A compendium of FEM integration formulas for symbolic work

Carlos A. Felippa

Department of Aerospace Engineering Sciences and Center for Aerospace Structures, University of Colorado, Boulder, Colorado, USA

Abstract *This paper presents a set of Mathematica modules that organizes numerical integration rules considered useful for finite element work. Seven regions are considered: line segments, triangles, quadrilaterals, tetrahedral, wedges, pyramids and hexahedra. Information can be returned in floating-point (numerical) form, or in exact symbolic form. The latter is useful for computer-algebra aided FEM work that carries along symbolic variables. A few quadrature rules were extracted from sources in the FEM and computational mathematics literature, and placed in symbolic form using Mathematica to generate own code. A larger class of formulas, previously known only numerically, were directly obtained through symbolic computations. Some unpublished non-product rules for pyramid regions were found and included in the collection. For certain regions: quadrilaterals, wedges and hexahedra, only product rules were included to economize programming. The collection embodies most FEM-useful formulas of low and moderate order for the seven regions noted above. Some gaps as regard region geometries and omission of non-product rules are noted in the conclusions. The collection may be used "as is" in support of symbolic FEM work thus avoiding contamination with floating arithmetic that precludes simplification. It can also be used as generator for low-level floating-point code modules in Fortran or C. Floating point accuracy can be selected arbitrarily. No similar modular collection applicable to a range of FEM work, whether symbolic or numeric, has been published before.*

1. Introduction

The use of symbolic computation in support of computational methods in engineering and sciences is steadily growing. This is due to technical improvements in general-purpose computer algebra systems (CAS) such as Mathematica and Maple, as well as availability on inexpensive personal computers and laptops. (This migration keeps licensing costs reasonable.) Furthermore, Maple is available as a toolbox of the widely used Matlab system. A related factor is wider exposure in higher education: many universities currently have site licences, which facilitate lab access and use of CAS in course assignments and projects.

In finite element work, CAS tools can be used for a spectrum of tasks: formulation, prototyping, implementation, performance and evaluation, and automatic code generation. Although occasionally advertised as "doing mathematics by computer"

the phrase is misleading: as of now only humans can do mathematics. But a CAS can provide timely help. Here is a first-hand FEM example: the writer needed four months to formulate, implement and test the 6-node membrane triangle in the summer and fall of 1965 as part of thesis work. Using a CAS, a similar process can be completed in less than a week, and demonstrated to students in 20 min.

The finite elements have developed with CAS support since 1984 – using the venerable Macsyma for the free formulation elements presented in Bergan and Felippa (1985) and Felippa and Bergan (1987). The development of templates as a unified framework for element families (Felippa, 2000, 2003) would not have been possible without that assistance.

Not all is good news, as can be observed when a beginner comes face-to-face with an unfamiliar phenomenon: exact versus floating-point work. The dichotomy does not exist in ordinary numerical computations, which are floating-point based. In computer algebra work, inadvertent use of just one floating-point number can be the kiss of death. Why? CAS algebraic expressions tend to "combinatorially explode" in intermediate stages. The inversion of a symbolic 16×16 matrix results in $16! = 20922789888000$ adjoint terms. How is then one able to get rules in minutes or hours? Selective simplification. At any sign of combinatorial explosion the human intervenes, requesting the program to carry out simplifications as appropriate. However a CAS may, and often will, balk at simplifying expressions that contain a mixture of symbols and floating-point numbers. A simple example: $3 \cdot a - 3 \cdot a$ will simplify to 0, but $3 \cdot a - 3 \cdot a$ will not.

From experience the following operational rule emerges: avoid mixing floating-point numbers and symbols in CAS calculations. Proceed to floating-point only when all expressions are numeric, or in display of final results.

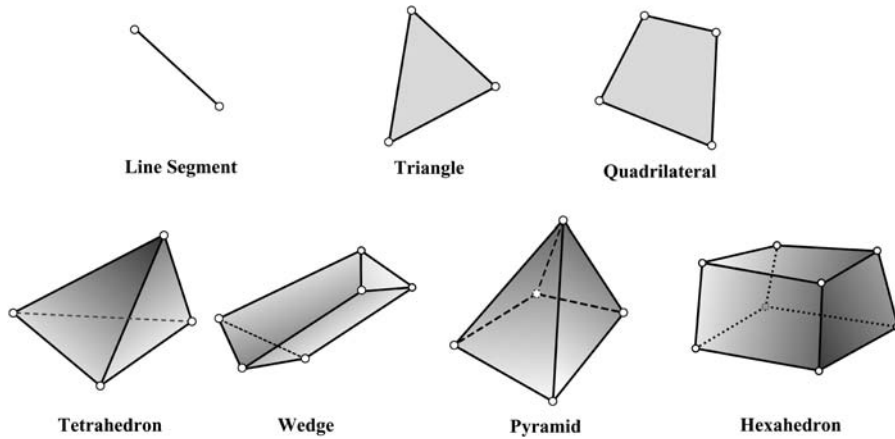## 2. General description
2.1 Integration regions
Numerical integration has been a staple of FEM work since the mid 1960s, as narrated in Section 2.7. While comprehensive collections of formulas for a wide variety of element regions are currently available, textbooks – and more recently Web sites – usually tabulate abscissas and weights in floating-point form. As discussed in the Introduction, this is undesirable for computer-aided symbolic manipulation.

This compilation is organized as a source library of Mathematica modules that store formulas useful for seven element geometries shown in Figure 1: line segment, triangle, quadrilateral, tetrahedron, wedge, pyramid and hexahedron. The regions may contain non-corner geometric nodes as shown in Figure 2. However, one- and two-dimensional regions must obey the restrictions listed in Table I.

2.2 Restrictions and omissions
The integration formulas collected here are restricted in the following sense:

(1) Formulas with exterior points or negative weights are excluded Only fully symmetric formulas (in the sense discussed in Section 2.6) are accepted.

(2) Preference is given to formulas for which exact expressions of abscissas and weights in rational or algebraic-quadratic form are either known or may be derived. For some high order rules, however, this is not possible and a "rationalization" procedure is used instead.

Region	Acronym[a]	C+E+F[b]	Global coordinates	Restrictions
Line segment	Line	2+1+0	x	Straight line, along x-axis
Triangle	Trig	3+3+1	x, y	Flat: in $\{x, y\}$ plane, curved sides allowed
Quadrilateral	Quad	4+4+1	x, y	Flat: in $\{x, y\}$ plane, curved sides allowed
Tetrahedron	Tetr	4+6+4	x, y, z	None
Wedge	Wedg	6+9+5	x, y, z	None
Pyramid	Pyra	5+8+5	x, y, z	None
Hexahedron	Hexa	8+12+6	x, y, z	None

Notes: [a]Acronym may be followed by a node count, e.g. Trig10 means a triangle with ten nodes;
[b]C – corners; E – edges; F – faces

(3) For quadrilaterals, wedges and hexahedra only tensor-product formulas are included to keep the logic of the modules simple and simplify the production of anisotropic rules. Non-product formulas for those regions are available in the literature but are not included in this compilation.

The compilation is admittedly incomplete as regards regions. It lacks polygons with more than four sides, polyhedra with more than six faces, curved line segments and non-flat surfaces (e.g. for doubly curved shell elements). It also omits non-product rules for three regions as noted above.

There are transition polyhedra, produced in 3D mesh generation, that connect a quadrilateral face on one side to a triangle, line, or apex point on the other. The latter two regions (wedge and pyramid, respectively) are included. The first one (as yet unnamed), shown in Figure 3, is excluded as being comparatively rare.

2.3 Organization, access and applications

The collection is organized as seven Mathematica modules, one for each region The hierarchical organization is shown in Figure 4. Modules for the line segment, triangle, tetrahedron and pyramid are self-contained. Modules for quadrilaterals, wedges and hexahedra build formulas as tensor products of lower dimension rules. Information can be extracted in exact (symbolic) form or in floating-point form, as specified by an input argument.

All modules are encapsulated in a single Mathematica Notebook, which is an ASCII file. The file is available from the author on e-mail request. The modules may be used directly as such, in support of CAS computations, or converted to C, C++ or Fortran procedures for use in numerical computations. The conversion may be done through
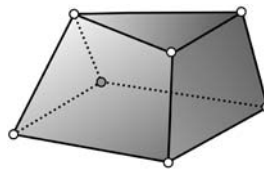
Figure 3.
Omitted transition region
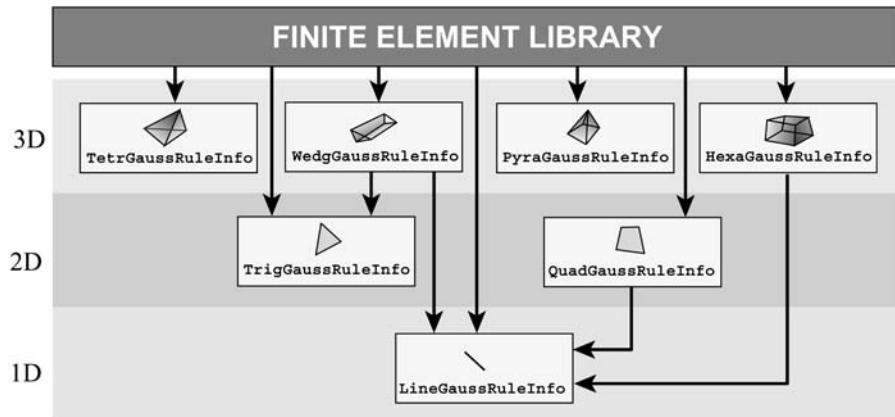
Figure 4.
Hierarchical organization of the seven integration rule modules

output filters such as //CForm and //FortranForm, or by hand. The availability of exact expressions makes relatively easy to pass, for example, from 64-bit double precision to 128-bit quadwords as PCs migrate to 64-bit CPUs over the next 10 years. Expressions may be conveniently made into C macros, C++ inline functions, or Fortran 90 parameters to force numerical evaluation of abscissas and weights at compile time.

