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**"

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa

On the identification of symmetric quadrature rules for finite element methods

F.D. Witherden *, P.E. Vincent

Department of Aeronautics, Imperial College London, SW7 2AZ, United Kingdom

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**.

$$\int_a^b f(x)\,\mathrm{d}x$$

Interpolatory Quadrature

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

$$\int_a^b f(x)\,\mathrm{d}x \approx \int_a^b p(x)\,\mathrm{d}x.$$

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

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

Interpolatory Quadrature

- Hence

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{i=0}^{m} \omega_i f(x_i),$$

$$\omega_i = \int_a^b \ell_i(x)\,\mathrm{d}x.$$

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 chose 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 $P_{m+1}(x)$

$$\int_{-1}^{1} P_i(x)P_j(x)\,\mathrm{d}x = h_m\delta_{ij}$$

Gaussian Quadrature

- *Proof*. Let f(x) be a polynomial of **degree 2*m* + 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

$$\int_{-1}^{1} f(x)\,\mathrm{d}x = \int_{-1}^{1} q(x)P_{m+1}(x)\,\mathrm{d}x + \int_{-1}^{1} r(x)\,\mathrm{d}x$$

Gaussian Quadrature

$$f(x) = q(x)P_{m+1}(x) + r(x)$$

- Applying quadrature we find

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

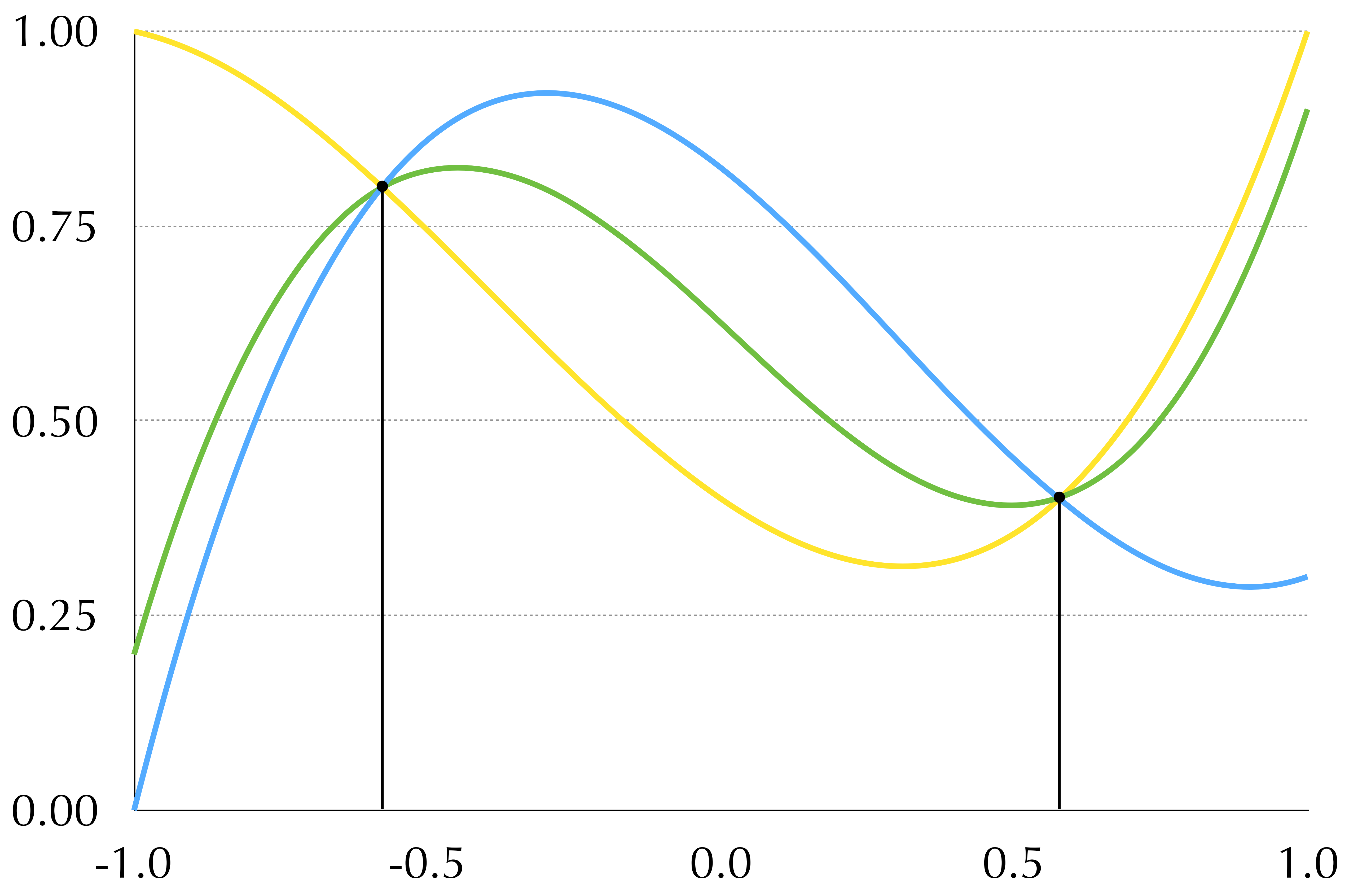
Gaussian Quadrature

- Since r(x) is of degree < m

$$\int_{-1}^{1} f(x)\,dx = \sum_{i=0}^{m} \omega_i f(x_i)$$

Gaussian Quadrature

- Consider a **cubic polynomial** p(*x*) between [-1, 1].
- The area is fixed by p(1/√3) and p(-1/√3).



