.842329219213245*x*z+.842329219213245*y*z+1.534481952840430
*x*y*z)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+
x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z +1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S13=S13+III121;
III112=c*((x+.842329219213245*x*y+.842329219213245*x*z+1.534481952840430*x
*y*z)*(y+.842329219213245*x*y+.842329219213245*y*z+1.534481952840430
*x*y*z)*pow(z+.842329219213245*x*z+.842329219213245*y*z+1.534481952840430
*x*y*z,2)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220 *(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z
+x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z+1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S14=S14+III112;
III221=c*(pow(x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z,2)*pow(y+.842329219213245*x*y+.842329219213245*y*z+1.53448195284043
*x*y*z,2)*(z+.842329219213245*x*z+.842329219213245*y*z+1.534481952840430
```

```
*x*y*z)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+
x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z+1.088743354233900*(x*x*y*z+
x*y*y*z+x*y*z*z)));
S15=S15+III221;
III122=c*((x+.842329219213245*x*y+.842329219213245*x*z+1.534481952840430*x
*y*z)*pow(y+.842329219213245*x*y+.842329219213245*y*z+1.534481952840430
*x*y*z,2)*pow(z+.842329219213245*x*z+.842329219213245*y*z+1.53448195284043
*x*y*z,2)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+
x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z+1.088743354233900*(x*x*y*z+
x*y*y*z+x*y*z*z)));
S16=S16+III122;
III212=c*(pow(x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z, 2)*(y+.842329219213245*x*y+.842329219213245*y*z+1.53448195284043
*x*y*z)*pow(z+.842329219213245*z*y+.842329219213245*x*z+1.53448195284043
*x*y*z,2)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+
x*z*z+y*y*z+y*z*z) +2.390592702111290*x*y*z +1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S17=S17+III212;
III222=c*(pow(x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z,2)*pow(y+.842329219213245*x*y+.842329219213245*y*z+1.53448195284043
*x*y*z,2)*pow(z+.842329219213245*x*z+.842329219213245*y*z+1.53448195284043
*x*y*z,2)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+
x*z*z+y*y*z+y*z*z) +2.390592702111290*x*y*z+1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S18=S18+III222;
III333=c*(pow(x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z,3)*pow(y+.842329219213245*x*y+.842329219213245*y*z+1.53448195284043
*x*y*z,3)*pow(z+.842329219213245*x*z+.842329219213245*y*z+1.53448195284043
*x*y*z,3)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+
x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z+1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S19=S19+III333;
III444=c*(pow(x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z,4)*pow(y+.842329219213245*x*y+.842329219213245*y*z+1.53448195284043
*x*y*z,4)*pow(z+.842329219213245*x*z+.842329219213245*y*z+1.53448195284043
*x*y*z,4)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+
x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z+1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S20=S20+III444;
```

```
III555=c*(pow(x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z,5)*pow(y+.842329219213245*x*y+.842329219213245*y*z+1.53448195284043
*x*y*z,5)*pow(z+.842329219213245*x*z+.842329219213245*y*z+1.53448195284043
*x*y*z,5)/ (1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+
x*x*z+x*z*z+y*y*z+y*z*z) +2.390592702111290*x*y*z+1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S21=S21+III555;
III666=c*(pow(x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z,6)*pow(y+.842329219213245*x*y+.842329219213245*y*z+1.53448195284043
*x*y*z,6)*pow(z+.842329219213245*x*z+.842329219213245*y*z+1.53448195284043
*x*y*z,6)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+
x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z+1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S22=S22+III666;
III777=c*(pow(x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z,7)*pow(y+.842329219213245*x*y+.842329219213245*y*z+1.53448195284043
*x*y*z,7)*pow(z+.842329219213245*x*z+.842329219213245*y*z+1.53448195284043
*x*y*z,7)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+
x*z*z+y*y*z+y*z*z)+2.390592702111290*x*y*z+1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S23=S23+III777;
III888=c*(pow(x+.842329219213245*x*y+.842329219213245*x*z+1.53448195284043
*x*y*z,8)*pow(y+.842329219213245*x*y+.842329219213245*y*z+1.53448195284043
*x*y*z,8)*pow(z+.842329219213245*x*z+.842329219213245*y*z+1.53448195284043
*x*y*z,8)/(1+1.684658438426490*(x+y+z)+.709518513540395*(x*x+y*y+z*z)
+2.953518979921220*(x*y+x*z+y*z)+1.292538985232890*(x*x*y+x*y*y+x*x*z+
x*z*z+y*y*z+y*z*z) +2.390592702111290*x*y*z +1.088743354233900
*(x*x*y*z+x*y*y*z+x*y*z*z)));
S24=S24+III888;     }}}
        printf("I1 = %0.15lf\n",S1);              printf("I2 = %0.15lf\n",S2);
        printf("I3 = %0.15lf\n",S3);              printf("I4 = %0.15lf\n",S4);
        printf("I5 = %0.15lf\n",S5);              printf("I6 = %0.15lf\n",S6);
        printf("III000 = %0.15lf\n",S7);          printf("III100 = %0.15lf\n",S8);
        printf("III111 = %0.15lf\n",S9);          printf("III210 = %0.15lf\n",S10);
        printf("III120 = %0.15lf\n",S11);         printf("III211 = %0.15lf\n",S12);
        printf("III121 = %0.15lf\n",S13);         printf("III112 = %0.15lf\n",S14);
        printf("III221 = %0.15lf\n",S15);         printf("III122 = %0.15lf\n",S16);
        printf("III212 = %0.15lf\n",S17);         printf("III222 = %0.15lf\n",S18);
        printf("III333 = %0.15lf\n",S19);         printf("III444 = %0.15lf\n",S20);
        printf("III555 = %0.15lf\n",S21);         printf("III666 = %0.15lf\n",S22);
        printf("III777 = %0.15lf\n",S23);         printf("III888 = %0.15lf\n",S24);
        getch();
        }
```

References