2.4 Natural coordinates, Jacobian, reference regions

Table II lists natural coordinates used for the different regions, as well as the geometry definition. The natural coordinates selected are those in common use in the FEM literature. The region geometry is defined isoparametrically, although element formulations need not be so. The definition is in terms of n geometric nodes, which for simple regions are the corners, and n shape functions N_i. The latter are part of shape

Region	Natural coordinates[a]	Range of natural coordinates	Iso-P geometry definition in terms of n geometric nodes and shape functions N_i
Line segment	ξ	$\|-1, 1\|$	$\|x\| = \begin{vmatrix} x_1 & x_2 & \ldots & x_n \end{vmatrix} \begin{bmatrix} N_1 \\ \vdots \\ N_n \end{bmatrix}$
Triangle	$\zeta_1, \zeta_2, \zeta_3$	$\|0, 1\|$	$\begin{bmatrix} 1 \\ x \\ y \end{bmatrix} = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ x_1 & x_2 & \ldots & x_n \\ y_1 & y_2 & \ldots & y_n \end{bmatrix} \begin{bmatrix} N_1 \\ \vdots \\ N_n \end{bmatrix}$
Quadrilateral	ξ, η	$\|-1, 1\|$	$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & \ldots & x_n \\ y_1 & y_2 & \ldots & y_n \end{bmatrix} \begin{bmatrix} N_1 \\ \vdots \\ N_n \end{bmatrix}$
Tetrahedron	$\zeta_1, \zeta_2, \zeta_3, \zeta_4$	$\|0, 1\|$	$\begin{bmatrix} 1 \\ x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ x_1 & x_2 & \ldots & x_n \\ y_1 & y_2 & \ldots & y_n \\ z_1 & z_2 & \ldots & z_n \end{bmatrix} \begin{bmatrix} N_1 \\ \vdots \\ N_n \end{bmatrix}$
Wedge	$\zeta_1, \zeta_2, \zeta_3, \xi$	$\zeta_i : \|0, 1\|, \xi : \|-1, 1\|$	same as tetrahedron
Pyramid	ξ, η, μ	$\|-1, 1\|$	same as hexahedron
Hexahedron	ξ, η, μ	$\|-1, 1\|$	$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & \ldots & x_n \\ y_1 & y_2 & \ldots & y_n \\ z_1 & z_2 & \ldots & z_n \end{bmatrix} \begin{bmatrix} N_1 \\ \vdots \\ N_n \end{bmatrix}$

Note: [a]NC constraints: $\zeta_1 + \zeta_2 + \zeta_3 = 1$ for triangles and wedges, $\zeta_1 + \zeta_2 + \zeta_3 + \zeta_4 = 1$ for tetrahedral

Table II.
Natural coordinates and isoparametric geometry definition

function modules and not covered here. The Jacobian matrices that relate Cartesian to natural coordinates, and the associated Jacobian determinant J, are defined in Table III.

If J is constant, the integration region is said to be a constant metric region, or CMR. If so J is directly related to the volume (area, length) measure of the region, as listed in the last column of Table III. The presence of scaling factors is due to the $[-1, 1]$ range of natural coordinates used in four of the regions; e.g. $J = (1/4)A$ and $J = (1/8)V$ for constant-metric quadrilaterals and hexahedra, respectively. The pyramid cannot be a CMR because $j = 0$ at the apex.

A reference region or RR is one of particularly simple geometry over which the integration rules are developed. For example, the quadrilateral RR is a rectangle of side lengths a and b. For all RR except the pyramid J is constant. The reference pyramid, defined in Section 8, has a J with simple polynomial dependence on the distance from the apex.

If the dimensions of the RR are simple numbers, it is called a unit reference region or URR. For example, the quadrilateral URR is the square of side 2.

Region	Jacobian matrix	Determinant	J for CMR
Line segment	$\mathbf{J} = [x_i \partial N_i / \partial \xi]$	$J = \det[\mathbf{J}]$	$\frac{1}{2}L$
Triangle	$\mathbf{J} = \begin{bmatrix} 1 & 1 & 1 \\ x_i\partial N_i/\partial\zeta_1 & x_i\partial N_i/\partial\zeta_2 & x_i\partial N_i/\partial\zeta_3 \\ y_i\partial N_i/\partial\zeta_1 & y_i\partial N_i/\partial\zeta_2 & y_i\partial N_i/\partial\zeta_3 \end{bmatrix}$	$J = \frac{1}{2}\det[\mathbf{J}]$	A
Quadrilateral	$\mathbf{J} = \begin{bmatrix} x_i\partial N_i/\partial\xi & x_i\partial N_i/\partial\eta \\ y_i\partial N_i/\partial\xi & y_i\partial N_i/\partial\eta \end{bmatrix}$	$J = \det[\mathbf{J}]$	$\frac{1}{4}A$
Tetrahedron	$\mathbf{J} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ x_i\partial N_i/\partial\zeta_1 & x_i\partial N_i/\partial\zeta_2 & x_i\partial N_i/\partial\zeta_3 & x_i\partial N_i/\partial\zeta_4 \\ y_i\partial N_i/\partial\zeta_1 & y_i\partial N_i/\partial\zeta_2 & y_i\partial N_i/\partial\zeta_3 & y_i\partial N_i/\partial\zeta_4 \\ z_i\partial N_i/\partial\zeta_1 & z_i\partial N_i/\partial\zeta_2 & z_i\partial N_i/\partial\zeta_3 & z_i\partial N_i/\partial\zeta_4 \end{bmatrix}$	$J = \frac{1}{6}\det[\mathbf{J}]$	V
Wedge	$\mathbf{J} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ x_i\partial N_i/\partial\zeta_1 & x_i\partial N_i/\partial\zeta_2 & x_i\partial N_i/\partial\zeta_3 & x_i\partial N_i/\partial\xi \\ y_i\partial N_i/\partial\zeta_1 & y_i\partial N_i/\partial\zeta_2 & y_i\partial N_i/\partial\zeta_3 & y_i\partial N_i/\partial\xi \\ z_i\partial N_i/\partial\zeta_1 & z_i\partial N_i/\partial\zeta_2 & z_i\partial N_i/\partial\zeta_3 & z_i\partial N_i/\partial\xi \end{bmatrix}$	$J = \frac{1}{2}\det[\mathbf{J}]$	$\frac{1}{2}V$
Pyramid	same as hexahedron		N/A
Hexahedron	$\mathbf{J} = \begin{bmatrix} x_i\partial N_i/\partial\xi & x_i\partial N_i/\partial\eta & x_i\partial N_i/\partial\mu \\ y_i\partial N_i/\partial\xi & y_i\partial N_i/\partial\eta & y_i\partial N_i/\partial\mu \\ z_i\partial N_i/\partial\xi & z_i\partial N_i/\partial\eta & z_i\partial N_i/\partial\mu \end{bmatrix}$	$J = \det[\mathbf{J}]$	$\frac{1}{8}V$

Table III.
Jacobian matrices and determinants

Notes: V = volume of 3D region, A = area of 2D region, L = length of line segment. Pyramid cannot be CMR. Summation convention over i assumed in expressions of $\mathbf{J}$

2.5 Integration rule notation
Denote the domain of integration by Ω. The set of k natural coordinates is genericly written as array $\beta = \{\beta_1, \ldots \beta_k\}$.

An integration rule with p points is defined by p abscissas β_i and corresponding weights w_i, for $i = 1, \ldots p$. The position located by the abscissas β_i is called a sample point or integration point.

Application of the rule to a function $\mathbf{F}(\beta)$ expressed in natural coordinates results in:

$$\int_{\Omega} \mathbf{F}(\beta) \, \mathrm{d}\Omega \approx \sum_{i=1}^{p} w_i J_i \, \mathbf{F}(\beta_i), \tag{1}$$

where $J_i = J(\beta_i)$ is the Jacobian determinant evaluated at the ith sample point. In FEM work $\mathbf{F}$ is usually a matrix (in stiffness or mass computations) or a vector (in force computations).

A formula is said of degree d if it integrates exactly all natural-coordinate monomials of the form $\beta_1^{i_1} \ldots \beta_k^{i_k}$, $i_1 + \cdots i_k \leq d$, over a CMR.

If the region has no CMR multiple definitions of degree are possible. For the pyramid two definitions are given in Section 8.

2.6 Symmetry and stars
All formulas implemented in the modules are fully symmetric in the sense of being observer invariant. More precisely: the same result must be obtained if the geometric nodes are cyclically renumbered, which changes the natural coordinates (Stated mathematically: the integral (1) remains invariant under all affine transformations of the region into itself). Translating this invariance requirement to the different regions gives the conditions listed in Table IV.

To give an example, consider the triangle. Suppose the ith sample point has natural coordinates ζ_{1i}, ζ_{2i}, ζ_{3i} linked by $\zeta_{1i} + \zeta_{2i} + \zeta_{3i} = 1$. Then all points obtained by permuting the three indices must be also sample points and have the same weight w_i. If the three values are different this gives six sample points:

$$\{\zeta_{1i}, \zeta_{2i}, \zeta_{3i}\}, \quad \{\zeta_{1i}, \zeta_{3i}, \zeta_{2i}\}, \quad \{\zeta_{2i}, \zeta_{1i}, \zeta_{3i}\}, \quad \{\zeta_{2i}, \zeta_{3i}, \zeta_{1i}\}, \quad \{\zeta_{3i}, \zeta_{1i}, \zeta_{2i}\}, \quad \{\zeta_{3i}, \zeta_{2i}, \zeta_{1i}\}. \tag{2}$$

This set is said to form a sample point star or simple star, which is denoted by S_{111}. If two values are equal, the set (2) coalesces to three different points, and the star is

Region	If sample point i has coordinates	These must be sample points with same weight:	Sample point stars
Line segment	ξ_I	$\pm \xi_i$	S_2, S_{11}
Triangle	$\zeta_{1i}, \zeta_{2i}, \zeta_{3i}$	$P_{123}(\zeta_{1i}, \zeta_{2i}, \zeta_{3i})$	S_3, S_{21}, S_{111}
Quadrilateral	ξ_i, η_i	$\pm \xi, \pm \eta$	$\{S_2, S_{11}\} \times \{S_2, S_{11}\}$
Tetrahedron	$\zeta_{1i}, \zeta_{2i}, \zeta_{3i}, \zeta_{4i}$	$P_{1234}(\zeta_{1i}, \zeta_{2i}, \zeta_{3i}, \zeta_{4i})$	$S_4, S_{31}, S_{22}, S_{211}, S_{1111}$
Wedge	$\zeta_{1i}, \zeta_{2i}, \zeta_{3i}, \xi_i$	$P_{123}(\zeta_{1i}, \zeta_{2i}, \zeta_{3i}), \pm \xi_i$	$\{S_3, S_{21}, S_{111}\} \times \{S_2, S_{11}\}$
Pyramid	ξ_i, η_i, μ_i	$\pm \xi_i, \pm \eta_i$ (no condition on μ)	$\{S_2, S_{11}\} \times \{S_2, S_{11}\} \times S_\mu$
Hexahedron	ξ_i, η_i, μ_i	$\pm \xi_i, \pm \eta_i, \pm \mu_i$	$\{S_2, S_{11}\} \times \{S_2, S_{11}\} \times \{S_2, S_{11}\}$

Notes: $P_{123}(.)$: the set of 3! Permutations of natural coordinate subscripts 1, 2, 3. Likewise for P_{1234}

denoted by S_{21}. Finally if the three values coalesce, which can only happen for the centroid $\zeta_{1i} = \zeta_{2i} = \zeta_{3i} = \frac{1}{3}$, the set (2) reduces to one point and the star is denoted by S_3.