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

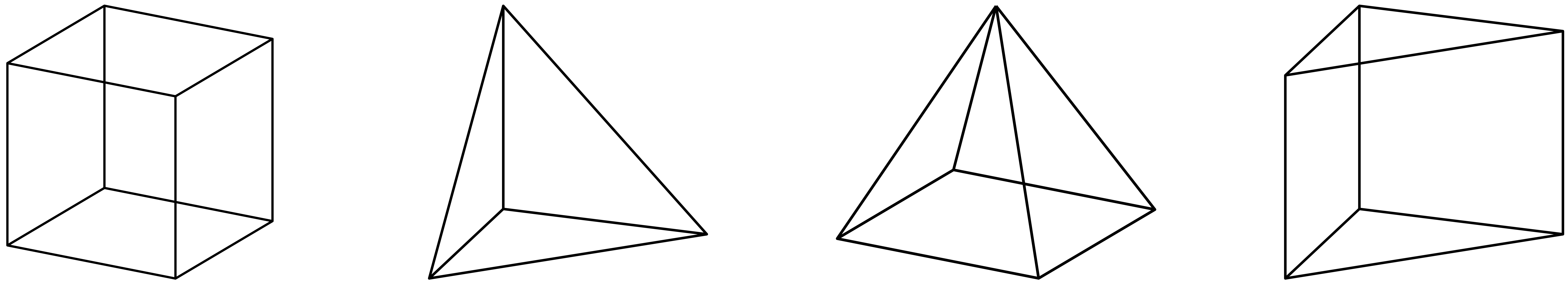
- Classically, the abscissa are computed using the **Golub–Welsch algorithm** at a cost of $O(m^2)$.
- Recent developments have enabled the abscissa to be determined in $O(m)$ time.

Cubature

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

Cubature

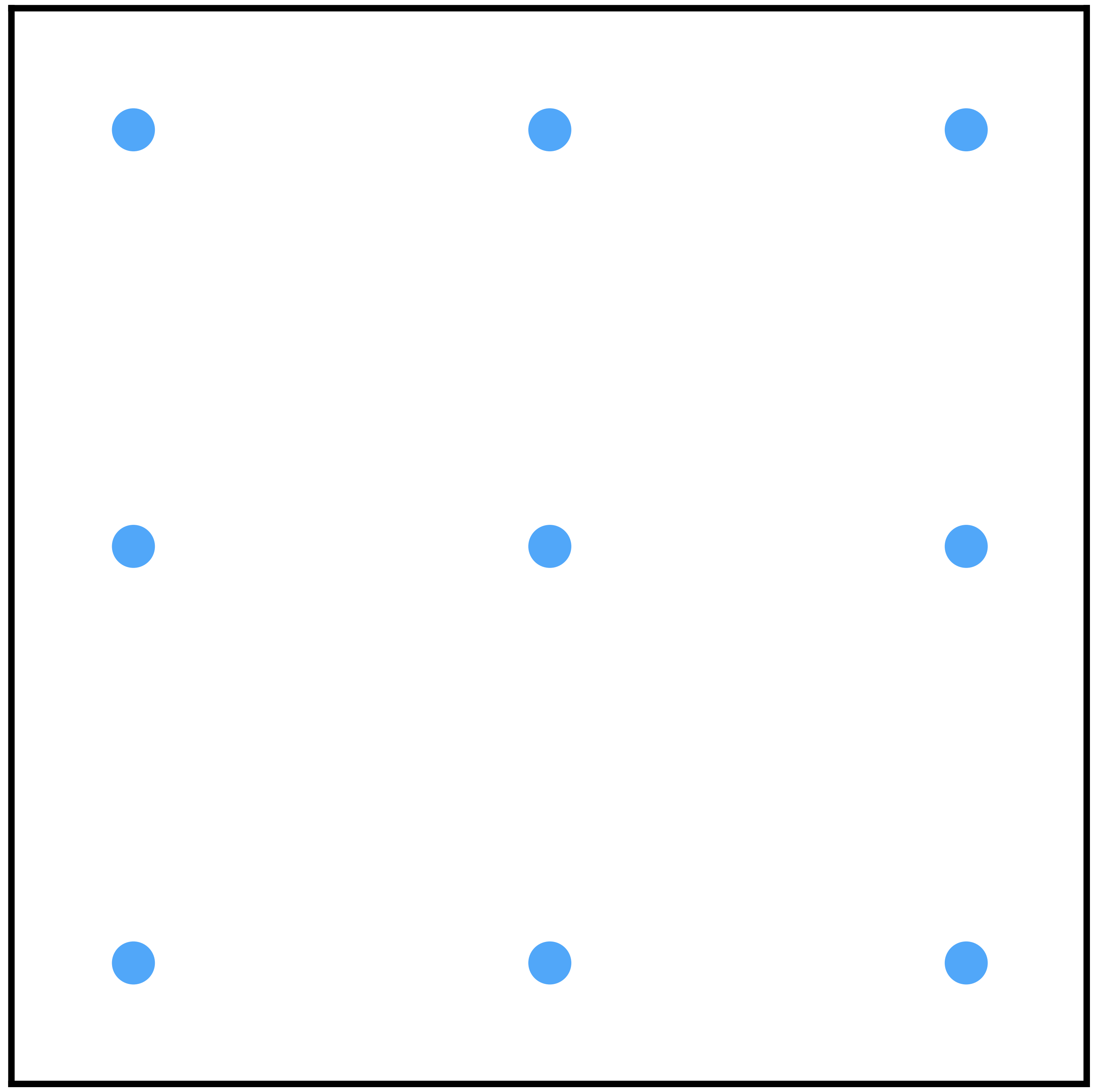
- The extension of quadrature to **multiple integrals** is oft referred to as **cubature**.



