Available online at www.sciencedirect.com

ScienceDirect

Applied Mathematics and Computation 190 (2007) 21–39

APPLIED MATHEMATICS AND COMPUTATION

www.elsevier.com/locate/amc

On the application of two symmetric Gauss Legendre quadrature rules for composite numerical integration over a triangular surface

H.T. Rathod [a,*], K.V. Nagaraja [b], B. Venkatesudu [b]

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

[b] *Department of Mathematics, Amrita School of Engineering, # 26&27, Kasavanahalli, Carmelram post, Bangalore 560 035, India*

Abstract

This paper first presents a Gauss Legendre quadrature rule for the evaluation of $I = \int\int_T f(x,y)\,dx\,dy$, where $f(x,y)$ is an analytic function in x, y and T is the standard triangular surface: $\{(x,y)|0 \leqslant x,y \leqslant 1, x+y \leqslant 1\}$ in the two space (x,y). We transform this integral into an equivalent integral $\int\int_S f(x(\xi,\eta),y(\xi,\eta))\frac{\partial(x,y)}{\partial(\xi,\eta)}\,d\xi\,d\eta$ where S is the 2-square in (ξ,η) space: $\{(\xi,\eta)|-1 \leqslant \xi,\eta \leqslant 1\}$. We then apply the one-dimensional Gauss Legendre quadrature rules in ξ and η variables to arrive at an efficient Quadrature rules with new weight coefficients and new sampling points. Then a second Gauss Legendre quadrature rule of composite type is obtained. This rule is derived by discretising T into three new triangles T_i^C $(i = 1,2,3)$ of equal size which are obtained by joining centroid of T, $C = (1/3,1/3)$ to the three vertices of T. By use of affine transformations defined over each T_i^C and the linearity property of integrals leads to the result:

$$I = \sum_{i=1}^{3}\int\int_{T_i^C} f(x,y)\,dx\,dy = \frac{1}{3}\int\int_T G(X,Y)\,dX\,dY,$$

where $G(X,Y) = \sum_{i=1}^{n\times n} f(x_i^C(X,Y), y_i^C(X,Y))$ and $x = x_i^C(X,Y)$ and $y = y_i^C(X,Y)$ refer to affine transformations which map each T_i^C into T the standard triangular surface. We then write $\int\int_T G(X,Y)\,dX\,dY = \int\int_S G(X(\xi,\eta),Y(\xi,\eta))\frac{\partial(X,Y)}{\partial(\xi,\eta)}\,d\xi\,d\eta$ and a composite rule of integration is thus obtained. We next propose the discretisation of the standard triangular surface T into n^2 right isosceles triangular surfaces T_i $(i = 1(1)n^2)$ each of which has an area equal to $1/(2n^2)$ units. We have again shown that the use of affine transformation over each T_i and the use of linearity property of integrals lead to the result:

$$\int\int_T f(x,y)\,dx\,dy = \sum_{i=1}^{n\times n}\int\int_{T_i} f(x,y)\,dx\,dy = \frac{1}{n^2}\int\int_T H(X,Y)\,dX\,dY,$$

where $H(X,Y) = \sum_{i=1}^{n\times n} f(x_i(X,Y), y_i(X,Y))$ and $x = x_i(X,Y)$, $y = y_i(X,Y)$ refer to affine transformations which map each T_i in (x,y) space into T a standard triangular surface T in the (x,y) space. We can now apply the two rules earlier derived to the integral $\int\int_T H(X,Y)\,dX\,dY$, this amounts to application of composite numerical integration of T into n^2 and $3n^2$ triangles of equal sizes respectively. We can now apply the rules, which are derived earlier to the evaluation of the integral, $\int\int_T f(x,y)\,dx\,dy$ and each of these procedures converges to the exact value of the integral $\int\int_T f(x,y)\,dx\,dy$ for sufficiently

* Corresponding author.
E-mail address: nagarajaitec123@yahoo.com (K.V. Nagaraja).

0096-3003/$ - see front matter
doi:10.1016/j.amc.2007.01.003

large value of n and the convergence is much faster for higher order rules. We have demonstrated this aspect by applying the above composite integration method to some typical integrals.

Keywords: Quadrature rules; Weight function; Numerical integration; FEM; Triangle

1. Introduction

In recent years, the Finite Element Method (FEM) has become a very powerful tool for the approximate solution of boundary value problems governing the diverse physical phenomena. Its use in industry and research is extensive and without it many practical problems in science and engineering would be incapable of solution. The triangular elements with either straight sides or curved sides are very widely used in finite element analysis [1–3].The basic problem of integrating a function of two variables over the surface of the triangle were first given by Hammer et al. [4] and Hammer and Stroud [5,6]. With the advent of finite element method, the triangular elements are proved to be versatile and there has been considerable interest in the area of numerical integration schemes over triangles. Cowper [7] provided a table of Gaussian quadrature formulae for symmetrically placed integration points. Lyness and Jespersen [8] made an elaborate study of symmetric quadrature rules and provided integration formulas with a precision of upto degree eleven by formulating the problem in terms of polar coordinates. Lannoy [9] discussed the symmetric 4-point integration rule, which is presented in Ref. [7]. Laursen and Gellert [10] also gave some new higher order formulas of precision upto degree 10. Dunavant [11] presented some extensions to the integration formulas given by Lyness and Jespersen [8] and also gave tables of integration formulas with precisions of degree from 11 to 20. Laurie [12] derived a seven- point formula and discussed the numerical error in integrating some functions. Sylvester [13] derived some numerical integration formulas for triangles as product of one-dimensional Newton Cotes rules of closed type as well as open type. The precision of these integration formulas is again limited to degree ten at most for various reasons. Lether [14] and Hillion [15] derived formulas for triangles as product of one-dimensional Gauss Legendre and Gauss Jacobi quadrature rules. The precision of these formulas is again limited to a degree seven. We also note that higher order quadrature rules of this type cannot be derived beyond degree $15 = 2 \times 8 - 1$ as the abscissas and weights of 1-D Gauss Jacobi quadrature rules are not tabulated even in the standard reference work of Abramowicz and Stegun [16] for a order higher than eighth. Reddy [17] and Reddy and Shippy [18] derived some 3-point, 4-point, 6-point and 7-point formulas of precision 3, 4, 6 and 7, respectively which gave improved accuracy as compared to some earlier works. Since all the above information on integration formulas which is documented in the works [4–18] is limited to a precision of degree at most 20 and it is not likely that the techniques proposed by these authors can be extended much further to give greater accuracy which may be demanded in future we have taken a significant note from the recent work of Lague and Baldur [19] on the above aspect who gave substantial reasons in favour of the product formulas based only on roots and weights of Gauss Legendre quadrature rules. The use of proposed method on product formulas [19] will remove the restrictions on the derivation of high precision numerical integration formulas and it is clear that now one can obtain formulas of very high degree of precision as the methods rely on standard Gauss Legendre quadrature rules. However Lague and Baldur [19] have not worked out explicit weights and abscissas required for this purpose. Rathod et al. [20–23] provided this information in a systematic manner in their recent works, for the first time.

Integration formulas resulting from interval subdivision and repeated application of a low order formula are called composite numerical integration formulas [24–27]. One way to reduce the error associated with low order integration formula in one dimension is to subdivide the interval of integration, say, $[a, b]$ into smaller intervals and then to use the formula repeatedly on each subinterval. We adopt a strategy similar to the above, which is normally used for the treatment of line integrals over arbitrary shaped curves to evaluation of double integrals also. We segment the given region into subregions and effect a transformation over each subregion into a standard region. The success of this strategy follows from the linearity property of double integrals. Repeated application of low order formula is usually preferred to the single application of a high order formula partly because of the lower order formulas and partly because of the computational difficulties

one such difficulty is due to the errors introduced because of only a fixed usually small number of digits can be retained after each computer operation. In addition there exist many functions for which the magnitude of the derivative increases without bound as the order of differentiation increases. Therefore a higher order formula may produce a larger error than a lower order one. It is in view of this that the numerical integration formulas employing more than eight points (for Newton Cotes rules) are almost never used. We feel that these important details cannot be simply ignored, and they need to be addressed in great rigor. Hence the derivation of algorithms for composite numerical integration formulas over dimensions higher than one is important for practical applications and it should be used wherever necessary. It is the main purpose of this paper to evolve a practical and workable algorithm for composite numerical integration over triangular surfaces by using the well-known Gauss Legendre quadrature rules. We have demonstrated the effectiveness of the above algorithm by applying it to some typical integrals.