Possible stars for symmetric rules are enumerated in the last column of Table IV. When stars are built as tensor products over individual natural coordinates, the symbol $\times$ is used.

2.7 Historical sketch, Web resources

Numerical integration came into FEM by the mid-1960s. Five triangle integration rules were tabulated in Felippa's (1966, pp. 38-9) thesis. These were gathered from three sources: two early papers on simplicial integration rules (Hammer and Stroud, 1956, 1958) and Section 25.4 of the Handbook of Mathematical Functions (Abramowitz and Stegun, 1964). They were adapted to FEM by converting Cartesian abscissas to triangle natural coordinates. The table has been reproduced as Table 8.2 of Zienkiewicz (1971) since its second edition and, with corrections and additions, on Strang and Fix (1973, p. 184). Compilation of tetrahedral rules lagged behind.

Gauss product formulas for quadrilaterals and hexahedra (bricks) were forcefully advocated by Irons (1966) and Irons and Ahmad (1980) as key ingredient of the isoparametric formulation he created. In doing so he converted the range of the natural coordinates originally defined by Tiag and Kerr (1964) from $[0, 1]$ to $[-1, 1]$ to simplify fit to tables. Irons also recommended the use of non-product formulas for high order hexahedra (Irons, 1971).

On the numerical analysis side, Stroud's monograph (Stroud, 1971) is regarded as "the bible" in the topic of numerical cubature. That book gathers most of the formulas known by 1970, as well as references until that year (Only a small fraction of Stroud's tabulated rules, however, are suitable for FEM work). The collection has been periodically kept up to date by Cools (1999a, b, 2003), who also maintains a dedicated Web site: `http://www.cs.kuleuven.ac.be/~nines/research/ecf/ecf.html`. This site provides rule information in 16- and 32-digit floating-point accuracy for many geometries and dimensionalities – far more than those treated here – as well as a linked "index card" of references to source publications.

3. Line segment

3.1 One-dimensional Gauss rules

The Mathematica module `LineGaussRuleInfo` listed in Figure 5 returns exact or floating-point information for the first five 1D Gauss rules, whose sample points are the zeros of the Gauss-Legendre polynomials. The basic properties of these rules are summarized in Table V.

To extract information for the ith point of the pth rule, in which $1 \leq i \leq p$ and $p = 1, \ldots, 5$, the module is invoked as:

$$\{xi, w\} = \text{LineGaussRuleInfo}[\{p, numer\}, i]$$

Here, logical flag `number` is `True` to get numerical (floating-point) information, `or` `False` to get exact information in the form of rational or algebraic numbers. `LineGaussRuleInfo` returns the sample point abscissa ξ_i in `xi` and the weight w_i in w. For example, `LineGaussRuleInfo[{3,False},2]` returns {0, 8/9}.

```
LineGaussRuleInfo[{rule_,numer_},point_]:= Module[
   {g2={-1,1}/Sqrt[3],w3={5/9,8/9,5/9},
    g3={-Sqrt[3/5],0,Sqrt[3/5]},
    w4={(1/2)-Sqrt[5/6]/6, (1/2)+Sqrt[5/6]/6,
       (1/2)+Sqrt[5/6]/6, (1/2)-Sqrt[5/6]/6},
    g4={-Sqrt[(3+2*Sqrt[6/5])/7],-Sqrt[(3-2*Sqrt[6/5])/7],
        Sqrt[(3-2*Sqrt[6/5])/7], Sqrt[(3+2*Sqrt[6/5])/7]},
    g5={-Sqrt[5+2*Sqrt[10/7]],-Sqrt[5-2*Sqrt[10/7]],0,
        Sqrt[5-2*Sqrt[10/7]], Sqrt[5+2*Sqrt[10/7]]}/3,
    w5={322-13*Sqrt[70],322+13*Sqrt[70],512,
        322+13*Sqrt[70],322-13*Sqrt[70]}/900,
    i=point,p=rule,info={Null,0}},
   If [p==1, info={0,2}];
   If [p==2, info={g2[[i]],1}];
   If [p==3, info={g3[[i]],w3[[i]]}];
   If [p==4, info={g4[[i]],w4[[i]]}];
   If [p==5, info={g5[[i]],w5[[i]]}];
   If [numer, Return[N[info]], Return[Simplify[info]]];
];
```

Figure 5.
Line-segment Gauss
integration rule
information module

Ident	Stars	Points	Degree
1	S_2	1	1
2	S_{11}	2	3
3	$S_2 + S_{11}$	3	5
4	$2S_{11}$	4	7
5	$S_2 + 2S_{11}$	5	9

Table V.
Line segment Gauss
formulas

But `LineGaussRuleInfo[{3,True},2]`, under default working precision of 10^{-16}, returns `{0.0, 0.888888888888889}`. If p is not 1 through 5, the module returns `{Null,0}`.

The p-point rule has degree $d = 2p - 1$. Abscissas and weights are available in handbooks. For example, Table 25.4 of Abramowitz and Stegun (1964) tabulates rules with up to 96 points. For $p = 6$ and 7 abscissas and weights can be exactly given in terms of radicals but the expressions are exceedingly complex and difficult to simplify. If $p \geq 8$ only numerical values are available. Line rules with more than 5 points, however, are rarely used in FEM work.

3.2 Application example

Suppose one wants the consistent translational mass matrix of a tapered, Bernoulli-Euler, two-node plane beam element with transverse displacements w defined by the standard cubic shape functions. The cross section A is interpolated linearly from the end areas A_1 and A_2. The integrand $\rho A \mathbf{N} \mathbf{N}^T$, where ρ is the mass density and $\mathbf{N}$ the shape function matrix, is of order 7 in the natural coordinate ξ. This should be integrated exactly by line-segment Gauss rules of 4 or more points. The Mathematica module listed in Figure 6 implements the symbolic computation of $\mathbf{M}^{(e)}$ using Gauss rules 1 through 5 points. The mass matrix returned by p = 4 and 5 is shown in Figure 7. The reproducing-matrix effect provides a good check of implementation correctness.

4. Triangles

Symmetric integration rules over triangles must be of non-product type. Sample point stars S_3, S_{21} and S_{111} have 1, 3, or 6 points, respectively, as discussed in Section 2.6. Consequently, symmetric rules can have $i + 3j + 6k$ points, where i, j, k are non-negative integers and i is 0 or 1. This restriction excludes low-order rules with 2, 5, 8, 10 and 11 points.

Table VI lists seven FEM-useful rules for the triangle geometry. All of them comply with the requirements listed in Section 2.2. These rules are implemented in the Mathematica module `TrigGaussRuleInfo` shown in Figure 8. The implementation is self-contained.

The module is called as

$$\{\{zeta1, zeta2, zeta3\}, w\} = TrigGaussRuleInfo[\{rule, numer\}, i] \tag{3}$$

The module has three arguments: `rule`, `numer` and `i`.

The first two are grouped in a two-item list. Argument `rule`: 1, 3, -3, 6, -6, 7 or 12, identifies the integration formulas as follows. `Abs [rule]` is the number of sample points. If there are two useful rules with the same number of points, the most accurate one is identified with a positive value and the other one with a minus sign. In the case of the triangle this happens for rules with 3 and 6 points. The degree 2 rule identified as $+3$ has three interior points. The mid-point rule is identified as -3 also has degree 2 but is less accurate. For six points `rule = 6` gives a formula of degree 4 whereas if `rule = $-6$` a degree 3 formula, which is a linear combination of the $+3$ and -3 rules, is selected.

Logical flag number is set to `True` or `False` to request floating-point or exact information, respectively, for rules with 1-7 points. As regards the 12-point rule see below.