Cubature

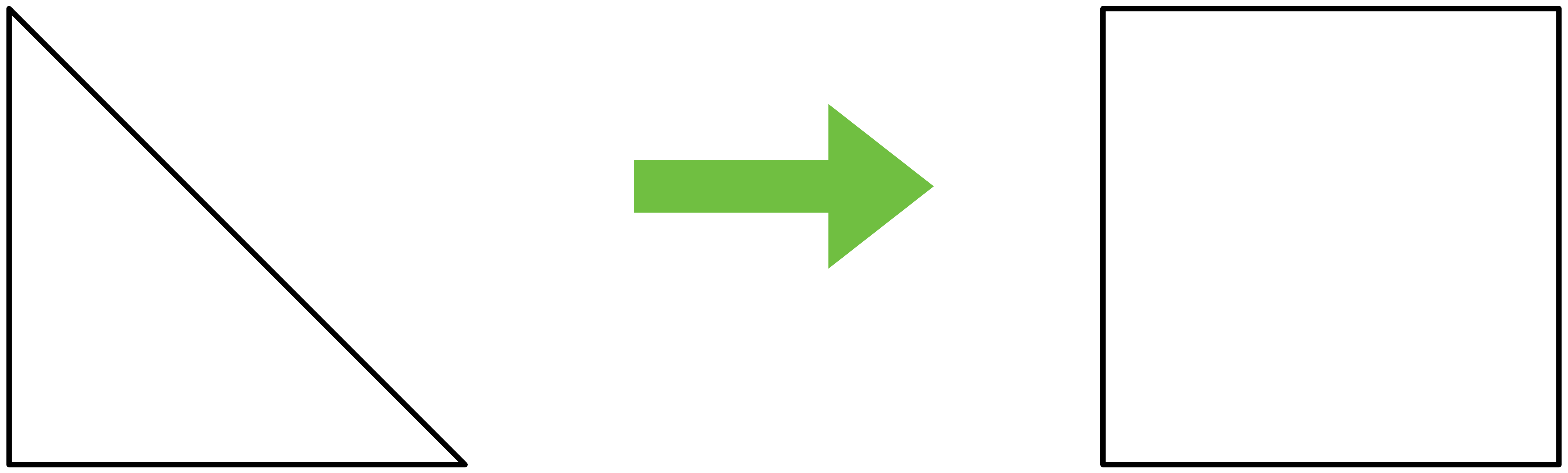
- Extension to **quads and hexes** possible through a **tensor-product construction** of a one dimensional rule.

- 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.