2. Formulation of integrals over a triangular area

The finite element method for two-dimensional problems with triangular elements requires the numerical integration of shape functions, product of shape function derivatives and rational functions whose denominators are bivariate polynomials, etc. Since an affine transformation makes it possible to transform any triangle into the two-dimensional standard triangle T with coordinates $(0,0)$, $(0,1)$, $(1,0)$ in Cartesian frame of (x,y) space (say), we have just to consider numerical integration on T. The integral of an arbitrary function, f, over the surface of a triangle T is given by

$$I = \int\int_T f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \int_0^1 \mathrm{d}x \int_0^{1-x} f(x,y)\mathrm{d}y = \int_0^1 \mathrm{d}y \int_0^{1-y} f(x,y)\,\mathrm{d}x. \tag{1}$$

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

$$I = \sum_{m=1}^{N} c_m f(x_m, y_m), \tag{2}$$

where c_m are the weights associated with sampling points (x_m, y_m) and N is the number total sampling points related to the required precision. One of these methods, which have an optimum precision upto a degree 20, is reported in recent work [11]. The other method is approximation of I by product formulas [14,15] which is of type (2) based on the roots and weights of Gauss Legendre and Gauss Jacobi quadrature rules. The reported precision of these formulas is limited to a degree seven. This is because the weights and roots of Gauss Jacobi quadrature rules are not tabulated even in the standard reference books of Abramowicz and Stegun [16] beyond an order of precision eight. Use of these will enable us to derive formulas of precision $2\times 8 - 1 = 15$ degree only. The product formulas proposed in this paper and in the recent work [19] are based on the sampling points and weight coefficients of Gauss Legendre Quadrature formulas, as this enables us to obtain formulas of very high degree of precision, as Gauss Legendre quadrature rules of order as large as 96 are well documented in Abramowicz and Stegun [16].

The integral I of Eq. (1) can be transformed into an integral over the surface of the square: $\{(u,v)|0 \leqslant u,v \leqslant 1,\}$ by the substitution (see Fig. 1):

$$x = uv, \quad y = u(1-v). \tag{3}$$

Then the determinant of the Jacobian and the differential area are:

$$|J| = \frac{\partial(x,y)}{\partial(u,v)} = \frac{\partial x}{\partial u}\frac{\partial y}{\partial v} - \frac{\partial x}{\partial v}\frac{\partial y}{\partial u} = (v)(-u) - u(1-v) = u \quad \text{and} \quad \mathrm{d}x\,\mathrm{d}y = \frac{\partial(x,y)}{\partial(u,v)}\mathrm{d}u\,\mathrm{d}v = u\,\mathrm{d}u\,\mathrm{d}v. \tag{4}$$

Then on using Eqs. (3) and (4) in Eq. (1), we have

$$I = \int_0^1 \int_0^{1-x} f(x,y)\,\mathrm{d}y\,\mathrm{d}x = \int_0^1 \int_0^1 f(uv, u(1-v))u\,\mathrm{d}u\,\mathrm{d}v. \tag{5}$$

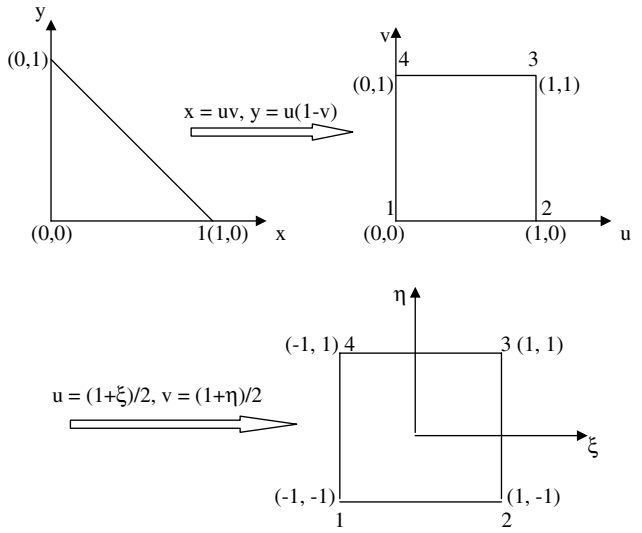

Fig. 1. Transformation of standard triangle T into equivalent 1-square in (u,v) space and 2-square in (ξ,η) space.

The integral I of Eq. (5) can be further transformed into an integral over the standard 2-square: $\{(\xi,\eta)|-1\leqslant\xi,\eta\leqslant 1\}$ by the substitution (see Fig. 1):

$$u=(1+\xi)/2,\quad v=(1+\eta)/2. \tag{6}$$

Then clearly the determinant of the Jacobian and the differential area are:

$$\begin{aligned}&\frac{\partial(u,v)}{\partial(\xi,\eta)}=\frac{\partial u}{\partial\xi}\frac{\partial v}{\partial\eta}-\frac{\partial u}{\partial\eta}\frac{\partial v}{\partial\xi}=(1/2)(1/2)-(0)(0)=1/4,\\&\mathrm{d}u\,\mathrm{d}v=\frac{\partial(u,v)}{\partial(\xi,\eta)}\mathrm{d}\xi\,\mathrm{d}\eta=\frac{1}{4}\mathrm{d}\xi\,\mathrm{d}\eta.\end{aligned} \tag{7}$$

Now on using Eqs. (6) and (7) in Eq. (5), we have:

$$\begin{aligned}I&=\int_0^1\int_0^{1-x}f(x,y)\mathrm{d}y\,\mathrm{d}x=\int_0^1\int_0^1 f(uv,u(1-v))u\,\mathrm{d}u\,\mathrm{d}v\\&=\int_{-1}^1\int_{-1}^1 f\left(\frac{(1+\xi)(1+\eta)}{4},\frac{(1+\xi)(1-\eta)}{4}\right)\left(\frac{1+\xi}{8}\right)\mathrm{d}\xi\,\mathrm{d}\eta.\end{aligned} \tag{8}$$

Eq. (8) represents an integral over the surface of a standard 2-square: $\{(\xi,\eta)|-1\leqslant\xi,\eta\leqslant 1\}$. Now efficient Gauss Legendre quadrature rules are readily available over the 2-square in the literature so that any desired accuracy can be readily obtained for the integral I of Eq. (1) [16].

From Eq. (8), we can write:

$$\begin{aligned}&I=\int_{-1}^1\int_{-1}^1 f(x(\xi,\eta),y(\xi,\eta))\left(\frac{1+\xi}{8}\right)\mathrm{d}\xi\,\mathrm{d}\eta,\\&I=\sum_{i=1}^s\sum_{j=1}^s\left(\frac{1+\xi_i}{8}\right)w_iw_jf(x(\xi_i,\eta_j),y(\xi_i,\eta_j)),\end{aligned} \tag{9}$$

where (ξ_i,η_j) are Gaussian points in the ξ, η directions and w_i and w_j are the corresponding weight coefficients.