Argument `i` is the index of the sample point, which may range from 1 through `Abs [rule]`.

```
TranMassTaperedHermitianBeamElement [{L_,A1_,A2_},r_,p_]:=
  Module[{i,x,w,A,Me=Table[0,{4},{4}]},
    For [i=1,i<=p,i++,
      {x,w}=LineGaussRuleInfo[{p,False},i];
      A=A1*(1-x)/2+A2*(1+x)/2;
      Ne={{2*(1-x)^2*(2+x),  (1-x)^2*(1+x)*L,
          2*(1+x)^2*(2-x),-(1+x)^2*(1-x)*L}}/8;
      Me+=w*(L/2)*r*A*Transpose[Ne].Ne];
    Return[Simplify[Me]]];
  ClearAll[L,r,p,A1,A2];
  For[p=1,p<=5,p++,
    Me=TranMassTaperedHermitianBeamElement[{L,A1,A2},r,p];
    Print[Me//MatrixForm]  ];
```

Figure 6.
Module for computing the transitional mass matrix of a tapered beam with Gauss rule order as parameter

Figure 7.
Exact transitional mass matrix produced by rules with 4 and 5 points

$$\begin{bmatrix} \frac{1}{35} H10 A1 + 3 A2L L r & \frac{1}{420} H15 A1 + 7 A2L L^2 r & \frac{9}{140} HA1 + A2L L r & -\frac{1}{420} H7 A1 + 6 A2L L^2 r \\ \frac{1}{420} H15 A1 + 7 A2L L^2 r & \frac{1}{840} H5 A1 + 3 A2L L^3 r & \frac{1}{420} H6 A1 + 7 A2L L^2 r & -\frac{1}{280} HA1 + A2L L^3 r \\ \frac{9}{140} HA1 + A2L L r & \frac{1}{420} H6 A1 + 7 A2L L^2 r & \frac{1}{35} H3 A1 + 10 A2L L r & -\frac{1}{420} H7 A1 + 15 A2L L^2 r \\ -\frac{1}{420} H7 A1 + 6 A2L L^2 r & -\frac{1}{280} HA1 + A2L L^3 r & -\frac{1}{420} H7 A1 + 15 A2L L^2 r & \frac{1}{840} H3 A1 + 5 A2L L^3 r \end{bmatrix}$$

Ident	Stars	Points	Degree	Comments
1	S_3	1	1	Centroid rule, useful for Trig3 stiffness
3	S_{21}	3	2	Useful for Trig6 stiffness and Trig3 mass
-3	S_{21}	3	2	Midpoint rule, simpler but less accurate that $+3$
6	$2S_{21}$	6	4	Useful for Trig10 stiffness and Trig6 mass
-6	$2S_{21}$	6	3	A linear combination of $+3$ and -3 rules
7	$S_3 + 2S_{31}$	7	5	Radon's formula, also useful for Trig10 stiffness
12	$2S_{21} + S_{111}$	12	6	Useful for Trig10 mass; exact form not available

Note: The 4- and 13-point symmetric rules, listed in some FEM books, have one negative weight. There is a 12-point rule degree 7 with internal points, but it is fully symmetric

Table VI.
Triangle integration
formulas

```
TrigGaussRuleInfo[{rule_,numer_},point_]:= Module[
 {zeta,p=rule,i=point,g1,g2,g3,g4,w1,w2,w3,eps=10.^(-24),
 jkl={{1,2,3},{2,1,3},{1,3,2},{3,1,2},{2,3,1},{3,2,1}},
 info={{Null,Null,Null},0}  },
 If [p== 1, info={{1/3,1/3,1/3},1}];
 If [p== 3, info={{1,1,1}/6,1/3}; info[[1,i]]=2/3];
 If [p==-3, info={{1,1,1}/2,1/3}; info[[1,i]]=0  ];
 If [p== 6,   g1=(8-Sqrt[10]+Sqrt[38-44*Sqrt[2/5]])/18;
              g2=(8-Sqrt[10]-Sqrt[38-44*Sqrt[2/5]])/18;
      If [i<4, info={{g1,g1,g1},(620+Sqrt[213125-
              53320*Sqrt[10]])/3720}; info[[1,i]]=1-2*g1];
      If [i>3, info={{g2,g2,g2},(620-Sqrt[213125-
              53320*Sqrt[10]])/3720}; info[[1,i-3]]=1-2*g2]];
 If [p== -6,
      If [i<4, info={{1,1,1}/6,3/10}; info[[1,i]]=2/3];
      If [i>3, info={{1,1,1}/2,1/30}; info[[1,i-3]]=0]];
 If [p== 7,   g1=(6-Sqrt[15])/21; g2=(6+Sqrt[15])/21;
      If [i<4,      info={{g1,g1,g1},(155-Sqrt[15])/1200};
                    info[[1,i]]=  1-2*g1];
      If [i>3&&i<7, info={{g2,g2,g2},(155+Sqrt[15])/1200};
                    info[[1,i-3]]=1-2*g2];
      If [i==7,     info={{1/3,1/3,1/3},9/40} ]];
 If [p== 12,
      g1=0.063089014491502228340331602870819157;
      g2=0.249286745170910421291638553107019076;
      g3=0.053145049844816947353249671631398147;
      g4=0.310352451033784405416607733956552153;
      If [!numer, {g1,g2,g3,g4}=Rationalize[{g1,g2,g3,g4},eps]];
      w1=(30*g2^3*(4*g3^2+(1-2*g4)^2+4*g3*(-1+g4))+
          g3^2*(1-15*g4)+(-1+g4)*g4-g3*(-1+g4)*(-1+15*g4)+
          2*g2*(1+60*g3*g4*(-1+g3+g4))-6*g2^2*(3+10*(-1+g4)*g4+
          10*g3^2*(1+3*g4)+10*g3*(-1+g4)*(1+3*g4)))/
          (180*(g1-g2)*(-(g2*(-1+2*g2)*(-1+g3)*g3)+(-1+g3)*(g2-2*g2^2-
          2*g3+3*g2*g3)*g4-(g2*(-1+2*g2-3*g3)+2*g3)*g4^2+
          2*g1^2*(g2*(-2+3*g2)+g3-g3^2+g4-g3*g4-g4^2)+
          g1*(-4*g2^2+(-1+g3)*g3+(-1+g3)*(1+3*g3)*g4+(1+3*g3)*
          g4^2-2*g2*(-1+g3^2+g3*(-1+g4)+(-1+g4)*g4))));
      w2=(-1+12*(2-3*g1)*g1*w1+4*g3^2*(-1+3*w1)+4*g3*(-1+g4)*(-1+3*w1)+
          4*(-1+g4)*g4*(-1+3*w1))/(12*(g2*(-2+3*g2)+g3-g3^2+g4-g3*g4-g4^2));
      w3=(1-3*w1-3*w2)/6;
      If [i<4,      info={{g1,g1,g1},w1}; info[[1,i]]=  1-2*g1];
      If [i>3&&i<7, info={{g2,g2,g2},w2}; info[[1,i-3]]=1-2*g2];
      If [i>6,   {j,k,l}=jkl[[i-6]]; info={{0,0,0},w3};
          info[[1,j]]=g3; info[[1,k]]=g4; info[[1,l]]=1-g3-g4]];
 If [numer, Return[N[info]], Return[Simplify[info]]];
 ];
```

Figure 8.
Triangle integration rule
information module

The module returns the list $\{\{\zeta_1, \zeta_2, \zeta_3\}, w\}$. Here $\zeta_1, \zeta_2, \zeta_3$ are the triangular coordinates of the sample point, stored in {zeta1, zeta2, zeta3}, and w is the integration weight, placed in w. For example, the call TrigGaussRuleInfo[{3, False},1] returns {{2/3, 1/6, 1/6}, 1/3}. If rule is not that of an implemented formula the module returns {{Null, Null, Null}, 0}.

Exact data for the 12-point rule is not available because abscissas are roots of a sixth order polynomial. Their values are given as floating-point numbers with 36-place accuracy. If flag number is False, abscissas are converted to rational numbers that represent them to 16 place accuracy, using the built-in function Rationalize. The weights are recovered from the abscissas. The conversion precision can be adjusted through the value of internal variable eps, which is set in the module preamble. This rationalization procedure should be used with care since it may lead to unwieldy fractions in the results; postprocessing those into simple fractions requires substantial CAS expertise.

5. Quadrilaterals

Only Gauss product rules are implemented for quadrilaterals. Although symmetric non-product rules can be useful to speed up high order element formation in numerical FEM codes, they are omitted here to keep the logic simple, and to facilitate production of anisotropic rules.

Product rules are obtained by applying the one-dimensional rules to each natural coordinate direction: ξ and η, in turn. They are implemented in the Mathematica module QuadGaussRuleInfo shown in Figure 9 as tensor products of one-dimensional rules with 1-5 points. This module calls LineGaussRuleInfo twice. If the number of points along ξ and η is the same, the rule is called isotropic, and anisotropic otherwise. The five isotropic rules available from QuadGaussRuleInfo are listed in Table VII.

For an isotropic $p \times p$ rule the module is called in either of two ways:

```
QuadGaussRuleInfo[{rule_,numer_},point_]:= Module[
 {ξ,η,p1,p2,i,j,w1,w2,m,info={{Null,Null},0}},
 If [Length[rule]==2, {p1,p2}=rule, p1=p2=rule];
 If [Length[point]==2, {i,j}=point, m=point;
    j=Floor[(m-1)/p1]+1; i=m-p1*(j-1) ];
 {ξ,w1}= LineGaussRuleInfo[{p1,numer},i];
 {η,w2}= LineGaussRuleInfo[{p2,numer},j];
 info={{ξ,η},w1*w2};
 If [numer, Return[N[info]], Return[Simplify[info]]];
];
```

Figure 9.
Quadrilateral integration rule information module

Ident	Product rule	Points	Degree	Comments
1	1×1	1	1	Used is reduced and selective integration
2	2×2	4	3	Useful for Quad4 stiffness and mass
3	3×3	9	5	Useful for Quad9 stiffness and mass
4	4×4	16	7	Useful for Quad16 stiffness and mass
5	5×5	25	9	Rarely used
Note: Anisotropic rules, such as 2×1, are requested using a two-entry identifier				

Table VII.
Isotropic Gauss-product formulas for quadrilaterals

$$\{\{xi,eta\},w\} = QuadGaussRuleInfo[\{p,numer\}, \{i,j\}]$$

$$\{\{xi,eta\},w\} = QuadGaussRuleInfo[\{p,numer\}, m]$$

(4)

The first form is used to get information for point $\{i, j\}$ for the $p \times p$ rule, in which $1 \leq i \leq p$ and $1 \leq j \leq p$. The second form specifies the sample point by a "visiting counter" m that runs from 1 through p

^{2}, if so i and j are internally extracted as $j = \mathrm{Floor}[(m-1)/p] + 1$ and $i = m - p*(j-1)$.

For an anisotropic rule with p_1 points in the ξ direction and p_2 points in the η direction, the module may be called also in two ways:

$$\{\{xi,eta\},w\} = QuadGaussRuleInfo[\{\{p1,p2\},numer\}, \{i,j\}]$$

$$\{\{xi,eta\},w\} = QuadGaussRuleInfo[\{\{p1,p2\},numer\}, m]$$

(5)

In the first form i runs from 1 to p_1 and j from 1 to p_2. In the second form m runs from 1 to $p_1 p_2$; if so i and j are extracted by $j = \mathrm{Floor}[(m-1)/p1] + 1$ and $i = m - p1*(i-1)$.