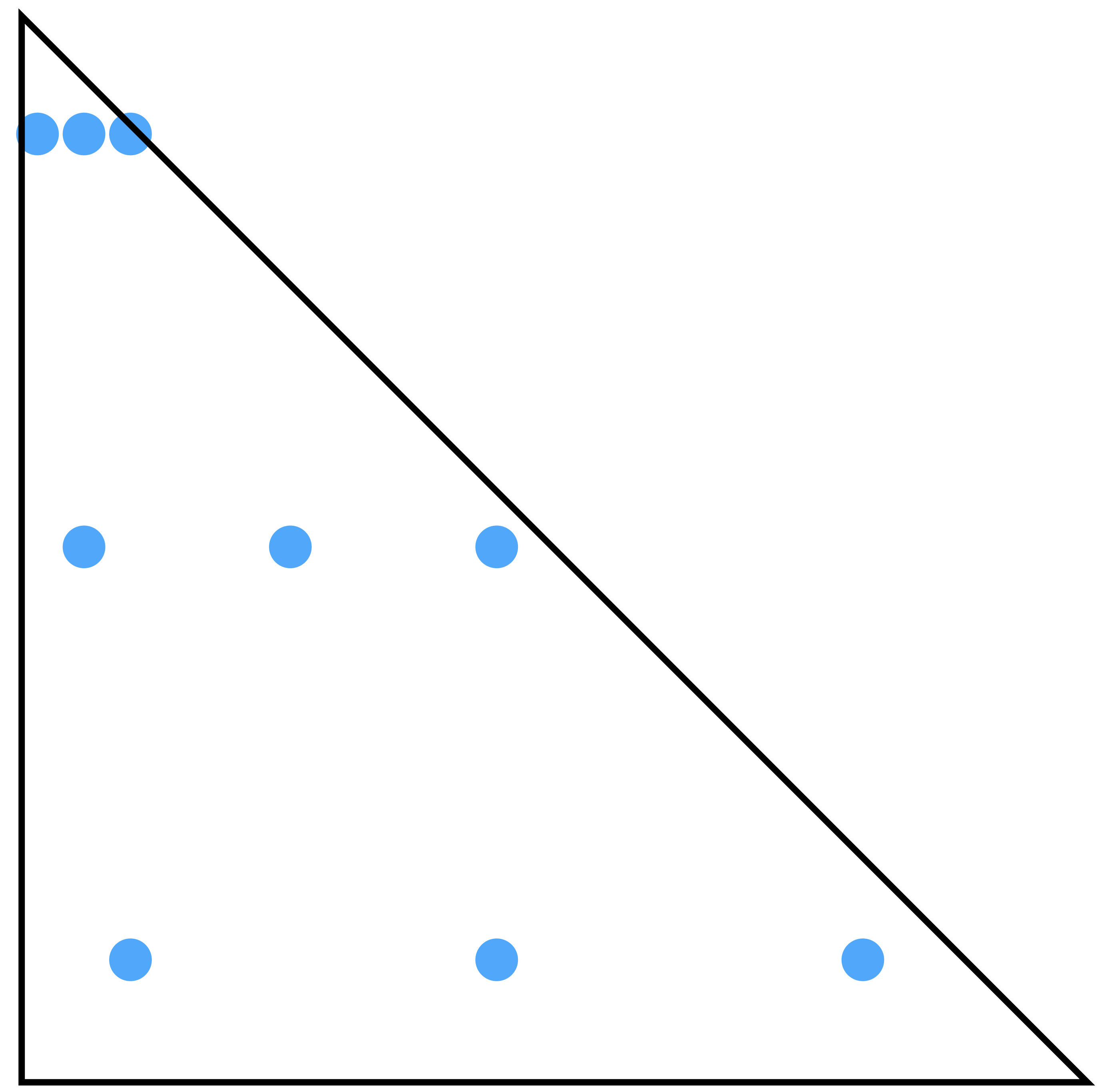
Cubature

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

$$\int_{-1}^{1}\int_{-1}^{-y} f(x,y)\,\mathrm{d}x\mathrm{d}y = \int_{-1}^{1}\int_{-1}^{1} f(x,y)|J|\,\mathrm{d}\tilde{x}\mathrm{d}\tilde{y}$$

Cubature

- 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 $\{\mathbf{x}_1, \omega_1, \ldots, \mathbf{x}_n, \omega_n\}$ where we desire

$$\int_\Omega p_i(\mathbf{x})\,\mathrm{d}\mathbf{x} = \sum_{j=1}^{n} \omega_j p_i(\mathbf{x}_j) \quad \text{for } 1 \leq i \leq m$$

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 × n for the weights.

$$\int_{\Omega} p_i(\mathbf{x})\,\mathrm{d}\mathbf{x} = \sum_{j=1}^{n} \omega_j p_i(\mathbf{x}_j) \quad \text{for } 1 \leq i \leq m$$

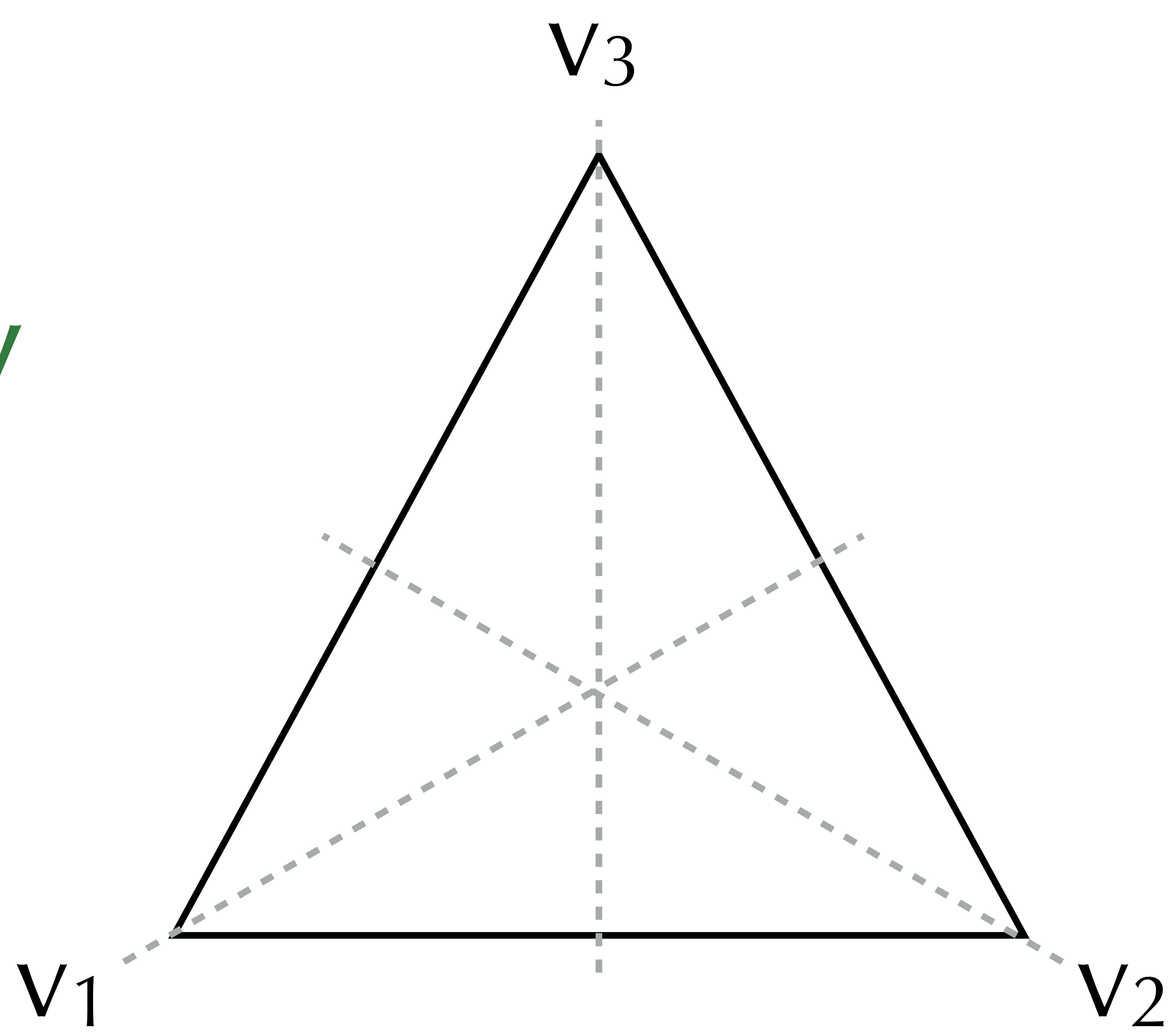
- This is simply a **linear least squares problem** which we may solve directly.