We can rewrite Eq. (9) as:

$$I=\sum_{k=1}^{N=s\times s}c_kf(x_k,y_k), \tag{10}$$

where, c_k, x_k and y_k can be obtained from the relations:

$$c_k = \frac{(1+\xi_i)}{8} w_i w_j, \quad x_k = \frac{(1+\xi_i)(1+\eta_j)}{4}, \quad y_k = \frac{(1+\xi_i)(1-\eta_j)}{4}, \tag{11}$$
$$(k = 1, 2, \ldots\ldots, N), (i, j = 1, 2, 3, \ldots, s).$$

The weighting coefficients c_k and sampling points (x_k, y_k) of various order can be now easily computed by formulas of Eq. (10) and (11).We have listed here a C- Program which generates c_k, x_k and y_k and then computes the integral $\int\int_T f(x,y)\mathrm{d}x\,\mathrm{d}y$. We have also given the sample output of the program for $n = 2, 3, 4, 5$.

C-Program

```
# include⟨stdio.h⟩
# include⟨conio.h⟩
# include⟨math.h⟩
void main ()
{
double c[10][10], x[10][10], y[10][10], p[10], q[10], w1[10], w2[10];
int k, i,j, n;
clrscr ();
printf (“input n\ n”);
scanf (“%d”, &n);
printf (“enter % d p values\ n”,n);
for (i=0; i<n; + + i){
scanf (“% lf”,& x[i]);
}
printf (“enter % d q values\ n”,n);
for (i=0; i<n; + + i){
scanf (“% lf”,& y[i]);
}
printf (“enter % d w1 values\ n”,n);
for (i=0; i<n; + + i){
scanf (“% lf”,& w1[i]);
}
printf (“enter % d w1 values\ n”,n);
for (i=0; i<n; + + i){
scanf (“% lf”,& w2[i]);
}
for (i=0; i<n; + + i){
for (j=0; j<n; + + j)
{
c[i][j] = ((1- p[i])/8.0)*(w1 [i]*w2[j]);
x[i][j] = ((1+ p[i])*(1+ q[j]))/4.0;
y[i][j] = ((1 + p[i])*(1- q[j]))/4.0;
}
for (i=0; i<n; + + i){
for (j=0; j<n; + + j)
{
printf (“% 0.15lf \ t % 0.15 lf \ t % 0.15 lf \ n”, c[i][j], x[i][j], y[i][j]);
}}
getch ();
```

Sample output

k	c_k	x_k	y_k
$s = 2$			
1	0.052831216351297	0.044658198738520	0.166666666666667
2	0.052831216351297	0.166666666666667	0.044658198738520
3	0.197168783648703	0.166666666666667	0.622008467928146
4	0.197168783648703	0.622008467928146	0.166666666666667
$s = 3$			
1	0.008696116155807	0.012701665379258	0.100000000000000
2	0.013913785849291	0.056350832689629	0.056350832689629
3	0.008696116155807	0.100000000000000	0.012701665379258
4	0.061728395061728	0.056350832689629	0.443649167310371
5	0.098765432098765	0.250000000000000	0.250000000000000
6	0.061728395061728	0.443649167310371	0.056350832689629
7	0.068464377671354	0.100000000000000	0.787298334620741
8	0.109543004274166	0.443649167310371	0.443649167310371
9	0.068464377671354	0.787298334620741	0.100000000000000
$s = 4$			
1	0.002100365244475	0.004820780989426	0.064611063213548
2	0.003937685608733	0.022913166676413	0.046518677526561
3	0.003937685608733	0.046518677526561	0.022913166676413
4	0.002100365244475	0.064611063213548	0.004820780989426
5	0.018715815315013	0.022913166676413	0.307096311531159
6	0.035087705252933	0.108906255706834	0.221103222500738
7	0.035087705252933	0.221103222500738	0.108906255706834
8	0.018715815315013	0.307096311531159	0.022913166676413
9	0.037997147647950	0.046518677526561	0.623471844265867
10	0.071235620499740	0.221103222500738	0.448887299291690
11	0.071235620499740	0.448887299291690	0.221103222500738
12	0.037997147647950	0.623471844265867	0.046518677526561
13	0.028150383076926	0.064611063213548	0.865957092583479
14	0.052775277354230	0.307096311531159	0.623471844265867
15	0.052775277354230	0.623471844265867	0.307096311531159
16	0.028150383076926	0.865957092583479	0.064611063213548
$s = 5$			
1	0.000658316657301	0.002200555327023	0.044709521703645
2	0.001329900683819	0.010825220107480	0.036084856923188
3	0.001580694532071	0.023455038515334	0.023455038515334
4	0.001329900683819	0.036084856923188	0.010825220107480
5	0.000658316657301	0.044709521703645	0.002200555327023
6	0.006542197529252	0.010825220107480	0.219940124839679
7	0.013216243082027	0.053252644428581	0.177512700518577
8	0.015708573902135	0.115382672473579	0.115382672473579
9	0.013216243082027	0.177512700518577	0.053252644428581
10	0.006542197529252	0.219940124839679	0.010825220107480
11	0.016848134048440	0.023455038515334	0.476544961484666
12	0.034035816568844	0.115382672473579	0.384617327526421

(*continued*)

k	c_k	x_k	y_k
13	0.040454320987654	0.250000000000000	0.250000000000000
14	0.034035816568844	0.384617327526421	0.115382672473579
15	0.016848134048440	0.476544961484666	0.023455038515334
16	0.021807802470748	0.036084856923188	0.733149798129653
17	0.044055107973971	0.177512700518577	0.591721954534264
18	0.052363059235553	0.384617327526421	0.384617327526421
19	0.044055107973971	0.591721954534264	0.177512700518577
20	0.021807802470748	0.733149798129653	0.036084856923188
21	0.013375270558306	0.044709521703645	0.908380401265687
22	0.027020099316181	0.219940124839679	0.733149798129653
23	0.032115573564810	0.476544961484666	0.476544961484666
24	0.027020099316181	0.733149798129653	0.219940124839679
25	0.013375270558306	0.908380401265687	0.044709521703645

3. Composite integration over standard triangle T

We can discretise T in (x,y) space into $n\times n = n^2$ right isosceles triangle T_i each of area $1/(2n^2)$. This is depicted in Fig. 2.

By use of the linearity property of integrals, we can write from Eq. (1) and from the above discretisation of Fig. 1, we have

$$I = \int\int_T f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \int_0^1\int_0^{1-x} f(x,y)\mathrm{d}y\,\mathrm{d}x = \int_0^1\int_0^{1-y} f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \sum_{i=1}^{s\times s}\int\int_{T_i} f(x,y)\,\mathrm{d}x\,\mathrm{d}y$$
$$= \frac{1}{n^2}\int_0^1\int_0^{1-x} H(X,Y)\,\mathrm{d}Y\,\mathrm{d}X = \frac{1}{n^2}\int_0^1\int_0^{1-y} H(X,Y)\,\mathrm{d}X\,\mathrm{d}Y, \tag{12}$$

where

$$H(X,Y) = \sum_{j=0}^{n-1}\sum_{i=1}^{n-j} f\left(\frac{i-1}{n}+\frac{X}{n},\frac{j}{n}-\frac{Y}{n}\right) + \sum_{j=1}^{n-1}\sum_{i=1}^{n-j} f\left(\frac{i}{n}-\frac{X}{n},\frac{j}{n}-\frac{Y}{n}\right). \tag{13}$$

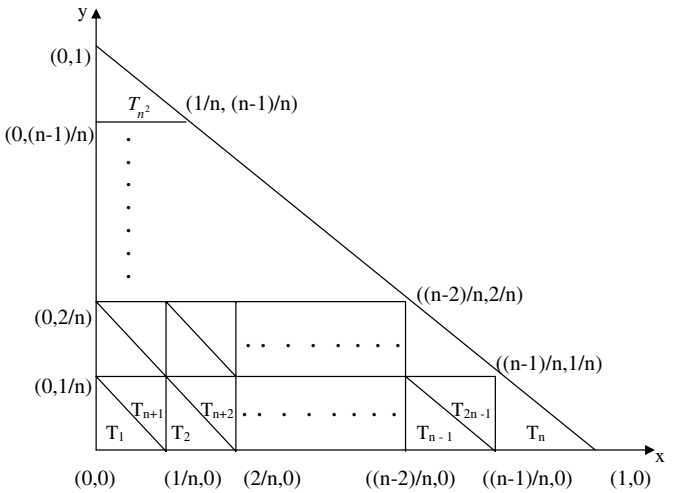

Fig. 2. Discretisation of T into n^2 subtriangles T_i.