In all four forms flag number is set to True if floating-point information is desired and to False if exact information is desired.

The module returns ξ_i and η_j in xi and eta, respectively, and the weight product $w_i w_j$ in w. If the number of points is outside the range of the implementation, the module returns {{Null, Null},0}.

For example: QuadGaussRuleInfo[{3,False},(2,3)] returns {{0,Sqrt[3/5]},40/81}.

6. Tetrahedra

As in the case of the triangle, fully symmetric integration rules over tetrahedra must be of non-product type. Sample point stars S_4, S_{31}, S_{22}, S_{211} and S_{1111} have 1, 4, 6, 12 or 24 points, respectively. Thus possible rules can have $i + 4j + 6k + 12l + 24m$ points, where i, j, k, l, m are non-negative integers and i is 0 or 1. This restriction exclude rules with 2, 3, 8 and 11 points. Table VIII lists nine FEM-useful rules for the tetrahedral geometry, all of which comply with the requirements listed in Section 2.1.

The rules of Table VIII are implemented in a Mathematica module called TetrGaussRuleInfo. Because of its length the module logic is split in two-figures: Figures 10 and 11. The implementation is self-contained. The module is invoked as

$$\{\{zeta1,zeta2,zeta3,zeta4\},w\}=TetrGaussRuleInfo[\{rule,numer\},i]$$

(6)

The module has three arguments: rule, numer and i. The first two are grouped in a two-item list.

Argument rule, which can be 1, 4, 8, -8, 14, -14, 15, -15 or 24, designates the integration formula as follows. Abs [rule] is the number of sample points. If there are two useful rules with the same number of points, the most accurate one is identified with a positive value and the other one with a minus value. For example, there are two useful 8-point rules. If rule $= 8$ a formula with all interior points is chosen.

Ident	Stars	Points	Degree	Comments
1	S_4	1	1	Centroid formula, useful for Tetr4 stiffness
4	S_{31}	4	2	Useful for Tetr4 mass and Tetr10 stiffness
8	$2S_{31}$	8	3	
− 8	$2S_{31}$	8	3	Has corners and face centers as sample points
14	$2S_{31} + S_{22}$	14	4	Useful for Tetr10 mass; exact form unavailable
− 14	$2S_{31} + S_{22}$	14	3	Has edge midpoints as sample points
15	$S_4 + 2S_{31} + S_{22}$	15	5	Useful for Tetr21 stiffness
− 15	$S_4 + 2S_{31} + S_{22}$	15	4	Less accurate than above one
24	$3S_{21} + S_{211}$	24	6	Useful for Tetr21 mass; exact form unavailable

Note: The five-point, degree-3, symmetric rule listed in some FEM textbooks has a negative weight

Table VIII.
Tetrahedra integration formulas

```
TetrGaussRuleInfo[{rule_,numer_},point_]:= Module[{
  jk6= {{1,2},{1,3},{1,4},{2,3},{2,4},{3,4}},
  jk12={{1,2},{1,3},{1,4},{2,3},{2,4},{3,4},
        {2,1},{3,1},{4,1},{3,2},{4,2},{4,3}},
  i=point,j,k,g1,g2,g3,g4,h1,w1,w2,w3,eps=10.^(-16),
  info={{Null,Null,Null,Null},0}  },
  If [rule==1, info={{1/4,1/4,1/4,1/4},1}];
  If [rule==4, g1=(5-Sqrt[5])/20; h1=(5+3*Sqrt[5])/20;
      info={{g1,g1,g1,g1},1/4}; info[[1,i]]=h1];
  If [rule==8, j=i-4;
      g1=(55-3*Sqrt[17]+Sqrt[1022-134*Sqrt[17]])/196;
      g2=(55-3*Sqrt[17]-Sqrt[1022-134*Sqrt[17]])/196;
      w1=1/8+Sqrt[(1715161837-406006699*Sqrt[17])/23101]/3120;
      w2=1/8-Sqrt[(1715161837-406006699*Sqrt[17])/23101]/3120;
      If [j<=0,info={{g1,g1,g1,g1},w1}; info[[1,i]]=1-3*g1];
      If [j> 0,info={{g2,g2,g2,g2},w2}; info[[1,j]]=1-3*g2]];
  If [rule==-8, j=i-4;
      If [j<=0,info={{0,0,0,0},  1/40}; info[[1,i]]=1];
      If [j> 0,info={{1,1,1,1}/3,9/40}; info[[1,j]]=0]];
  If [rule==14,    (* g1,g2 +roots of P(g)=0, P=9+96*g-
      1712*g^2-30464*g^3-127232*g^4+86016*g^5+1060864*g^6 *)
      g1=0.09273525031089122640232391373703060;
      g2=0.31088591926330060979734573376345783;
      g3=0.45449629587435035050811947372066056;
      If [!numer,{g1,g2,g3}=Rationalize[{g1,g2,g3},eps]];
      w1=(-1+6*g2*(2+g2*(-7+8*g2))+14*g3-60*g2*(3+4*g2*
          (-3+4*g2))*g3+4*(-7+30*g2*(3+4*g2*(-3+4*g2))*g3^2)/
          (120*(g1-g2)*(g2*(-3+8*g2)+6*g3+8*g2*(-3+4*g2)*g3-4*
          (3+4*g2*(-3+4*g2))*g3^2+8*g1^2*(1+12*g2*
          (-1+2*g2)+4*g3-8*g3^2)+g1*(-3-96*g2^2+24*g3*(-1+2*g3)+
          g2*(44+32*(1-2*g3)*g3)))));
      w2=(-1-20*(1+12*g1*(2*g1-1))*w1+20*g3*(2*g3-1)*(4*w1-1))/
          (20*(1+12*g2*(2*g2-1)+4*g3-8*g3^2));
      If [i<5,       info={{g1,g1,g1,g1},w1};info[[1,i]]=1-3*g1];
      If [i>4&&i<9,  info={{g2,g2,g2,g2},w2};info[[1,i-4]]=1-3*g2];
      If [i>8,       info={{g3,g3,g3,g3},1/6-2*(w1+w2)/3};
          {j,k}=jk6[[i-8]]; info[[1,j]]=info[[1,k]]=1/2-g3] ];
  If [rule==-14,
      g1=(243-51*Sqrt[11]+2*Sqrt[16486-9723*Sqrt[11]/2])/356;
      g2=(243-51*Sqrt[11]-2*Sqrt[16486-9723*Sqrt[11]/2])/356;
      w1=31/280+Sqrt[(13686301-3809646*Sqrt[11])/5965]/600;
      w2=31/280-Sqrt[(13686301-3809646*Sqrt[11])/5965]/600;
      If [i<5,       info={{g1,g1,g1,g1},w1};info[[1,i]]=1-3*g1];
      If [i>4&&i<9,  info={{g2,g2,g2,g2},w2};info[[1,i-4]]=1-3*g2];
      If [i>8&&i<15,info={{0,0,0,0},2/105};
          {j,k}=jk6[[i-8]]; info[[1,j]]=info[[1,k]]=1/2]];
```

Figure 10.
Tetrahedral integration rule information module, part 1 of 2

```
If [rule==15,
    g1=(7-Sqrt[15])/34;  g2=7/17-g1; g3=(10-2*Sqrt[15])/40;
    w1=(2665+14*Sqrt[15])/37800; w2=(2665-14*Sqrt[15])/37800;
    If [i<5,        info={{g1,g1,g1,g1},w1};info[[1,i]]=1-3*g1];
    If [i>4&&i<9, info={{g2,g2,g2,g2},w2};info[[1,i-4]]=1-3*g2];
    If [i>8&&i<15,info={{g3,g3,g3,g3},10/189};
        {j,k}=jk6[[i-8]]; info[[1,j]]=info[[1,k]]=1/2-g3];
    If [i==15,info={{1/4,1/4,1/4,1/4},16/135}] ];
If [rule==-15,   g1=(13-Sqrt[91])/52;
    If [i<5,        info={{1,1,1,1}/3,81/2240};info[[1,i]]=0];
    If [i>4&&i<9, info={{1,1,1,1}/11,161051/2304960};
                    info[[1,i-4]]=8/11];
    If [i>8&&i<15,info={{g1,g1,g1,g1},338/5145};
        {j,k}=jk6[[i-8]]; info[[1,j]]=info[[1,k]]=1/2-g1];
    If [i==15,info={{1/4,1/4,1/4,1/4},6544/36015}] ];
If [rule==24,
    g1=0.21460287125915202928883921938628499l;
    g2=0.040673958534611353115579448956410059;
    g3=0.322337890142275510343994470762492125;
    If [!numer,{g1,g2,g3}=Rationalize[{g1,g2,g3},eps]];
    w1= (85+2*g2*(-319+9*Sqrt[5]+624*g2)-638*g3-
        24*g2*(-229+472*g2)*g3+96*(13+118*g2*(-1+2*g2))*g3^2+
        9*Sqrt[5]*(-1+2*g3))/(13440*(g1-g2)*(g1-g3)*(3-8*g2+
        8*g1*(-1+2*g2)-8*g3+16*(g1+g2)*g3));
    w2= -(85+2*g1*(-319+9*Sqrt[5]+624*g1)-638*g3-
        24*g1*(-229+472*g1)*g3+96*(13+118*g1*(-1+2*g1))*g3^2+
        9*Sqrt[5]*(-1+2*g3))/(13440*(g1-g2)*(g2-g3)*(3-8*g2+
        8*g1*(-1+2*g2)-8*g3+16*(g1+g2)*g3));
    w3= (85+2*g1*(-319+9*Sqrt[5]+624*g1)-638*g2-
        24*g1*(-229+472*g1)*g2+96*(13+118*g1*(-1+2*g1))*g2^2+
        9*Sqrt[5]*(-1+2*g2))/(13440*(g1-g3)*(g2-g3)*(3-8*g2+
        8*g1*(-1+2*g2)-8*g3+16*(g1+g2)*g3));
    g4=(3-Sqrt[5])/12; h4=(5+Sqrt[5])/12; p4=(1+Sqrt[5])/12;
    If [i<5,        info={{g1,g1,g1,g1},w1};info[[1,i]]=  1-3*g1];
    If [i>4&&i<9, info={{g2,g2,g2,g2},w2};info[[1,i-4]]=1-3*g2];
    If [i>8&&i<13,info={{g3,g3,g3,g3},w3};info[[1,i-8]]=1-3*g3];
    If [i>12,info={{g4,g4,g4,g4},27/560};
        {j,k}=jk12[[i-12]];info[[1,j]]=h4;info[[1,k]]=p4] ];
    If [numer, Return[N[info]], Return[Simplify[info]]];
];
```

Figure 11.
Tetrahedral integration
rule information module,
part 2 of 2

If rule = − 8 a formula with sample points at the four corners and the four face
centers (less accurate but simpler and easy to remember) is picked.