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 $\omega_i > 0$.

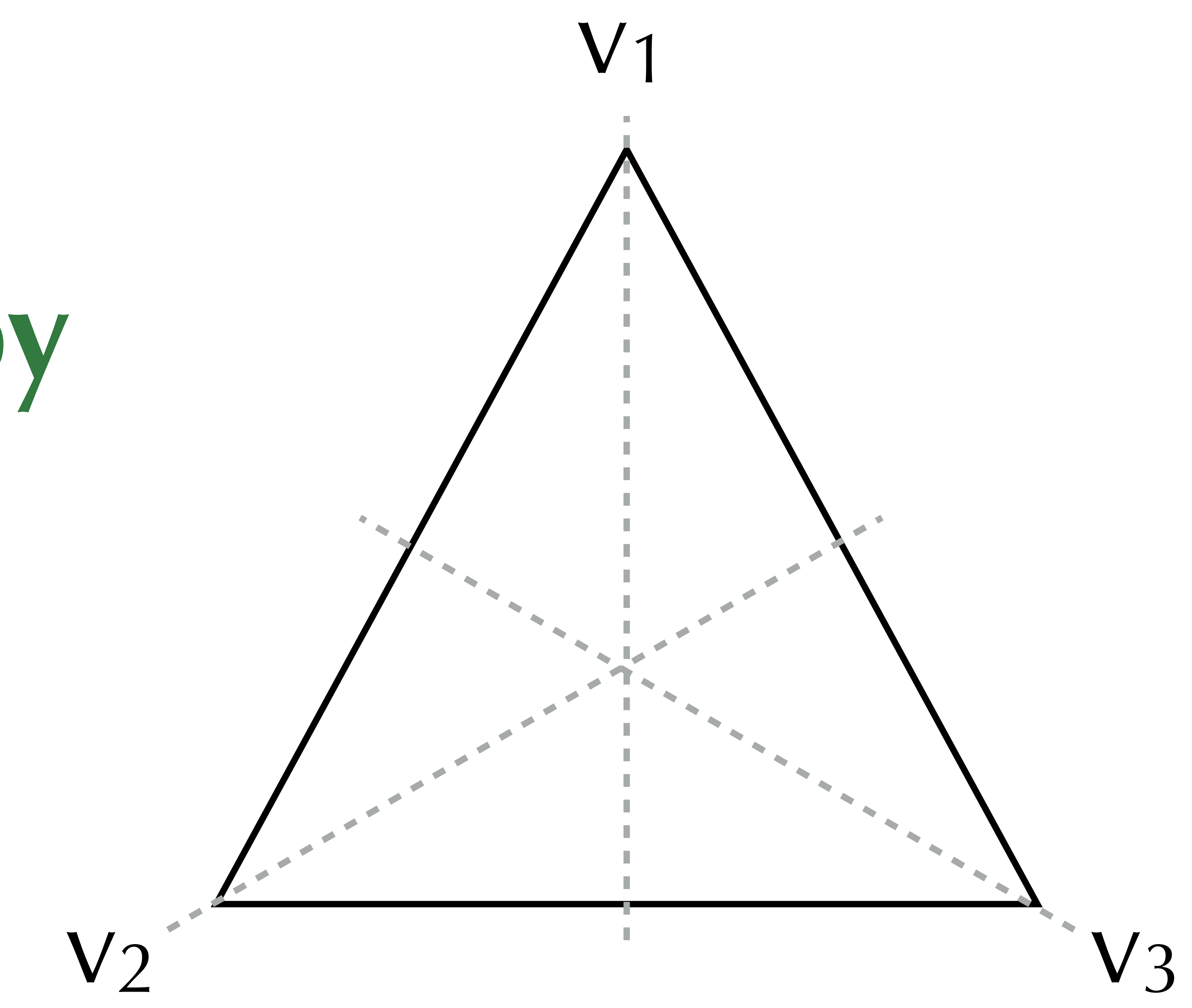
Improvement #2: Symmetry

- Many shapes have **symmetries**.
- Desirable for these to be **displayed by the quadrature rule**.
- Can accomplish this via **symmetry orbits**.