We can now apply Gauss Legendre quadrature rules on the integral, in a manner similar to the procedure we already developed for integral $\int\int_T f(x,y)\,dx\,dy$. Following the method already developed in previous section, we have now on using the transformation:

$$X = (1+\xi)/2, \quad Y = (1-\xi)(1+\eta)/4. \tag{14}$$

The integral I in Eq. (12) can be written as:

$$\begin{aligned} I &= \int\int_T f(x,y)\,dx\,dy = \frac{1}{n^2}\int\int_T H(X,Y)\,dX\,dY = \frac{1}{n^2}\int_{-1}^{1}\int_{-1}^{1}\left(\frac{1-\xi}{8}\right)H(X(\xi,\eta),Y(\xi,\eta))\,d\xi\,d\eta \\ &= \frac{1}{n^2}\sum_{p=1}^{s}\sum_{q=1}^{s}\left(\frac{1-\xi_p}{8}\right)W_pW_qH(X(\xi_p,\eta_q),Y(\xi_p,\eta_q)), \end{aligned} \tag{15}$$

where

$$\begin{aligned} H(X,Y) &= \sum_{j=0}^{n-1}\sum_{i=1}^{n-j} f\left(\frac{i-1}{n}+\frac{X}{n},\frac{j}{n}+\frac{Y}{n}\right) + \sum_{j=1}^{n-1}\sum_{i=1}^{n-j} f\left(\frac{i}{n}-\frac{X}{n},\frac{j}{n}-\frac{Y}{n}\right), \\ X(\xi_p,\eta_q) &= \frac{(1+\xi_p)(1+\eta_q)}{4}, \\ Y(\xi_p,\eta_q) &= \frac{(1+\xi_p)(1-\eta_q)}{4} \quad (p,q=1,2,3,\ldots,s). \end{aligned} \tag{16}$$

From Eqs. (13)–(15), it is clear that, we have obtained the following composite integration rule:

$$I = \frac{1}{n^2}\sum_{k=1}^{N=s\times s} c_kH(x_k,y_k), \tag{17}$$

where

$$\begin{aligned} H(x_k,y_k) &= \sum_{j=0}^{n-1}\sum_{i=1}^{n-j} f\left(\frac{i-1}{n}+\frac{x_k}{n},\frac{j}{n}+\frac{y_k}{n}\right) + \sum_{j=1}^{n-1}\sum_{i=1}^{n-j} f\left(\frac{i}{n}-\frac{x_k}{n},\frac{j}{n}-\frac{y_k}{n}\right), \\ c_k &= \frac{(1+\xi_p)}{8}w_pw_q, \quad x_k = \frac{(1+\xi_p)(1+\eta_q)}{4}, \quad y_k = \frac{(1+\xi_p)(1-\eta_q)}{4}, \\ &(k=1,2,\ldots,N), \ (p,q=1,2,3,\ldots,s). \end{aligned} \tag{18}$$

We have listed here a **C- Program**, which computes integral $\int\int_T f(x,y)\,dx\,dy$ by the above-explained Composite Numerical Integration method:

C-Program

```
# include⟨stdio.h⟩
# include⟨alloc.h⟩
# include⟨math.h⟩
# include⟨complex.h⟩
# include⟨conio.h⟩
double fun(double X, double Y)
{
return (double) sqrt(X+Y);
}
void main()
{
double u,v,x[25],y[25],w,f,s,C[200];
double Fl[15][20][25];
```

```
double far *FP1,*FP2;
double F2[15][20][25];
double t1,t2,t3,t4,t6,t7,t8,t9,t11;
double sum1=0.0;
static double SUM11[200],SUM22[200],SUM33[200];
double far *S11,*S22,*S33;
int i,j,k,l,m,n;
clrscr();
printf("input m and n\n");
scanf("%d%d",&m,&n);
FP1=(double*)malloc(m*n*sizeof(double));
FP1=&F1[0][0][0];
FP2=(double*)malloc(m*n*sizeof(double));
FP2=&F2[0][0][0];
S11=(double*)malloc(m*m*sizeof(double));
S11=&SUM11[1];
S22=(double*)malloc(m*m*sizeof(double));
S22=&SUM22[1];
printf("Input x,y,C\n");
for(k=1;k<=m*m;++k){
fflush(stdin);
scanf("%lf%lf%lf",&x[k],&y[k],&C[k]);
}
for(k=1;k<=m*m;++k){
for(j=0;j<=n-1;++j){
  for(i=1;i<=n-j;++i){
  t1=(float)(i-1)/n;
  t2=(float)j/n;
  t3=(float)(x[k]/n);
  t4=(float)(y[k]/n);
  *(*(*(F1+k)+j)+i)=fun(t1+t3,t2+t4);
  *(S11+k)+=*(*(*(F1+k)+j)+i);
}}}
for(k=1;k<=m*m;++k)
for(j=1;j<=n-1;++j)
  for(i=1;i<=n-j;++i)
  t6=(float)i/n;
  t7=(float)j/n;
  t8=(float)x[k]/n;
  t9=(float)y[k]/n;
    *(*(*(F2+k)+j)+i)=fun(t6-t8,t7-t9);
*(S22+k)+=*(*(*(F2+k)+j)+i);
}
for(k=1;k<=m*m;++k){
SUM33[k]=*(S11+k)+*(S22+k);
SUM33[k]*=C[k];
sum1+=SUM33[k];
}
t11=(float)1/(n*n);
printf("The solution is:%12.12lf\n",t11*sum1);
getch();
}
```

4. Composite integration over standard triangle T

We now derive a new composite integration rule over the standard triangular surface T. We can evaluate the integrals over T by adopting a strategy similar to that used for the treatment of line integrals over arbitrarily shaped curves. We discretise the triangle T into three new triangles of equal area 1/6 units by joining the centroidal point of T, say $c = (1/3, 1/3)$ to the three vertices of T is $(0,0)$, $(1,0)$ and $(0,1)$, so that we can write $T = T_1^c + T_2^c + T_3^c$. This is depicted in Fig. 3

$$\int\int_T f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \int_0^1 \int_0^{1-x} f(x,y)\,\mathrm{d}y\,\mathrm{d}x = \int_0^1 \int_0^{1-y} f(x,y)\,\mathrm{d}x\,\mathrm{d}y$$
$$= \int\int_{T_1^c} f(x,y)\,\mathrm{d}x\,\mathrm{d}y + \int\int_{T_2^c} f(x,y)\,\mathrm{d}x\,\mathrm{d}y + \int\int_{T_3^c} f(x,y)\,\mathrm{d}x\,\mathrm{d}y. \tag{19}$$

We can transform each T_k^c $(k = 1,2,3)$ into the standard triangle T by use of the well-known affine transformations:

$$\int\int_{T_1^c} f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \frac{1}{3}\int_0^1 \int_0^{1-\xi} f\left(\frac{1}{3} - \frac{1}{3}\xi - \frac{1}{3}\eta, \frac{1}{3} + \frac{1}{3}\xi - \frac{1}{3}\eta\right)\mathrm{d}\xi\,\mathrm{d}\eta, \tag{20}$$

$$\int\int_{T_2^c} f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \frac{1}{3}\int_0^1 \int_0^{1-\xi} f\left(\frac{1}{3} - \frac{1}{3}\xi + \frac{2}{3}\eta, \frac{1}{3} - \frac{1}{3}\xi - \frac{1}{3}\eta\right)\mathrm{d}\xi\,\mathrm{d}\eta, \tag{21}$$

$$\int\int_{T_3^c} f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \frac{1}{3}\int_0^1 \int_0^{1-\xi} f\left(\frac{1}{3} + \frac{2}{3}\xi - \frac{1}{3}\eta, \frac{1}{3} - \frac{1}{3}\xi + \frac{2}{3}\eta\right)\mathrm{d}\xi\,\mathrm{d}\eta. \tag{22}$$

