On the Identification of Symmetric Quadrature Rules for Finite Element Methods

F.D. Witherden
Department of Aeronautics & Astronautics, Stanford University
Outline

• Presentation is based on the paper “On the identification of symmetric quadrature rules for finite element methods”
Motivation

• The **finite element method** is one of the **backbones** of modern science and engineering.
Motivation

• During the assembly process it is necessary to evaluate integrals inside of various 2D and 3D domains.

• This is accomplished via numerical quadrature schemes.
Motivation

• We therefore seek quadrature rules which are both accurate and efficient.
Motivation

• However, there also exists a deep connection between symmetric quadrature rules and...

• summation-by-parts (SBP) operators,

• multivariate Lagrange interpolation.
Quadrature

- Quadrature is concerned with the numerical evaluation of integrals.

\[ \int_{a}^{b} f(x) \, dx \]
Interpolatory Quadrature

- Most quadrature schemes are based around approximating the function $f(x)$ by a polynomial $p(x)$ such that

$$\int_a^b f(x) \, dx \approx \int_a^b p(x) \, dx.$$
Interpolatory Quadrature

- Consider sampling $f(x)$ at a set of $m + 1$ abscissa \( \{x_0, x_1, \ldots, x_m\} \) and constructing a Lagrange interpolating polynomial as

$$
  p(x) = \sum_{i=0}^{m} l_i(x) f(x_i), \quad l_i(x_j) = \delta_{ij}.
$$
Interpolatory Quadrature

- Hence

\[ \int_{a}^{b} f(x) \, dx \approx \sum_{i=0}^{m} \omega_i f(x_i), \]

\[ \omega_i = \int_{a}^{b} \ell_i(x) \, dx. \]
Interpolatory Quadrature

• If \( f(x) \) is a polynomial of degree \( m \) or below then the **quadrature is exact** for any choice of abscissa.

• The question is what to do in the case where \( f(x) \) is of higher degree or non-polynomial.
Interpolatory Quadrature

- There are several schools of thought.
- Most prevalent is to choose the abscissa to **maximise the strength** of the quadrature rule.
Interpolatory Quadrature

• **Theorem.** The maximum strength of an $m + 1$ abscissa rule is $2m + 1$ and is achieved by taking the abscissa to be the roots of the Legendre polynomial $P_{m+1}(x)$

\[
\int_{-1}^{1} P_i(x) P_j(x) \, dx = h_m \delta_{ij}
\]
Gaussian Quadrature

• *Proof*. Let \( f(x) \) be a polynomial of degree \( 2m + 1 \) or less.

• Using the polynomial remainder theorem we have

\[
f(x) = q(x)P_{m+1}(x) + r(x)
\]

where \( q(x) \) and \( r(x) \) are of degree \( m \) or less.
Gaussian Quadrature

\[ f(x) = q(x)P_{m+1}(x) + r(x) \]

- Integrating we see that

\[ \int_{-1}^{1} f(x) \, dx = \int_{-1}^{1} q(x)P_{m+1}(x) \, dx + \int_{-1}^{1} r(x) \, dx \]
Gaussian Quadrature

\[ f(x) = q(x)P_{m+1}(x) + r(x) \]

- Applying quadrature we find

\[
\sum_{i=0}^{m} \omega_i f(x_i) = \sum_{i=0}^{m} \omega_i \left[ q(x_i)P_{m+1}(x_i) + r(x_i) \right]
\]
Gaussian Quadrature

• Since $r(x)$ is of degree $< m$

\[
\int_{-1}^{1} f(x) \, dx = \sum_{i=0}^{m} \omega_i f(x_i)
\]
Consider a cubic polynomial $p(x)$ between $[-1, 1]$.

The area is fixed by $p(1/\sqrt{3})$ and $p(-1/\sqrt{3})$. 

Gaussian Quadrature
Gaussian Quadrature

• Are the roots of $P_{m+1}(x)$ necessarily real?
• Are they always between -1 and 1?
• Are the weights always positive?
Gaussian Quadrature

• The answer to all three is yes.
Gaussian Quadrature

• Classically, the abscissa are computed using the Golub–Welsch algorithm at a cost of $O(m^2)$.

• Recent developments have enabled the abscissa to be determined in $O(m)$ time.
Cubature

- The extension of quadrature to multiple integrals is oft referred to as cubature.
Cubature

• The extension of quadrature to multiple integrals is oft referred to as cubature.
Cubature

- Extension to **quads and hexes** possible through a **tensor-product construction** of a one-dimensional rule.

- With \((m + 1)^2\) points can integrate all monomials \(x^i y^j\) where \(i, j \leq 2m + 1\).
Cubature

• For other shapes one can employ a **Duffy transform** to map them onto a quad or hex.
Cubature

• For other shapes one can employ a **Duffy transform** to map them onto a quad or hex.

\[
\int_{-1}^{1} \int_{-1}^{-y} f(x, y) \, dx \, dy = \int_{-1}^{1} \int_{-1}^{1} f(x, y) |J| \, d\tilde{x} \, d\tilde{y}
\]
• Such rules are functional…

• …but **inefficient**.