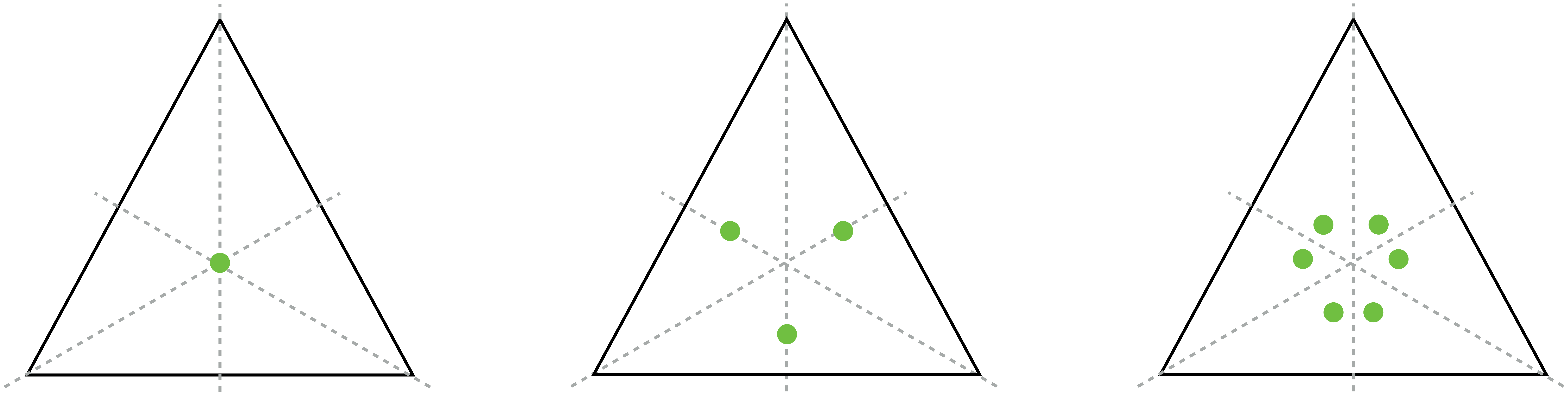
Improvement #2: Symmetry

- Many shapes have **symmetries**.
- Desirable for these to be **displayed by the quadrature rule**.
- Can accomplish this via **symmetry orbits**.



Improvement #2: Symmetry

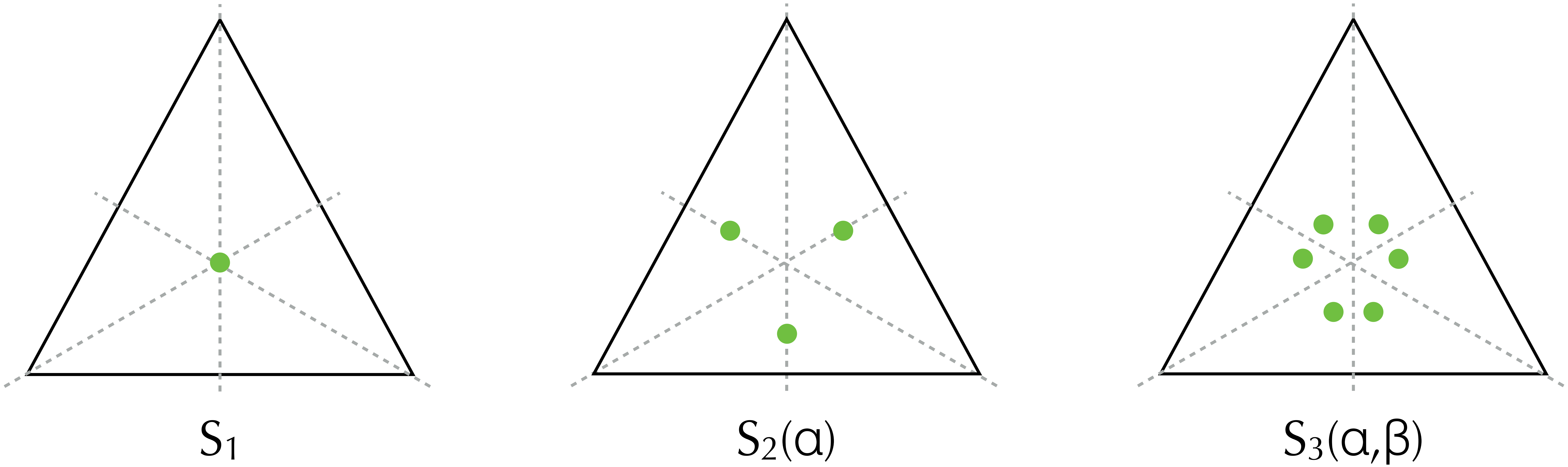
- For example, a triangle has **six symmetries** which can be represented by **three orbits**.



S_1 $S_2(\alpha)$ $S_3(\alpha,\beta)$

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

$$\int_{-1}^{1} x^i \,\mathrm{d}x = 0 \quad \text{for } i \text{ odd}$$

which is satisfied by **all symmetric abscissa**.