Logical flag numer is set to True or False to request floating-point or exact
information, respectively, for rules other than +14 or +24. For the latter see below.

Argument i is the index of the sample point, which may range from one through
Abs [rule].

The module returns the list $\{\{\zeta_1, \zeta_2, \zeta_3, \zeta_4\}, w\}$, where ζ_1, ζ_2, ζ_3, ζ_4 are the natural
coordinates of the sample point, and w is the integration weight. For example,
TetrGaussRuleInfo [{4,False},2] returns {{(5− Sqrt[5])/20,
(5+3*Sqrt[5])/20,(5 − Sqrt[5])/20,(5 − Sqrt[5])/20},1/4}.

If rule is not implemented the module returns {{Null, Null, Null,
Null}, 0}.

Exact information for rules +14 and +24 is either unknown or only partly known.
For these the abscissas are given in floating-point form with 36 exact digits. If flag
numer is False, the abscissas are converted to rational numbers that represent the
data to 16 place accuracy, using the built-in function Rationalize. Weights are
recovered from the abscissas. The conversion precision can be changed by adjusting
the value of internal variable eps, which is set in the module preamble.

7. Wedges
Wedge regions, also called pentahedra (an incorrect term, since the pyramid also has five faces) and triangular prisms, appear in FEM meshes as transition elements. Wedge integration rules are of product type. They are constructed as a tensor product of:

(1) triangle rules for the triangular cross sections, which have natural coordinates $\{\zeta_1, \zeta_2, \zeta_3\}$, by invoking TrigGaussRuleInfo, and

(2) one-dimensional Gauss rules for the longitudinal (prismatic) direction, which has natural coordinate ξ, by invoking LineGaussRuleInfo.

Module WedgGaussRuleInfo, shown in Figure 12, can return any combination of the seven rules implemented in TrigGaussRuleInfo and the five rules in LineGaussRuleInfo. It is invoked as

$$\{\{\text{zeta1},\text{zeta2},\text{zeta3},\text{xi}\},w\}= \text{WedgGaussRuleInfo}[\{\{p1,p2\},$$
$$\text{numer}\},\{i1,i2\}]$$
$$\{\{\text{zeta1},\text{zeta2},\text{zeta3},\text{xi}\},w\}= \text{WedgGaussRuleInfo}[\{\{p1,p2\},$$
$$\text{numer}\},i] \tag{7}$$

The first argument is an integer pair: {p1, p2}. Here p1 = 1, 3, −3, 6, −6, 7, 12 specifies the triangle rule and p2 = 1, 2, 3, 4, 5 the line-segment rule, respectively. The total number of points is Abs [p1]*p2. The degree of the rule is the minimum of the degree of the triangle rule and of the line segment rule. For example, if the argument is {7, 4}, the degree is min(5, 7) = 5.

Argument numer is a logical flag specifying floating-point information if True and exact information if False.

The last argument may be a two-integer list: {i1, i2}, or a single integer: i. In the first form i1 is interpreted as triangular rule point index, which can vary from 1 to Abs [p1], whereas i2 is the line-segment rule point index, which can vary from 1 to p2. In the second case i is an overall "visiting index" which varies from 1 through Abs [p1]*p2.

The module returns $\{\{\zeta_1, \zeta_2, \zeta_3, \xi\}, w\}$ where $\{\zeta_1, \zeta_2, \zeta_3\}$ are the triangular coordinates over the cross section, and ξ the natural line coordinate in the longitudinal direction.

If the rule argument does not match a pair of implemented rules, the module returns {{Null, Null, Null, Null}, 0}.

```
WedgGaussRuleInfo[{rule_,numer_},point_]:= Module[
 {ζ1,ζ2,ζ3,ξ,p1,p2,i,j,k,m,w1,w2,
  info={{Null,Null,Null,Null},0}}, {p1,p2}=rule;
 If [Length[point]==2, {i,j}=point, k=point;
     m=Abs[p1]; j=Floor[(k-1)/m]+1; i=k-m*(j-1)];
 {{ζ1,ζ2,ζ3},w1}= TrigGaussRuleInfo[{p1,numer},i];
  {ξ,w2}=          LineGaussRuleInfo[{p2,numer},j];
 info={{ζ1,ζ2,ζ3,ξ},w1*w2};
 If [numer, Return[N[info]],Return[Simplify[info]]];
 ];
```

Figure 12.
Wedge integration rule information module

8. Pyramids

Unlike the other regions, there is scant information as regards closed-form, low-order integration rules for pyramids. For this reason this geometry is treated in more detail than the others.

8.1 Pyramid geometry

Pyramid solid elements are useful as transitions between bricks and tetrahedra in automated 3D mesh generation Figure 13 shows three configurations useful for such objective:

Pyra5. A 5-node pyramid element useful as transition between Hexa8 and Tetr4.

Pyra13. A 13-node pyramid element obtained from Pyra5 by adding eight midside nodes. Useful as transition between Hexa20 (serendipity brick) and Tetr10.

Pyra14. A 14-node pyramid element derived from Pyra13 by injecting a node at the center of the quadrilateral face. Useful as transition between Hexa27 (Lagrangian brick) and Tetr10.

The geometry of a pyramid is shown in Figure 14. The region has five corners, eight edges and five faces. One of the faces is a quadrilateral, called the base, which may be warped. The corner opposite to the base is the apex. Four triangular faces, called apex faces, meet at the apex. The apex faces are planar in Pyra5 but may be warped in Pyra13 and Pyra14. By analogy to pyramidal monuments, the normal distance from the quadrilateral base to the apex is called the height. The line joining the base center with the apex is the apex axis. This apex direction is not generally normal to the base. The shape functions for Pyra5 are:

$$N_1 = \frac{1}{8}(1 - \xi)(1 - \eta)(1 + \mu), \quad N_2 = \frac{1}{8}(1 + \xi)(1 - \eta)(1 + \mu),$$

$$N_3 = \frac{1}{8}(1 + \xi)(1 + \eta)(1 + \mu), \quad N_4 = \frac{1}{8}(1 - \xi)(1 + \eta)(1 + \mu),$$

$$\text{and } N_5 = \frac{1}{2}(1 - \mu).$$

The reference pyramid (RP) has a flat rectangular base with side lengths a and b, and apex direction normal to the base with height h. The Jacobian matrix and Jacobian determinant of a RP are:
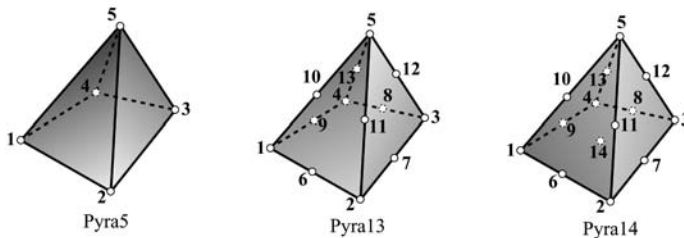
Figure 13.
Three useful nodal configurations of pyramid elements

$$\mathbf{J} = \begin{bmatrix} \frac{1}{4}a(1-\mu) & 0 & 0 \\ 0 & \frac{1}{4}b(1-\mu) & 0 \\ -\frac{1}{4}a\xi & -\frac{1}{4}b\eta & \frac{1}{2}h \end{bmatrix}, \tag{8}$$

$$J = \det\mathbf{J} = \frac{1}{32}abh(1-\mu)^2.$$

Unlike the other reference regions, J varies. It depends quadratically on $1-\mu$. At $\mu = 1, J = 0$. Consequently the inverse Jacobian, which appears in the computation of Cartesian derivatives of shape functions, is undefined at the apex.

8.2 Integration rules
The pyramid is the only region for which information on two types of rules is provided

Conical Product Rules, or CPR. These are tensor products of standard quadrilateral rules along the $\{\xi, \eta\}$ directions at sections $\mu = \text{const}$, with one-dimensional Gauss rules having kernel $(1 - \mu)2$ along the apex direction. The latter appear as ingredient of rules for general 3D cone-shaped regions; thus the name. A CPR is said to be isotropic if the number of points in the ξ, η and μ directions is the same, and anisotropic otherwise.

Non-product rules. These comply with the symmetry requirements of Table IV but cannot be classified as CPR. They are listed here since they are believed to be new. Except for the 5- and 13-point rules, their usefulness for FEM work is still unclear.

Table IX lists the nine pyramid integration rules implemented here. Four of them are CPR (one is anisotropic) whereas five are of non-product type. Two definitions of degree are used in that table. The ordinary definition agrees with that given in Section 2.5: a formula has degree d if it integrates exactly all monomials of the form $\xi^i \eta^j \mu^k, i + j + k \leq d$, over a RP. A formula has degree $\bar{d}$ if it integrates exactly all monomials of the form $\xi^i \eta^j \mu^k$, with $\min(i+j, k+2) \leq \bar{d}$ over a RP. The second definition accounts better for the fact that J varies quadratically in μ over a reference pyramid.