Using Eqs. (20)–(22) in Eq. (19), we obtain:

$$\int\int_T f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \frac{1}{3}\int_0^1 \int_0^{1-\xi} [f(R(\xi,\eta), P(\xi,\eta) + f(Q(\xi,\eta), R(\xi,\eta) + f(P(\xi,\eta), Q(\xi,\eta)]\mathrm{d}\xi\,\mathrm{d}\eta, \tag{23}$$

where

$$P(\xi,\eta) = \frac{1}{3} + \frac{2}{3}\xi - \frac{1}{3}\eta,$$
$$Q(\xi,\eta) = \frac{1}{3} - \frac{1}{3}\xi + \frac{2}{3}\eta, \tag{24}$$
$$R(\xi,\eta) = \frac{1}{3} - \frac{1}{3}\xi - \frac{1}{3}\eta.$$

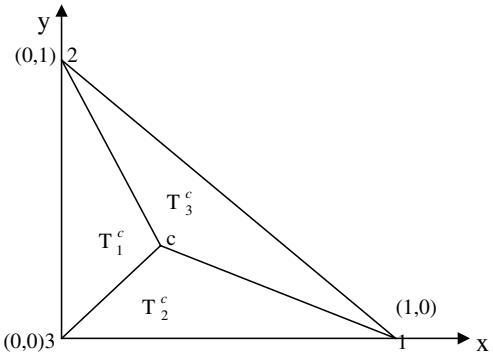

Fig. 3. Discretisation of T into three triangles T_i^c $(i = 1,2,3)$, each of equal area 1/6 unit, c: centroid of T.

Now, we can apply the quadrature rule of Eqs. (10), (11) to the above Eqs. (19)–(24), we find:

$$\int\int_T f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \frac{1}{3}\sum_{k=1}^{N=s\times s} C_k[f(P_k,Q_k)+f(Q_k,R_k)+f(R_k,P_k], \tag{25}$$

$$P_k = P(\xi_k,\eta_k), \quad Q_k = Q(\xi_k,\eta_k), \quad R_k = R(\xi_k,\eta_k). \tag{26}$$

We can compute the values of $P_k = P(\xi_k,\eta_k)$, $Q_k = Q(\xi_k,\eta_k)$, $R_k = R(\xi_k,\eta_k)$.

By identifying the equivalence of variables (ξ_k,η_k) with (x_k,y_k). We list below a C-Program which generates $(\xi_k,\eta_k)\equiv(x_k,y_k), C_k$, and then the quantities P_k, Q_k and R_k.

C-Program

```
#include⟨stdio.h⟩
#include⟨conio.h⟩
void main()
{
clrscr();
int n,i,j,k=l; double
pp[25][25],qq[25][25],rr[25][25],c[25][25],x[25][25],
y[25][25],wi[25],wj[25],p[25],q[25];
printf("Enter the value of n:");
scanf ("%d", &n);
printf("Enter the values of p:");
for(i=0; i<n; i++){
scanf("%lf",&p[i]);
}
printf("enter the values of q:");
for(i=0;i<n;i++){
scanf("%lf",&q[i]);
}
printf("enter the values of wi:");
for(i=0;i<n;i++){
scanf("%lf",&wi[i]);
  }
printf("enter the values of wj:");
for(i=0;i<n;i++){
scanf("%lf",&wj[i]);
   }
for(i=0;i<n;i++){
for(j=0;j<n;j++)
{
  c[i][j]=((l-p[i])/8.0)*(wi[i]*wj[j]);
  x[i][j]=((l+p[i])*(l+0*q[j])/2.0);
  y[i][j]=((l-p[i])*(l+q[j])/4.0);
}}
for(i=0;i<n;i++){
for(j=0;j<n;j++)
{
pp[i][j]=(l.0/3.0)+(2.0/3.0)*x[i][j]-(l.0/3.0)*y[i][j];
qq[i][j]=(l.0/3.0)-(l.0/3.0)*x[i][j]+(2.0/3.0)*y[i][j];
rr[i][j]=(l.0/3.0)-(l.0/3.0)*x[i][j]-(l.0/3.0)*y[i][j];
  }}
```

```
printf("%-l8s%-l8s%-l8s%-l8s\n",''k'',"ck","Pk","Qk","Rk");
  for(i=0;i<n;i++){
  for(j=0;j<n;j++)
    {
printf("%-2d%-l8.l5lf%-l8.l5lf%-l8.l5lf%-
l8.l5lf\n",k,c[i][j],pp[i][j],qq[i][j],rr[i][j]);
    }}
getch();
}
```

Sample output

k	c_k	P_k	Q_k	R_k
s = 2				
1	0.197168783	0.418661021000	0.374002823000	0.207336156000
2	0.197168783	0.266880421000	0.677564023000	0.055555556000
3	0.052831216	0.844230690000	0.100213754000	0.055555556000
4	0.052831216	0.803561200333	0.181552733333	0.014886066333
s = 3				
1	0.068464377	0.375134443333	0.362432778333	0.262432778333
2	0.109543004	0.260584721000	0.591532223000	0.147883056000
3	0.068464377	0.146034998667	0.820631667667	0.033333333667
4	0.061728395	0.647883056000	0.204233888000	0.147883056000
5	0.098765432	0.583333333333	0.333333333333	0.083333333333
6	0.061728395	0.518783611000	0.462432778000	0.018783611000
7	0.008696116	0.920631667667	0.046034998667	0.033333333667
8	0.013913785	0.906081945333	0.075134443333	0.018783611333
9	0.008696116	0.891532222667	0.104233888667	0.004233888667
s = 4				
1	0.028150383	0.358084208333	0.353263427333	0.288652364333
2	0.052775277	0.277255792333	0.514920259333	0.207823948333
3	0.052775277	0.171797281000	0.725837282000	0.102365437000
4	0.028150383	0.090968865000	0.887494114000	0.021537021000
5	0.037997148	0.537833426333	0.254342625333	0.207823948333
6	0.071235621	0.479638578000	0.370732322000	0.149629100000
7	0.071235621	0.403710552000	0.522588374000	0.073701074000
8	0.037997148	0.345515703667	0.638978070667	0.015506225667
9	0.018715815	0.772355959000	0.125278604000	0.102365437000
10	0.035087705	0.743691596000	0.182607330000	0.073701074000
11	0.035087705	0.706292607333	0.257405307333	0.036302085333
12	0.018715815	0.677628244333	0.314734033333	0.007637722333
13	0.002100365	0.952105177000	0.026357802000	0.021537021000
14	0.003937686	0.946074381667	0.038419392667	0.015506225667
15	0.003937686	0.938205878333	0.054156399333	0.007637722333
16	0.002100365	0.932175083000	0.066217990000	0.001606927000
s = 5				
1	0.013375271	0.349703544000	0.347502989000	0.302793467000
2	0.027020099	0.291293343000	0.464323391000	0.244383266000