Improvement #2: Symmetry

- For example, consider $m = 10$.

	Without symmetry	With symmetry
Triangle	66	36
Quadrangle	66	12
Hexahedron	286	16
Prism	286	91
Pyramid	286	56
Tetrahedron	286	67

Improvement #3: Conditioning

- Using a **monomial basis** such that $p_k(\mathbf{x}) = x^i y^j$ 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

$$\int_\Omega \psi_i(\mathbf{x})\psi_j(\mathbf{x})\,\mathrm{d}\mathbf{x} = \delta_{ij}$$

- Hence

$$\int_\Omega \psi_i(\mathbf{x})\,\mathrm{d}\mathbf{x} = \sum_{j=1}^{n} \omega_j \psi_i(\mathbf{x}_j) \quad \text{for } 1 \leq i \leq m$$

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** Ω a **target order** m and a **point count** n…
  - for each orbital decomposition of n…
    - for i = 1..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

m	Tri	Quad	Tet	Pri	Pyr	Hex
1	1	1	1	1	1	1
2	3	4	4	5	5	6
3	6	4	8	8	6	6
4	6	8	14	11	10	14
5	7	8	14	16	15	14
6	12	12	24	28	24	34

New Rules

m	Tri	Quad	Tet	Pri	Pyr	Hex
7	15	12	<u>35</u>	<u>35</u>	<u>31</u>	<u>34</u>
8	16	20	46	<u>46</u>	<u>47</u>	58
9	19	20	<u>59</u>	<u>60</u>	<u>62</u>	58
10	25	<u>28</u>	81	<u>85</u>	<u>83</u>	<u>90</u>
11	28	<u>28</u>				
12	33	<u>37</u>				

New Rules

m	Tri	Quad	Tet	Pri	Pyr	Hex
13	37	<u>37</u>				
14	42	<u>48</u>				
15	49	<u>48</u>				
16	55	60				
17	60	60				
18	67	<u>72</u>				

New Rules

m	Tri	Quad	Tet	Pri	Pyr	Hex
15	49	<u>48</u>				
16	55	60				
17	60	60				
18	67	<u>72</u>				
19	73	<u>72</u>				
20	79	<u>85</u>				

Back To Interpolation

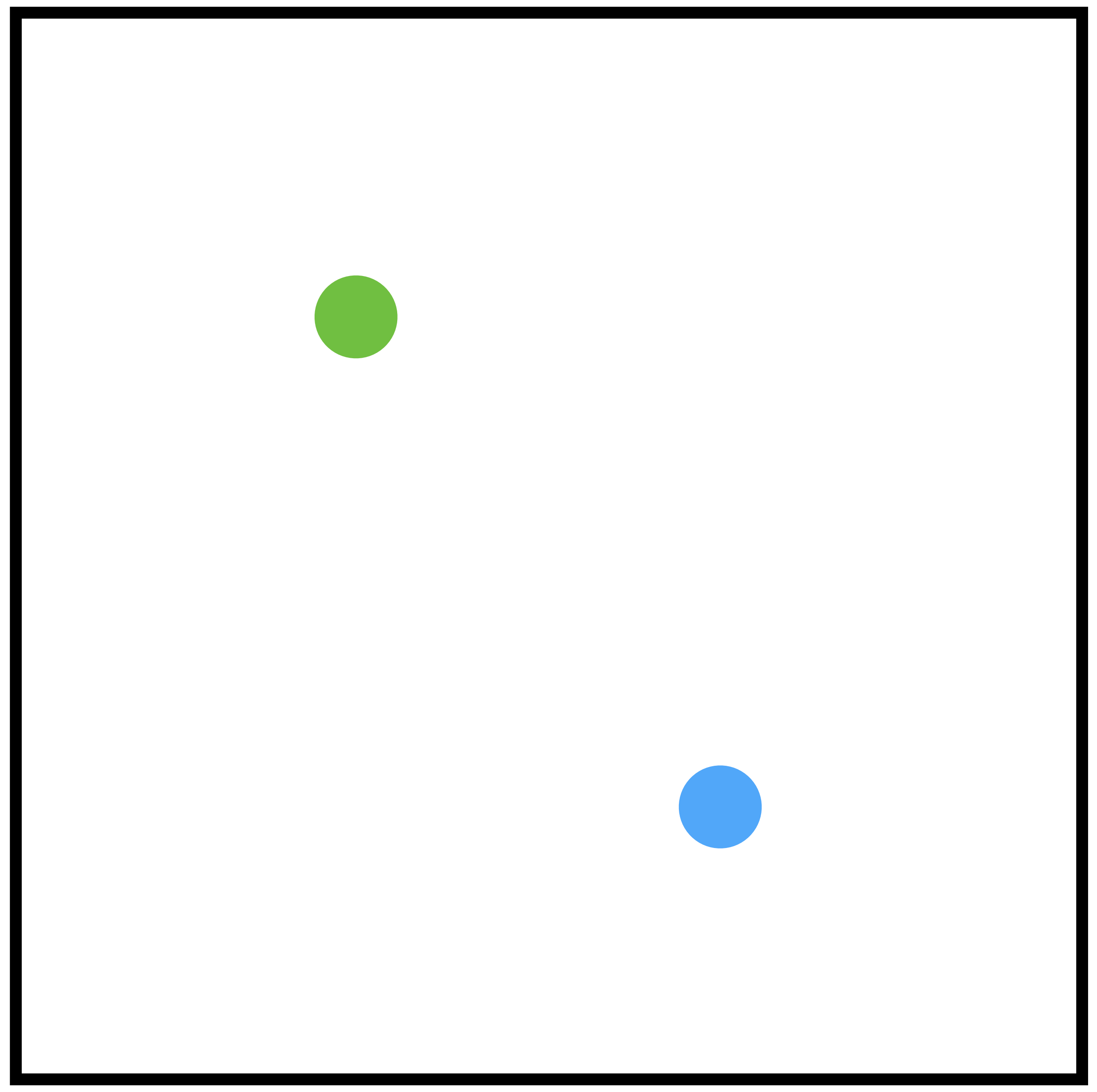
- Given a set of points $\{x_1, x_2, \ldots, x_n\}$ can construct a multivariate Lagrange interpolation polynomial as