Module `PyraGaussRuleInfo`, shown in Figures 15 and 16, implements the nine rules of Table IX. The module is invoked as:

$$\{\{\texttt{xi,eta,mu}\},\texttt{w}\} = \texttt{PyraGaussRuleInfo}[\{\texttt{rule,numer}\}, \texttt{i}] \tag{9}$$
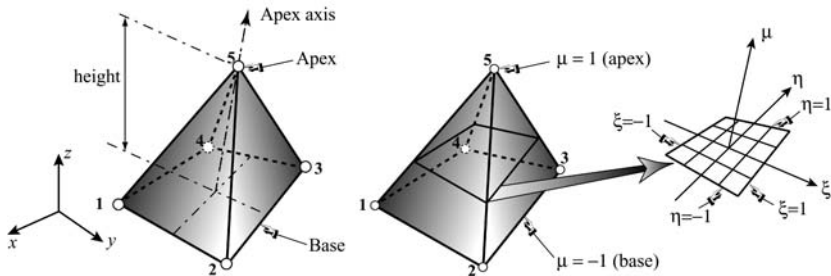
Figure 14.
Geometric information for
pyramid element

Ident	Points	Degree[a]	Comments
1	1	1,1	Isotropic CPR: gives correct RP volume
5	5	2,3	Useful for Pyra5 stiffness and mass
6	6	2,3	
8	8	3,3	Isotropic CPR: useful for Pyra13 and Pyra14 stiffness
−8	8	2,3	
9	9	2,3	
13	13	2,3	Useful for Pyra13 and Pyra14 mass
18	18	3,3	Anisotropic CPR; ditto
27	27	5,5	Isotropic CPR

Note: [a]Degree given as pair d, $\bar{d}$. Refer text for definitions

Argument `rule`, which can be 1, 5, 6, 8, − 8, 9, 13, 18 or 27, designates the integration formula as follows. `Abs[rule]` is the number of sample points. If there are two useful rules with the same number of points, the most accurate one is identified with a positive value and the other one with a minus value. For the pyramid region this only happens for the 8-point rules. If `rule = 8` an isotropic CPR formula is chosen. If `rule = − 8` a non-product formula is picked.

If the number of points is 1 through 18, `numer` is a logical flag specifying floating-point information if `True`, and exact information if `False`. For the 27-point rule see below.

```
PyraGaussRuleInfo[{rule_,numer_},point_]:= Module[
    {g1,g2,g3,g4,g5,w1,w2,w3,w4,jk={{-1,-1},{1,-1},{1,1},{-1,1}},
    jk4={{-1,0},{1,0},{0,-1},{0,1}},jk9={{-1,-1},{0,-1},{1,-1},
        {-1,0},{0,0},{-1,1},{0,1},{1,1}},
    wg9={64,40,25}/81,eps=10^(-16),info={{Null,Null,Null},0},
    j,k,m,p=rule,i=point},
    If [p==1, info={{0,0,-1/2},128/27}];
    If [p==5, g1=8*Sqrt[2/15]/5;
        If [i<5, {j,k}=jk[[i]]; info={{j*g1,k*g1,-2/3},81/100}];
        If [i==5, info={{0,0,2/5},125/27}]];
    If [p==6, g1=Sqrt[12/35]; g2={1/6,1/2}; w2={576/625,64/15};
        If [i<5, {j,k}=jk[[i]]; info={{j*g1,k*g1,-2/3},504/625}];
        If [i>4, info={{0,0,g2[[i-4]]},w2[[i-4]]}]];
    If [p==8,  g1=Sqrt[1/3]; g2=(2*Sqrt[10]-5)/15; g3=-2/3-g2;
        w1=5*(68+5*Sqrt[10])/432; w2=85/54-w1;
        If [i<5, {j,k}=jk[[i]];   info={{j*g1,k*g1,g2},w1}];
        If [i>4, {j,k}=jk[[i-4]]; info={{j*g1,k*g1,g3},w2}]];
    If [p==-8,
        g1=Sqrt[(2/15)*(573-2*Sqrt[51])]/15;
        g2=Sqrt[(2/15)*(573+2*Sqrt[51])]/15;
        g3=-(2*Sqrt[51]+13)/35; g4=(2*Sqrt[51]-13)/35;
        w1=(11764-461*Sqrt[51])/15300; w2=346/225-w1;
        If [i<5, {j,k}=jk[[i]];   info={{j*g1,k*g1,g3},w1}];
        If [i>4, {j,k}=jk[[i-4]]; info={{j*g2,k*g2,g4},w2}]];
    If [p==9,
        g1=8*Sqrt[(573+5*Sqrt[2865])/(109825+969*Sqrt[2865])];
        g2=Sqrt[(2*(8025+Sqrt[2865]))/35]/37;
        g3=-(87+Sqrt[2865])/168; g4=(-87+Sqrt[2865])/168;
        w1=7*(11472415-70057*Sqrt[2865])/130739500;w2=84091/68450-w1;
        If [i<5,        {j,k}=jk[[i]];    info={{j*g1,k*g1,g3},w1}];
        If [i>4&&i<9, {j,k}=jk[[i-4]]; info={{j*g2,k*g2,g4},w2}];
        If [i==9,     info={{0,0,2/3},18/5}]];
```

Figure 15.
Pyramid integration
rule information module:
part 1 of 2

Figure 16.
Pyramid integration
rule information module:
part 2 of 2

```
If [p==13,
    g1=7*Sqrt[35/59]/8; g2=224*Sqrt[336633710/33088740423]/37;
    g3=Sqrt[37043/35]/56; g4=-127/153; g5=1490761/2842826;
    w1=170569/331200; w2=276710106577408/1075923777052725;
    w3=12827693806929/30577384040000;
    w4=1066338334065507064354192/431017052887936519704375;
    If [i<5,     {j,k}=jk[[i]];    info={{j*g1,k*g1,-1/7},w1}];
    If [i>4&&i<9, {j,k}=jk4[[i-4]];info={{j*g2,k*g2,-9/28},w2}];
    If [i>8&&i<13,{j,k}=jk[[i-8]]; info={{j*g3,k*g3,g4},w3}];
    If [i==13,   info={{0,0,g5},w4}]];
If [p==18, g1=Sqrt[3/5]; g2=1-2*(10-Sqrt[10])/15;
    g3=-2/3-g2; w1=5*(68+5*Sqrt[10])/432; w2=85/54-w1;
    If [i<10,    {j,k}=jk9[[i]];    m=Abs[j]+Abs[k];
       info={{j*g1,k*g1,g2},w1*wg9[[m+1]] }];
    If [i>9&&i<19, {j,k}=jk9[[i-9]]; m=Abs[j]+Abs[k];
       info={{j*g1,k*g1,g3},w2*wg9[[m+1]] }]];
If [p==27, g1=Sqrt[3/5];
    g3=-0.85401195185370053568832404197599 3416;
    g4=-0.30599246792329623055564729131921 03090;
    g5= 0.41000441977699676624479695516809 6505;
    If [!numer, {g3,g4,g5}=Rationalize[{g3,g4,g5},eps]];
    w1=(4/15)*(4+5*(g4+g5)+10*g4*g5)/((g3-g4)*(g3-g5)*(1-g3)^2);
    w2=(4/15)*(4+5*(g3+g5)+10*g3*g5)/((g3-g4)*(g5-g4)*(1-g4)^2);
    w3=(4/15)*(4+5*(g3+g4)+10*g3*g4)/((g3-g5)*(g4-g5)*(1-g5)^2);
    If [i<10,    {j,k}=jk9[[i]];    m=Abs[j]+Abs[k];
       info={{j*g1,k*g1,g3},w1*wg9[[m+1]]}];
    If [i>9&&i<19, {j,k}=jk9[[i-9]]; m=Abs[j]+Abs[k];
       info={{j*g1,k*g1,g4},w2*wg9[[m+1]]}];
    If [i>18,    {j,k}=jk9[[i-18]];m=Abs[j]+Abs[k];
       info={{j*g1,k*g1,g5},w3*wg9[[m+1]]}]];
If [numer, Return[N[info]], Return[info]];
];
```

Argument i is the index of the sample point, which can vary from 1 through Abs
[rule].

If rule = 27, exact information for the μ coordinates of the abscissas is not given.
These are roots of a cubic Jacobi polynomial in μ, cf. Table 25.8 of Abramowitz and
Stegun (1964), with three real roots; thus the exact expressions contain cubic roots of
complex numbers. Such forms are difficult to simplify in symbolic work. Instead the μ
abscissas are listed in floating-point form with 36 exact digits. If flag numer is
False, those abscissas are converted to rational numbers that represent the data to 16
place accuracy, using the built-in function Rationalize. Weights are recovered
from the abscissas. The conversion precision can be changed by adjusting the value of
internal variable eps, which is set in the module preamble.

8.3 Application example

The example illustrates the computation of the lumped mass matrix of a reference
Pyra5 element using the HRZ mass lumping scheme (Hinton et al., 1976). The symbolic
computation of the lumped mass is implemented as shown in Figure 17. Five rules are
placed in a For loop: 1, 5, 8, 18 and 27 points. The lumped mass for the three
freedoms of the apex node is $\gamma_a \rho abh$ while for the four base nodes is $\gamma_b \rho abh$. Here ρ is
the mass density whereas γ_a, γ_b are dimensionless coefficients to be determined. As an
implementation check, $\gamma_a + 4\gamma_b = 1/3$ because the total mass of the reference pyramid
is $\frac{1}{3}\rho abh$. The results are collected in Table X. The rules with 8, 18 and 27 points give

```
IsoPPyra5LumpedMass[{a_,b_,h_},r_,rule_]:= Module[{k,x,h,m,c,w,Jdet,
  N1,N2,N3,N4,N5,Ne,sx,sy,sz,Vol=0,MeC=Table[0,{15},{15}],MeD},
  For [k=1, k<=Abs[rule], k++,
      {{x,h,m},w}=PyraGaussRuleInfo[{rule,False},k];
       {N1,N2,N3,N4,N5}=
       {(1-x)*(1-h)*(1+m)/8, (1+x)*(1-h)*(1+m)/8,
        (1+x)*(1+h)*(1+m)/8, (1-x)*(1+h)*(1+m)/8,(1+m)/2};
       Jdet=a*b*h*(1-m)^2/32; c=w*Jdet;
       Ne={{N1,0,0,N2,0,0,N3,0,0,N4,0,0,N5,0,0},
           {0,N1,0,0,N2,0,0,N3,0,0,N4,0,0,N5,0},
           {0,0,N1,0,0,N2,0,0,N3,0,0,N4,0,0,N5}};
       MeC+=c*r*Transpose[Ne].Ne; Vol+=c]; MeC=Simplify[MeC];
     MeD=Table[MeC[[i,i]],{i,15}]; {sx,sy,sz}={0,0,0};
     For [i=1,i<=13,i+=3,
         {sx,sy,sz}+={MeD[[i]],MeD[[i+1]],MeD[[i+2]]}/r];
     For [i=1,i<=13,i+=3,MeD[[i]]*=(Vol/sx);MeD[[i+1]]*=(Vol/sy);
         MeD[[i+2]]*=(Vol/sz)]; Return[Simplify[MeD]]
  ];
ClearAll[a,b,h,r];
For [ip=1,ip<=5,ip++, p={1,5,8,18,27}[[ip]];
  MeD=IsoPPyra5LumpedMass[{a,b,h},r,p];
  sMeD=Simplify[MeD/(a*b*h*r)];
  Print["Rule=",p," Lumped Mass=a*b*h*r",sMeD//InputForm];
  Print["check=",Sum[sMeD[[i]],{i,1,15,3}]]
  ];
```

Source: Hinton *et al.* (1976)

Figure 17.
Module to form
lumped mass matrix of
reference Pyra5 element
using the HRZ scheme

Rule	γ_a	γ_b	Check: $\gamma_a + 4\gamma_b$
1	4/15	1/60	1/3
5	12500/48631	11131/583572	1/3
8	3/13	1/39	1/3
18	3/13	1/39	1/3
27	3/13	1/39	1/3

Table X.
Results from the lumped
mass Pyra5 analysis

$\alpha_a = 3/13$, which means that (9/13)th of the total mass (roughly 70 per cent) goes to the apex node. Analytical integration, which is possible for the reference pyramid, gives the same answer.