1. O. C. Zienkiewicz, The Finite Element Method, 3^{rd} Edn., London, Mc-Graw Hill, 1977.
2. T. J. R. Hughes, The finite element method, static and dynamic analysis, Prentice Hall, Englewood Cliffs, NJ, 1987.
3. K. J. Bathe, Finite element procedures, Prentice Hall, Englewood Cliffs, NJ, 1996.
4. P. C. Hammer, O. J. Marlowe, and A. H. Stroud, Numerical integration over simplexes and cones, Math Tables Other Aids to Computation 10 (1956), 130–136.
5. P. C. Hammer and A. H. Stroud, Numerical integration over simplexes, Math Tables Other Aids to Computation 10 (1956), 137–139.
6. P. C. Hammer and A. H. Stroud, Numerical evaluation of multiple integrals, Math Tables Other Aids to Computation 12 (1958), 272–280.
7. G. R. Cowper, Gaussian quadrature formulas for triangle, Int J Numer Meth Engng 7 (1973), 405–408.
8. J. N. Lyness and D. Jespersen, Moderate degree symmetric quadrature rules for the triangle, J Inst Math Appl 15 (1975), 9–32.
9. F. G. Lannoy, Triangular finite elements and numerical integration, Comput Struct 7 (1977), 613.
10. D. P. Laurie, Automatic numerical integration over a triangle, SIR Spec. Rep. WISK 273, National Research Institute of Mathematical Sciences, Pretoria, 1977.
11. M. E. Laursen and M. Gellert, Some criteria for numerically integrated matrices and quadrature formulas for triangles, Int J Numer Meth Engng 12 (1978), 67–76.
12. F. G. Lether, Computation of double integrals over a triangle, J Comp Appl Math 2 (1976), 219–224.
13. P. Hillion, Numerical integration on a triangle, Int J Numer Meth Engng 11 (1977), 797–815.
14. M. Abramowicz and I. A. Stegun, Handbook of mathematical functions, Dover Publications, New York, 1965.
15. C. T. Reddy, Improved three point integration schemes for triangular finite elements, Int J Numer Meth Engng 12 (1978), 1890–1896.
16. C. T. Reddy and D. J. Shippy, Alternative integration formulae for triangular finite elements, Int J Numer Meth Engng 17 (1981), 133–153.
17. G. Lague and R. Baldur, Extended numerical integration method for triangular surfaces, Int J Numer Meth Engng 11 (1977), 388–392.
18. H. T. Rathod, K. V. Nagaraja, B. Venkatesudu, and N. L. Ramesh, Gauss Legendre quadrature over a triangle, J Indian Inst Sci 84 (2004), 183–188.
19. H. T. Rathod and H. S. Govinda Rao, Integration of polynomials over an arbitrary tetrahedron in Euclidean three dimensional space, Comput Struct 59(1) (1996), 55–65.
20. H. T. Rathod and H. S. Govinda Rao, Integration of trivariate polynomials over linear polyhedra in euclidean three-dimensional space, J Austral Math Soc Ser B-39 (1998), 355–385.
21. R. F. S. Hearmon, An introduction to applied anisotropic elasticity, Oxford University Press, Oxford, 1961.
22. S. G. Leknitskii, Theory of elasticity of an anisotropic elastic body, Holden Day, San Francisco, 1963.
23. H. T. Rathod, Explicit stiffness matrices of the Lagrange rectangular prism elements for three-dimensional problems, Comput Struct 25(6) (1987), 895–907.