• Also suffer from an undesirable **clustering of abscissa**.
Economical Cubature Rules

- Rules designed specifically for integrating functions inside of a given element are termed economical.

- Have the potential to greatly reduce the number of required abscissa to integrate $f(x)$. 
Economical Cubature Rules

- Can view as a non-linear least squares problem for the unknowns \( \{x_1, \omega_1, \ldots, x_n, \omega_n\} \) where we desire

\[
\int_{\Omega} p_i(x) \, dx = \sum_{j=1}^{n} \omega_j p_i(x_j) \quad \text{for } 1 \leq i \leq m
\]
Economical Cubature Rules

• Although this approach works “as is” it is prone to failure and often results in poor quality rules.
Improvement #1: Weights

• If the abscissa are known then the system reduces to a linear system of dimension \( m \times n \) for the weights.

\[
\int_{\Omega} p_i(x) \, dx = \sum_{j=1}^{n} \omega_j p_i(x_j) \quad \text{for } 1 \leq i \leq m
\]

• This is simply a **linear least squares problem** which we may solve directly.
Improvement #1: Weights

• Thus, by treating the weights as dependent variables we may **halve the number of non-linear unknowns**.

• Further, by using **non-negative linear least squares** we can trivially enforce the requirement that $\omega_i > 0$. 
Improvement #2: Symmetry

• Many shapes have symmetries.

• Desirable for these to be displayed by the quadrature rule.

• Can accomplish this via symmetry orbits.
Improvement #2: Symmetry

• Many shapes have symmetries.

• Desirable for these to be displayed by the quadrature rule.

• Can accomplish this via symmetry orbits.
Improvement #2: Symmetry

• For example, a triangle has six symmetries which can be represented by three orbits.
Improvement #2: Symmetry

• For example, a triangle has six symmetries which can be represented by three orbits.
Improvement #2: Symmetry

• Given a desired number of points \( n \) there are usually multiple different orbital configurations.

• However, sometimes there are no solutions;

• for example a triangle with \( n = 44 \).
Improvement #2: Symmetry

• Respecting symmetry not only results in better rules but it also substantially reduces the number of unknowns in the non-linear problem.
Improvement #2: Symmetry

- Moreover it also enables us to greatly reduce the number of basis functions we need to test.

- For example, in one dimension we have

\[ \int_{-1}^{1} x^i \, dx = 0 \text{ for } i \text{ odd} \]

which is satisfied by all symmetric abscissa.
Improvement #2: Symmetry

- For example, consider $m = 10$.

<table>
<thead>
<tr>
<th>Shape</th>
<th>Without symmetry</th>
<th>With symmetry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Triangle</td>
<td>66</td>
<td>36</td>
</tr>
<tr>
<td>Quadrangle</td>
<td>66</td>
<td>12</td>
</tr>
<tr>
<td>Hexahedron</td>
<td>286</td>
<td>16</td>
</tr>
<tr>
<td>Prism</td>
<td>286</td>
<td>91</td>
</tr>
<tr>
<td>Pyramid</td>
<td>286</td>
<td>56</td>
</tr>
<tr>
<td>Tetrahedron</td>
<td>286</td>
<td>67</td>
</tr>
</tbody>
</table>
Improvement #3: Conditioning

- Using a **monomial basis** such that $p_k(x) = x^i y^j$ results in an extremely **ill-conditioned** problem that places **undue weight on certain modes**.
Improvement #3: Conditioning

• We can fix this by changing to an orthonormal basis with

\[ \int_{\Omega} \psi_i(x) \psi_j(x) \, dx = \delta_{ij} \]

• Hence

\[ \int_{\Omega} \psi_i(x) \, dx = \sum_{j=1}^{n} \omega_j \psi_i(x_j) \quad \text{for} \ 1 \leq i \leq m \]
Improvement #4: Constraints

• Easiest way to ensure that the points remain inside the domain is to **clamp the orbital parameters**.

• Enables the use of simpler **unconstrained optimisation algorithms** such as Levenberg–Marquardt.
Algorithm

• Given a shape $\Omega$ a target order $m$ and a point count $n$…
  • for each orbital decomposition of $n$…
    • for $i = 1..\text{maximum attempt count}$…
      • randomly seed the orbits…
      • attempt to solve the non-linear least squares problem…
    • if the residual is zero then output the rule.
Rule Selection

• Typical to stop the process after having found a single rule with $n$ points of degree $m$.

• We however keep going and can thus identify multiple rules.

• Leads us to the concept of rule selection.
Rule Selection

• Given $N$ rules of degree $m$ with $n$ abscissa we can assess them by comparing how they perform integrating the basis functions of degree $m + 1$…

• …and prefer the rule with the smallest overall error.
Implementation

• Have implemented this approach in software package Polyquad.