(*continued*)

k	c_k	P_k	Q_k	R_k
3	0.032115574	0.205758397333	0.635393282333	0.158848320333
4	0.027020099	0.120223452000	0.806463173000	0.073313375000
5	0.013375271	0.061813251000	0.923283575000	0.014903174000
6	0.021807802	0.475148611000	0.280468123000	0.244383266000
7	0.044055108	0.428005996333	0.374753352333	0.197240651333
8	0.052363059	0.358971120667	0.512823103667	0.128205775667
9	0.044055108	0.289936245333	0.650892854333	0.059170900333
10	0.021807802	0.242793630667	0.745178083667	0.012028285667
11	0.016848134	0.658848320333	0.182303359333	0.158848320333
12	0.034035817	0.628205775667	0.243588448667	0.128205775667
13	0.040454321	0.583333333333	0.333333333333	0.083333333333
14	0.034035817	0.538460890667	0.423078218667	0.038460890667
15	0.016848134	0.507818346000	0.484363308000	0.007818346000
16	0.006542198	0.842548030000	0.084138595000	0.073313375000
17	0.013216243	0.828405555333	0.112423544333	0.059170900333
18	0.015708574	0.807695545667	0.153843563667	0.038460890667
19	0.013216243	0.786985536333	0.195263582333	0.017750881333
20	0.006542198	0.772843061667	0.223548531667	0.003608406667
21	0.000658317	0.967993097000	0.017103729000	0.014903174000
22	0.001329901	0.965118208667	0.022853505667	0.012028285667
23	0.001580695	0.960908269000	0.031273385000	0.007818346000
24	0.001329901	0.956698329667	0.039693263667	0.003608406667
25	0.000658317	0.953823441333	0.045443040333	0.000733518333

5. Composite integration over the standard triangle T with a mesh size of $3 \times n \times n$ subtriangles

We can discretise the standard triangle T in (x, y) space first into $n \times n = n^2$ right isosceles triangles T_i each of which has an area equal to $1/(2n^2)$. This is explained in the previous section (Fig. 2). We have seen that use of this discretisation transforms the integral $I = \int\int_T f(x,y)\,\mathrm{d}x\,\mathrm{d}y$ into an equivalent integral as in Eq. (12):

$$I = \int\int_T f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \frac{1}{n^2}\int\int_T H(X,Y)\,\mathrm{d}X\,\mathrm{d}Y, \tag{27}$$

where

$$H(X,Y) = \sum_{j=0}^{n-1}\sum_{i=1}^{n-j} f\left(\frac{i-1}{n}+\frac{X}{n}, \frac{j}{n}+\frac{Y}{n}\right) + \sum_{j=1}^{n}\sum_{i=1}^{n-j} f\left(\frac{i}{n}-\frac{X}{n}, \frac{j}{n}-\frac{Y}{n}\right). \tag{28}$$

Now using the composite integration rule over T with a mesh size of 3-subtriangles to the integral $\int\int_T G(X,Y)\,\mathrm{d}X\,\mathrm{d}Y$ in (X, Y) space, we obtain from Eqs. (27) and (28), see Fig. 4

$$\begin{aligned} I &= \int\int_T f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \frac{1}{n^2}\int\int_T H(X,Y)\,\mathrm{d}X\,\mathrm{d}Y \\ &= \frac{1}{3n^2}\sum_{k=1}^{s\times s} C_k\{H(P_k,Q_k)+H(Q_k,R_k)+H(R_k,P_k)\}, \end{aligned} \tag{29}$$

where

$$P_k = P(\xi_k,\eta_k), \quad Q_k = Q(\xi_k,\eta_k), \quad R_k = R(\xi_k,\eta_k). \tag{30}$$

We now list here a C-Program which evaluates the integral $\int\int_T f(x,y)\,\mathrm{d}x\,\mathrm{d}y$ by use of the composite numerical integration method with mesh size of $3 \times n \times n = 3n^2$ subtriangles.

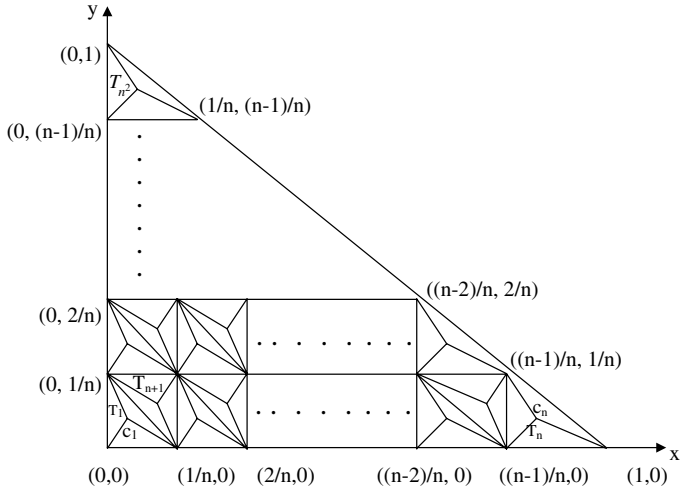

Fig. 4. Discretisation of T into n^2 subtriangles T_i and further discretisation of each T_i into 3-subtriangles, where c_i is centroid of T_i.

C-Program

```
#include⟨stdio.h⟩
#include⟨alloc.h⟩
#include⟨math.h⟩
#include⟨complex.h⟩
#include⟨conio.h⟩
double fun(double X, double Y)
{
return (double) l/sqrt(X+Y);
}
void main()
{
double u,v,x[25],y[25],w,f,p,q,r,s,P[25],Q[25],R[25],C[200];
double Fl[l5][20][25];
double far *FPl,*FP2;
double F2[l5][20][25];
double tl,t2,t3,t4,t5,t6,t7,t8,t9,tl0,tll;
double suml=0.0;
static double SUMll[200],SUM22[200],SUM33[200];
double far *Sll,*S22,*S33;
int i,j,k,l,m,n;
clrscr();
printf("input m and n\n");
scanf("%d%d",&m,&n);
FPl=(double*)malloc(m*n*sizeof(double));
FPl=&Fl[0][0][0];
FP2=(double*)malloc(m*n*sizeof(double));
FP2=&F2[0][0][0];
Sll=(double*)malloc(m*m*sizeof(double));
Sll=&SUMll[l];
S22=(double*)malloc(m*m*sizeof(double));
S22=&SUM22[l];
```

```
printf("Input x,y,C\n");
for(k=l;k<=m*m;++k){
fflush(stdin);
scanf("%lf%lf%lf",&x[k],&y[k],&C[k]);
}
for(k=l;k<=m*m;++k){
p=(float)l/3;
q=(float)2/3;
q*=x[k];
r=(float)l/3;
r*=y[k];
P[k]=p+q-r;
p=(float)l/3;
q=p;
q*=x[k];
r=(float)2/3;
r*=y[k];
Q[k]=p-q+r;
p=(float)l/3;
q=p;
r=p;
q*=x[k];
r*=y[k];
R[k]=p-q-r;
}
for(k=l;k<=m*m;++k){
  for(j=0;j<=n-l;++j){
    for(i=l;i<=n-j;++i){
      tl=(float)(i-l)/n;
      t2=(float)j/n;
      t3=(float)(P[k]/n);
      t4=(float)(Q[k]/n);
      t5=(float)(R[k]/n);
         *(*(*(Fl+k)+j)+i)=fun(tl+t3,
      t2+t4)+fun(tl+t4,t2+t5)+fun(tl+t5,t2+t3);
      *(Sll+k)+=*(*(*(Fl+k)+j)+i);
    }}}
    for(k=l;k<=m*m;++k){
    p=(float)l/3;
  q=(float)2/3;
q*=x[k];
r=(float)l/3;
r*=y[k];
P[k]=p+q-r;
p=(float)l/3;
q=p;
q*=x[k];
r=(float)2/3;
r*=y[k];
Q[k]=p-q+r;
p=(float)l/3;
q=p;
```

```
r=p;
q*=x[k];
r*=y[k];
R[k]=p-q-r;
}
  for(k=l;k<=m*m;++k)
    for(j=l;j<=n-l;++j)
for(i=l;i<=n-j;++i){
  t6=(float)i/n;
  t7=(float)j/n;
  t8=(float)P[k]/n;
  t9=(float)Q[k]/n;
  tl0=(float)R[k]/n;
    *(*(*(F2+k)+j)+i)=fun(t6-t8,t7-t9)+fun(t6-t9, t7-tl0)+fun(t6-tl0,t7-t8);
  *(S22+k)+=*(*(*(F2+k)+j)+i);
}
  for(k=l;k<=m*m;++k){
SUM33[k]=*(Sll+k)+*(S22+k);
SUM33[k]*=C[k];
suml+=SUM33[k];
}
  tll=(float)l/(3*n*n);
printf("The solution is:%l2.l2lf\n",tll*suml);
getch();
}
```