For a Pyra13 or Pyra14 element, a similar process only reaches exactness at 27 integration points.

9. Hexahedra

Only rules of product type are implemented for hexahedra. As in the case of quadrilaterals, symmetric non-product rules can be used to speed up high order element formation in numerical FEM codes. These are omitted here to keep the logic simple, and to facilitate production of anisotropic rules.

Product rules are obtained by applying the one-dimensional rules to each of the three natural coordinate directions: ξ, η and μ, in turn. They are implemented in the Mathematica module HexaGaussRuleInfo shown in Figure 18 as tensor products of one-dimensional rules with 1-5 points. HexaGaussRuleInfo calls LineGaussRuleInfo thrice. If the number of points along ξ, η and μ is the same, the rule is called isotropic, and anisotropic otherwise. Anisotropic rules are important in thick shell elements constructed by degenerating hexahedra along a thickness direction.

The five isotropic rules available from HexaGaussRuleInfo are listed in Table XI. For isotropic rules the module is invoked in either of two ways:

$$\{\{xi,eta,mu\},w\} = \text{HexaGaussRuleInfo}[\{p,numer\}, \{i,j,k\}]$$

$$\{\{xi,eta,mu\},w\} = \text{HexaGaussRuleInfo}[\{p,numer\}, m] \tag{10}$$

The first form is used to get information for point $\{i, j, k\}$ of the $p \times p \times p$ rule, in which $1 \leq i \leq p$, $1 \leq j \leq p$ and $1 \leq k \leq p$. The second form specifies that point by a "visiting counter" m that runs from 1 through p^3; if so i, j and k are internally extracted from m.

If the integration rule has p_1, p_2 and p_3 points in the ξ, η and μ directions, respectively, the module may be also invoked in two ways:

$$\{\{xi,eta,mu\},w\} = \text{HexaGaussRuleInfo}[\{\{p1,p2,p3\},numer\},$$

$$\{i,j,k\}]\{\{xi,eta,mu\},w\} = \text{HexaGaussRuleInfo} \tag{11}$$

$$[\{\{p1,p2,p3\},numer\}, m]$$

In the first form i, j and k run from 1 to p_1, p_2 and p_3, respectively. In the second form m runs from 1 to $p_1 p_2 p_3$; if so i, j and k are extracted from m as shown in the module logic.

In all four forms flag numer is set to True if floating point numerical information is desired and to False if exact information is desired.

The module returns ξ_i, η_j and μ_k in xi, eta, and mu, respectively, and the weight product $w_i w_j w_k$ in w. If the number of points is outside the range of the implementation, the module returns {{Null, Null, Null}, 0}.

```
HexaGaussRuleInfo[{rule_,numer_},point_]:= Module[
 {x,h,m,p1,p2,p3,p12,i,j,jj,k,m,w1,w2,w3,
  info={{Null,Null,Null},0}},
  If [Length[rule]==3,  {p1,p2,p3}=rule,  p1=p2=p3=rule];
  If [Length[point]==3, {i,j,k}=point,    m=point;
    p12=p1*p2; k=Floor[(m-1)/p12]+1; jj=m-p12*(k-1);
    j=Floor[(jj-1)/p1]+1;  i=jj-p1*(j-1)];
  {x,w1}= LineGaussRuleInfo[{p1,numer},i];
  {h,w2}= LineGaussRuleInfo[{p2,numer},j];
  {m,w3}= LineGaussRuleInfo[{p3,numer},k];
  info={{x,h,m},w1*w2*w3};
  If [numer, Return[N[info]], Return[Simplify[info]]];
 ];
```

Figure 18.
Hexahedron integration rule information module

Ident	Product rule	Points	Degree	Comments
1	$1 \times 1 \times 1$	1	1	Useful in reduced and selective integration
2	$2 \times 2 \times 2$	8	3	Useful for Hexa8 stiffness and mass
3	$3 \times 3 \times 3$	27	5	Useful for Hexa20 and Hexa27 stiffness and mass
4	$4 \times 4 \times 4$	64	7	Rarely used
5	$5 \times 5 \times 5$	125	9	Rarely used

Note: Anisotropic rules, such as $2 \times 1 \times 3$, are requested using a three-entry identifier

Table XI.
Isotropic Gauss-product formulas for hexahedra

10. Conclusions

The present compilation is selective and far from exhaustive gaps and omissions are noted in Section 2.2. The hierarchical configuration, however, supports extendibility in the sense that adding more rules to individual modules (or additional modules for other regions) is not a major undertaking. The organization also facilitates selective low-level code generation using, for example, the `//CForm or //FortranForm` output filters of Mathematica.

The contribution in terms of new results includes the following.

(1) Exact expressions are supplied for several rules only available numerically in the literature. These tend to be of moderate degree, typically 3, 4 or 5. Exact forms for higher degree rules may be either unavailable or too unwieldy for CAS simplification. For those a rationalization procedure of controllable accuracy (which can reach up to 36 places) is provided, although this technique should be used with care since fraction components may explode.

(2) The non-product formulas for the pyramid are believed to be new. Of these, the 5-point and 13-point rules appear to be useful for FEM work.

(3) Automatic production of anisotropic rules for quadrilaterals, wedges and hexahedra.

One useful extension to this collection may be integration rules for more general polyhedral regions. These arise in automatic 3D "Delauney polyhedrization" in support of meshfree Lagrangian methods.

References

Abramowitz, M. and Stegun, L.A. (Eds) (1964), *Handbook of Mathematical Functions with Formulas, Graphs and Mathematical Tables*, Applied Mathematics Series 55, Natl. Bur. Standards, US Department of Commerce, Washington, DC.

Bergan, P.G. and Felippa, C.A. (1985), "A triangular membrane element with rotational degrees of freedom", *Comp. Meths. Appl. Mech. Eng.*, Vol. 50, pp. 25-69.

Cools, R. (1999a), "Constructing cubature formulas – the science behind the art", *Acta Numerica*, Cambridge University Press, Cambridge, Vol. 6, pp. 1-54.

Cools, R. (1999b), "Monomial cubature rule science 'Stroud': a compilation - part 2", *J. Comput. Appl. Math.*, Vol. 112, pp. 21-7.

Cools, R. (2003), "An encyclopedia of cubature formulas", *J. Complexity*, Vol. 19, pp. 445-53.

Felippa, C.A. (1966), "Refined finite element analysis of linear and nonlinear two-dimensional structures", PhD dissertation, Department of Civil Engineering, University of California at Berkeley, Berkeley, CA.

Felippa, C.A. (2000), "Recent advances in finite element templates", in Topping, B.H.V. (Ed.), *Computational Mechanics for the Twenty-First Century*, Saxe-Coburn Publications, Edinburgh, pp. 71-98.

Felippa, C.A. (2003), "A study of optimal membrane triangles with drilling freedoms", *Comp. Meths. Appl. Mech. Eng.*, Vol. 192, pp. 2125-68.

Felippa, C.A. and Bergan, P.G. (1987), "A triangular plate bending element based on an energy-orthogonal free formulation", *Comp. Meths. Appl. Mech. Eng.*, Vol. 61, pp. 129-60.

Hammer, P.C. and Stroud, A.H. (1956), "Numerical integration over simplices", *Math. Tables Aids Comput.*, Vol. 10, pp. 137-9.

Hammer, P.C. and Stroud, A.H. (1958), "Numerical evaluation of multiple integrals", *Math. Tables Aids Comput.*, Vol. 12, pp. 272-80.

Hinton, E., Rock, T. and Zienkiewicz, O.C. (1976), "A note on mass lumping and related processes in the finite element method", *Earthquake Eng. Struct. Dynamics*, Vol. 4, pp. 245-9.

Irons, B.M. (1966), "Engineering applications of numerical integration in stiffness methods", *AIAAJ*, Vol. 4, pp. 2035-7.

Irons, B.M. (1971), "Quadrature rules for brick-based finite elements", *Int. J. Numer. Meth. Eng.*, Vol. 3, pp. 293-4.

Irons, B.M. and Ahmad, S. (1980), *Techniques of Finite Elements*, Ellis Horwood Ltd, Chichester.

Strang, G. and Fix, G. (1973), *An Analysis of the Finite Element Method*, Prentice-Hall, Englewood Cliffs, NJ.

Stroud, A.H. (1971), *Approximate Calculation of Multiple Integrals*, Prentice-Hall, Englewood Cliffs, NJ.

Taig, I.C. and Kerr, R.I. (1964), "Some problems in the discrete element representation of aircraft structures", in Fraeijs de Veubke, B.M. (Ed.), *Matrix Methods of Structural Analysis*, Pergamon Press, London, pp. 267-315.

Zienkiewicz, O.C. (1971), *The Finite Element Method in Engineering Science*, 2nd ed., (3rd ed., 1977), McGraw-Hill, New York, NY.