• Available on GitHub and released under the GPL.
# New Rules

<table>
<thead>
<tr>
<th>$m$</th>
<th>Tri</th>
<th>Quad</th>
<th>Tet</th>
<th>Pri</th>
<th>Pyr</th>
<th>Hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>4</td>
<td>8</td>
<td>8</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>8</td>
<td>14</td>
<td>11</td>
<td>10</td>
<td>14</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>8</td>
<td>14</td>
<td>16</td>
<td>15</td>
<td>14</td>
</tr>
<tr>
<td>6</td>
<td>12</td>
<td>12</td>
<td>24</td>
<td>28</td>
<td>24</td>
<td>34</td>
</tr>
</tbody>
</table>
# New Rules

<table>
<thead>
<tr>
<th>$m$</th>
<th>Tri</th>
<th>Quad</th>
<th>Tet</th>
<th>Pri</th>
<th>Pyr</th>
<th>Hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>15</td>
<td>12</td>
<td>35</td>
<td>35</td>
<td>31</td>
<td>34</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
<td>20</td>
<td>46</td>
<td>46</td>
<td>47</td>
<td>58</td>
</tr>
<tr>
<td>9</td>
<td>19</td>
<td>20</td>
<td>59</td>
<td>60</td>
<td>62</td>
<td>58</td>
</tr>
<tr>
<td>10</td>
<td>25</td>
<td>28</td>
<td>81</td>
<td>85</td>
<td>83</td>
<td>90</td>
</tr>
<tr>
<td>11</td>
<td>28</td>
<td>28</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>33</td>
<td>37</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### New Rules

<table>
<thead>
<tr>
<th>$m$</th>
<th>Tri</th>
<th>Quad</th>
<th>Tet</th>
<th>Pri</th>
<th>Pyr</th>
<th>Hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>37</td>
<td>37</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>42</td>
<td>48</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>49</td>
<td>48</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>55</td>
<td>60</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>60</td>
<td>60</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>67</td>
<td>72</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## New Rules

<table>
<thead>
<tr>
<th>$m$</th>
<th>Tri</th>
<th>Quad</th>
<th>Tet</th>
<th>Pri</th>
<th>Pyr</th>
<th>Hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>49</td>
<td>48</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>55</td>
<td>60</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>60</td>
<td>60</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>67</td>
<td>72</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>73</td>
<td>72</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>79</td>
<td>85</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• Given a set of points \( \{x_1, x_2, \ldots, x_n\} \) can construct a multivariate Lagrange interpolation polynomial as

\[
\ell_i(x) = \sum_{k=1}^{n} V_{i k}^{-1} \Psi_k(x), \quad V_{ij} = \Psi_i(x_j)
\]

• We therefore require \( V \) to be non-singular.
Unisolvency

• In 1D we simply require the points to be unique.

• Consider a quad with two points and \( \det(V) \neq 0 \).

• Interchanging the two points will therefore flip the sign of \( \det(V) \).
Unisolvency

• In 1D we simply require the **points to be unique**.

• Consider a quad with **two points** and $\det(V) \neq 0$.

• Interchanging the two points will therefore flip the sign of $\det(V)$. 
Unisolvency

- Let us now consider moving the points continuously on separate paths.

- By the mean value theorem there is a location wherein $\det(V) = 0$...even though the points are necessarily distinct.
Unisolvency

- Remarkably, many good quadrature rules with relevant abscissa counts suffer from this issue.
- Thus the interpolation interpretation of quadrature breaks down in multiple dimensions.
Back To Interpolation

• In terms of the $L^2$ norm all of the best-known nodes do happen to correspond to the abscissa of quadrature rules.
Conclusions

• Have described an numerical algorithm for identifying numerical quadrature rules suitable for finite element methods.
Future Work

- Decomposition count increases rapidly with $n$.

<table>
<thead>
<tr>
<th>$n$</th>
<th>Tri</th>
<th>Quad</th>
<th>Tet</th>
<th>Pri</th>
<th>Pyr</th>
<th>Hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>—</td>
<td>12</td>
<td>3</td>
<td>35</td>
<td>34</td>
<td>2</td>
</tr>
<tr>
<td>40</td>
<td>7</td>
<td>36</td>
<td>13</td>
<td>260</td>
<td>161</td>
<td>6</td>
</tr>
<tr>
<td>80</td>
<td>—</td>
<td>121</td>
<td>50</td>
<td>2380</td>
<td>946</td>
<td>56</td>
</tr>
<tr>
<td>160</td>
<td>27</td>
<td>441</td>
<td>308</td>
<td>29330</td>
<td>6391</td>
<td>462</td>
</tr>
</tbody>
</table>
Future Work

• Approach of picking \((m,n)\) also does not scale to high \(m\).

• Therefore need to investigate ‘knockout’ type approaches where the optimal \(n\) is found by starting with a high \(n\) and then eliminating orbits of ‘least importance’.
Future Work

• Need to develop a better understanding of the complex relationship between $L^2$ optimal interpolation nodes and quadrature rules…

• …and ideally a **direct means** of identifying such nodes.