6. Some numerical results

We consider some typical integrals with known exact values [13]

$$I_1 = \int_0^1 \int_0^{1-y} (x+y)^{\frac{1}{2}} \mathrm{d}x\,\mathrm{d}y = 0.400000000,$$

Table Ia
Numerical results of double integration using n^2 subtriangles (Figs. 1 and 2) ($s = 2 =$ order of Gauss Legendre quadrature rule)

n^2	I_1	I_2	I_3	I_4	I_5
1^2	0.398773985	0.673887339	0.784678327	0.990476629	0.741130436
2^2	0.399774578	0.669239502	0.832879825	0.999463357	0.724537717
3^2	0.399917340	0.668068874	0.849030932	0.999896006	0.721108882
4^2	0.399959554	0.667577770	0.857113936	0.999967309	0.719881314
5^2	0.399976793	0.667318700	0.861965085	0.999986650	0.719308254
6^2	0.399985267	0.667162723	0.865199543	0.999993573	0.718995664
7^2	0.399989969	0.667060334	0.867509993	0.999996534	0.71880741
8^2	0.399992811	0.666988887	0.869242878	0.999997970	0.718683944
9^2	0.399994642	0.666936708	0.870590701	0.999998733	0.718599673
10^2	0.399995882	0.666897235	0.871668970	0.999999169	0.718539355
20^2	0.399999271	0.666748187	0.876521255	0.999999948	0.71834662
40^2	0.399999871	0.666695488	0.878947420	0.999999996	0.718297942
60^2	0.399999953	0.666682355	0.879756142	0.999999999	0.718288990
80^2	0.399999977	0.666676856	0.880160503	0.999999999	0.718285857
100^2	0.399999986	0.666673958	0.880403120	0.999999999	0.718284406
150^2	0.399999995	0.666670635	0.880726609	0.999999999	0.718282974
180^2	0.399999996	0.666669686	0.880834438	0.999999999	0.718282624

Table Ib
Numerical results of double integration using $3n^2$ subtriangles (Figs. 3 and 4) ($s = 2 =$ order of Gauss Legendre quadrature rule)

$3n^2$	I_1	I_2	I_3	I_4	I_5
3×1^2	0.400586257	0.653832751	0.795621029	0.995523202	0.725605886
3×2^2	0.400108254	0.662085929	0.838444724	0.999743374	0.720348543
3×3^2	0.400039740	0.664169457	0.852749211	0.999950091	0.719220497
3×4^2	0.400019454	0.665043935	0.859904276	0.999984294	0.718813824
3×5^2	0.400011165	0.665505314	0.864197835	0.999993583	0.718623492
3×6^2	0.400007089	0.665783113	0.867060347	0.999996909	0.718519543
3×7^2	0.400004827	0.665965477	0.869105045	0.999998333	0.718456675
3×8^2	0.400003459	0.666092734	0.870638588	0.999999023	0.718415794
3×9^2	0.400002578	0.666185671	0.871831352	0.999999391	0.718387731
3×10^2	0.400001982	0.666255979	0.872785568	0.999999600	0.718367640
3×20^2	0.400000350	0.666521461	0.877079568	0.999999975	0.718303306
3×40^2	0.400000062	0.666615328	0.879226657	0.999999998	0.718287199
3×60^2	0.400000022	0.666638721	0.879942247	0.999999999	0.718284216
3×80^2	0.400000010	0.666648516	0.880300082	0.999999999	0.718283171
3×100^2	0.400000006	0.666653678	0.880514783	0.999999999	0.718282687

Table IIa
Numerical results of double integration using n^2 subtriangles (Figs. 1 and 2) ($s = 3 =$ order of Gauss Legendre quadrature rule)

n^2	I_1	I_2	I_3	I_4	I_5
1^2	0.399812412	0.669179634	0.830150053	1.000145446	0.695312789
2^2	0.399966763	0.667555451	0.855760342	0.999999400	0.712492884
3^2	0.399987936	0.667150469	0.864298061	0.999999948	0.715704859
4^2	0.399994123	0.666980906	0.868566939	0.999999990	0.716831464
5^2	0.399996635	0.666891518	0.871128268	0.999999997	0.717353351
6^2	0.399997867	0.666837717	0.872835821	0.999999999	0.717636690
7^2	0.399998549	0.666802405	0.874055502	0.999999999	0.717808007
8^2	0.399998961	0.666777767	0.874970262	0.999999999	0.717919038
9^2	0.399999226	0.666759775	0.875681743	0.999999999	0.717995168
10^2	0.399999405	0.666746164	0.876250927	0.999999999	0.718049627
20^2	0.399999894	0.666694773	0.878812257	0.999999999	0.718223773
40^2	0.399999981	0.666676604	0.880092922	0.999999999	0.718267314
60^2	0.399999993	0.666672076	0.880519810	0.999999999	0.718275378
80^2	0.399999996	0.666670180	0.880733255	0.999999999	0.718278200
100^2	0.399999998	0.666691806	0.880861321	0.999999999	0.718279506

Table IIb
Numerical results of double integration using $3n^2$ subtriangles (Figs. 3 and 4) ($s = 3 =$ order of Gauss Legendre quadrature rule)

$3n^2$	I_1	I_2	I_3	I_4	I_5
3×1^2	0.400111238	0.661702315	0.835177742	1.000055058	0.710624869
3×2^2	0.400019710	0.664910806	0.858276209	1.000000776	0.716352170
3×3^2	0.400007154	0.665710876	0.865975347	1.000000066	0.717422837
3×4^2	0.400003485	0.666045860	0.869824908	1.000000011	0.717798373
3×5^2	0.400001995	0.666222453	0.872134644	1.000000003	0.717972336
3×6^2	0.400000860	0.666328742	0.873674467	1.000000001	0.718066872
3×7^2	0.400000616	0.666398503	0.874774342	1.000000000	0.718123888
3×8^2	0.400000458	0.666447178	0.875599247	1.000000000	0.718160898
3×9^2	0.400000352	0.666482723	0.876240841	1.000000000	0.718186275
3×10^2	0.400000062	0.666509613	0.876754115	1.000000000	0.718204428
3×20^2	0.400000011	0.666611140	0.879063851	0.999999999	0.718262477
3×40^2	0.400000003	0.666647035	0.880218719	0.999999999	0.718276990
3×60^2	0.400000001	0.666655980	0.880603675	0.999999999	0.718279678
3×80^2	0.400000000	0.666659725	0.880796153	0.999999999	0.718280618
3×100^2	0.400000000	0.666661700	0.880911639	0.999999999	0.718281054

Table IIIa
Numerical results of double integration using n^2 subtriangles (Figs. 1 and 2) ($s = 4 =$ order of Gauss Legendre quadrature rule)

n^2	I_1	I_2	I_3	I_4	I_5
1^2	0.399950385	0.667827645	0.849816063	0.999998699	0.705297478
2^2	0.399982352	0.667170624	0.860150053	0.999999446	0.713312710
3^2	0.399996763	0.666555451	0.865760542	0.999999900	0.716501184
4^2	0.399999436	0.666850469	0.874298062	0.999999999	0.716704859
5^2	0.399999823	0.666780912	0.876566940	0.999999999	0.717921421
6^2	0.399999935	0.666741512	0.878128268	0.999999999	0.717353351
7^2	0.399999967	0.666717717	0.879835201	0.999999999	0.717636690
8^2	0.399999989	0.666692405	0.880055502	0.999999999	0.718008007
9^2	0.399999991	0.666687767	0.880270262	0.999999999	0.718069038
10^2	0.399999993	0.666679775	0.880481421	0.999999999	0.718195168
20^2	0.399999995	0.666676164	0.880650920	0.999999999	0.718249627
40^2	0.399999997	0.666674721	0.880812257	0.999999999	0.718263721
60^2	0.399999999	0.666672604	0.880992922	0.999999999	0.718277314
80^2	0.399999999	0.666671076	0.881089810	0.999999999	0.718280878
100^2	0.399999999	0.666660180	0.881133220	0.999999999	0.718281200

Table IIIb
Numerical results of double integration using $3n^2$ subtriangles (Figs. 3 and 4) ($s = 4 =$ order of Gauss Legendre quadrature rule)

$3n^2$	I_1	I_2	I_3	I_4	I_5
3×1^2	0.400041256	0.663202305	0.845567709	1.000001054	0.714158652
3×2^2	0.400009760	0.665910821	0.863267209	1.000000078	0.717342180
3×3^2	0.400004127	0.666210812	0.869975343	1.000000006	0.717922582
3×4^2	0.400002485	0.666445860	0.872834907	1.000000001	0.718098893
3×5^2	0.400000995	0.666452478	0.874138600	1.000000000	0.718142336
3×6^2	0.400000160	0.666524743	0.876570462	1.000000000	0.718186872
3×7^2	0.400000096	0.666608503	0.877774356	1.000000000	0.718203876
3×8^2	0.400000052	0.666625178	0.879512247	1.000000000	0.718220998
3×9^2	0.400000012	0.666632723	0.880040867	1.000000000	0.718246235
3×10^2	0.400000002	0.666644613	0.880154123	1.000000000	0.718254428
3×20^2	0.400000000	0.666651180	0.880342678	0.999999999	0.718272477
3×40^2	0.400000000	0.666655035	0.880218719	0.999999999	0.718280990
3×60^2	0.400000001	0.666661980	0.880903672	0.999999999	0.718281178
3×80^2	0.400000000	0.666663145	0.881296143	0.999999999	0.718281318
3×100^2	0.400000000	0.666664771	0.881411678	0.999999999	0.718281454

$$I_2 = \int_0^1 \int_0^{1-y} (x+y)^{\frac{-1}{2}} \,\mathrm{d}x\,\mathrm{d}y = 0.666666667,$$

$$I_3 = \int_0^1 \int_0^{y} (x^2+y^2)^{\frac{-1}{2}} \,\mathrm{d}x\,\mathrm{d}y = 0.881373587,$$

$$I_4 = \int_0^{\frac{\pi}{2}} \int_0^{y} \sin(x+y) \,\mathrm{d}x\,\mathrm{d}y = 1.000000000,$$

$$I_5 = \int_0^1 \int_0^{y} \mathrm{e}^{|x+y-1|} \,\mathrm{d}x\,\mathrm{d}y = 0.71828183.$$

These integrals were evaluated using the two integration schemes of previous Sections 2 and 3 derived in the present paper and it is found that excellent convergence occurs to the exact value. The results are summarized in Tables I–III.

7. Conclusions

We have derived various orders ($s = 2, 3, 4, 5 \ldots$) extended numerical integration rules based on classical Gauss Legendre quadrature. This is made possible by transforming the triangular surface: $0 \leqslant x,$

$y \leqslant 1, x + y \leqslant 1$ to a standard 2-square; $-1 \leqslant \xi, \eta \leqslant 1$. Over the 2-square, the Gauss Legendre quadrature rule of all orders is applicable. It is the main purpose of this paper to evolve a practical and workable algorithm for composite numerical integration over triangular surfaces and it converges to the exact value of the integral, for sufficiently large value of n, even for the lower order Gauss Legendre quadrature rules.

References

[1] O.C. Zienkiewicz, The Finite Element Method, third ed., McGraw-Hill, London, 1977.
[2] J.N. Reddy, An Introduction to the Finite Element Method, McGraw-Hill Book Company, New York, 1984.
[3] T.J.R. Hughes, The Finite Element Method, Static and Dynamic Analysis, Prentice Hall, Englewood Cliffs, NJ, 1987.
[4] P.C. Hammer, O.J. Marlowe, A.H. Stroud, Numerical integration over simplexes and cones, Math. Tables and other aids to computation 10 (1956) 130–136.
[5] P.C. Hammer, A.H. Stroud, Numerical integration over simplexes, Math. Tables and other aids to computation 10 (1956) 137–139.
[6] P.C. Hammer, A.H. Stroud, Numerical evaluation of multiple integrals, Math. Tables and other aids to computation 12 (1958) 272–280.
[7] G.R. Cowper, Gaussian quadrature formulas for triangles, Int. J. Numer. Meth. Eng. 7 (1973) 405–408.
[8] J.N. Lyness, D. Jespersen, Moderate degree symmetric quadrature rules for the triangle, J. Inst. Math. Appl. 15 (1975) 19–32.
[9] F.G. Lannoy, Triangular finite elements and numerical integration, Comput. Struct. 7 (1977) 613.
[10] M.E. Lauresn, M. Gellert, Some criteria for numerically integrated matrices and quadrature formulas for triangles, Int. J. Numer. Meth. Eng. 12 (1978) 67–76.
[11] D.A. Dunavant, High degree efficient symmetrical Gaussian Quadrature Rules for the triangle, Int. J. Numer. Meth. Eng. 21 (1985) 1129–1148.
[12] D.P. Laurie, Automatic numerical integration over a triangle, CSIR Spec. Rep. WISK 273, National Institute for Mathematical Sciences, Pretoria, 1977.
[13] P. Sylvester, Symmetric quadrature formulae for simplexes, Math. Comput. 24 (1970) 95–100.
[14] F.G. Lethor, Computation of double integrals over a triangle, J. Comp. Appl. Math. 2 (1976) 219–224.
[15] P. Hillion, Numerical integration on a triangle, Int. J. Numer. Meth. Eng. 11 (1977) 797–815.
[16] M. Abramowicz, I.A. Stegun (Eds.), Handbook of Mathematical Functions, Dover Publications, 1964.
[17] C.T. Reddy, Improved three point integration schemes for triangular finite elements, Int. J. Numer. Meth. Eng. 12 (1978) 1890–1896.
[18] C.T. Reddy, D.J. Shippy, Alternative integration formulae for triangular finite elements, Int. J. Numer. Meth. Eng. 17 (1981) 133–139.
[19] G. Lague, R. Baldur, Extended numerical integration method for triangular surfaces, Int. J. Numer. Meth. Eng. 11 (1977) 388–392.
[20] H.T. Rathod, K.V. Nagaraja, Symmetric Gauss Legendre quadrature over a triangle, J. Sci. Eng. Technol. 17 (1) (2005) 1–8.
[21] H.T. Rathod, K.V. Nagaraja, B. Venkatesudu, N.L. Ramesh, Gauss Legendre quadrature over a triangle, J. Indian Inst. Sci. 84 (2004) 183–188.
[22] H.T. Rathod, K.V. Nagaraja, B. Venkatesudu, Gauss Legendre quadrature over a triangle, Acharya Nagarjuna Int. J. Math. Inform. Technol. 1 (1) (2004) 33–52.
[23] H.T. Rathod, K.V. Nagaraja, B. Venkatesudu, Symmetric Gauss Legendre quadrature formulas for composite numerical integration over a triangular surface, Appl. Math. Comput., in press, doi:10.1016/j.amc.2006.10.041.
[24] Brice Carnahan, H.A. Luther, James.O. Wilkes, Applied Numerical Methods, John Wiley and Sons, 1969.
[25] C.F. Gerald, P.O. Wheatley, Applied Numerical Analysis, sixth ed., Lowprice edition., Pearson Education, Asia, 1999.
[26] S.C. Chapra, R.P. Canale, Numerical Methods for Science and Engineering Tata, McGraw-Hill, New Delhi, 2000.
[27] R.L. Burden, J.D. Fairs, Numerical Analysis, fourth ed., PWS-KENT Publishing Company, 1989.