$$\ell_i(\mathbf{x}) = \sum_{k=1}^{n} \mathcal{V}_{ik}^{-1} \Psi_k(\mathbf{x}), \quad \mathcal{V}_{ij} = \Psi_i(\mathbf{x}_j)$$

- We therefore require *V* to be **non-singular**.

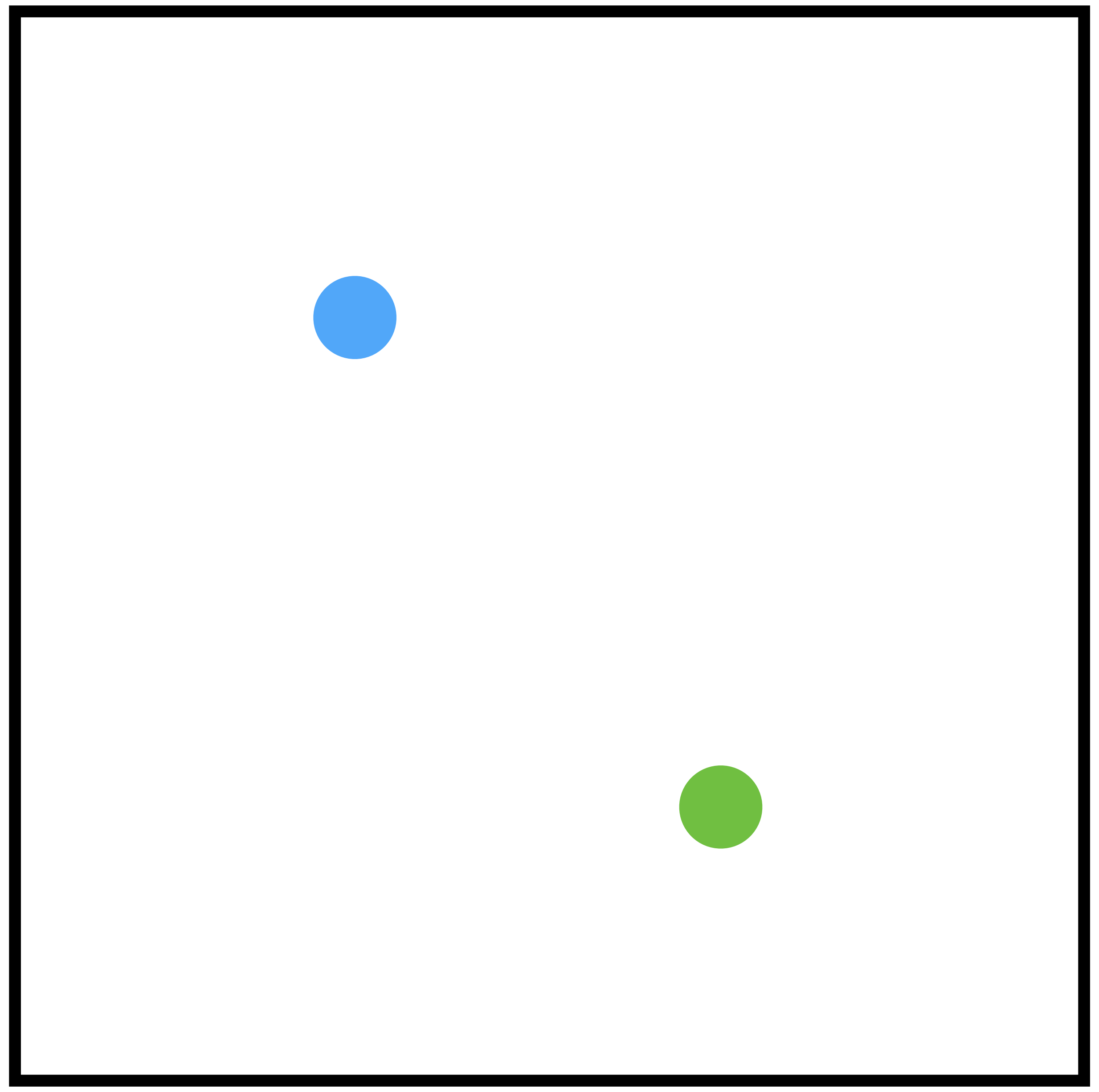
Unisolvency

- 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 $\det(V)$.



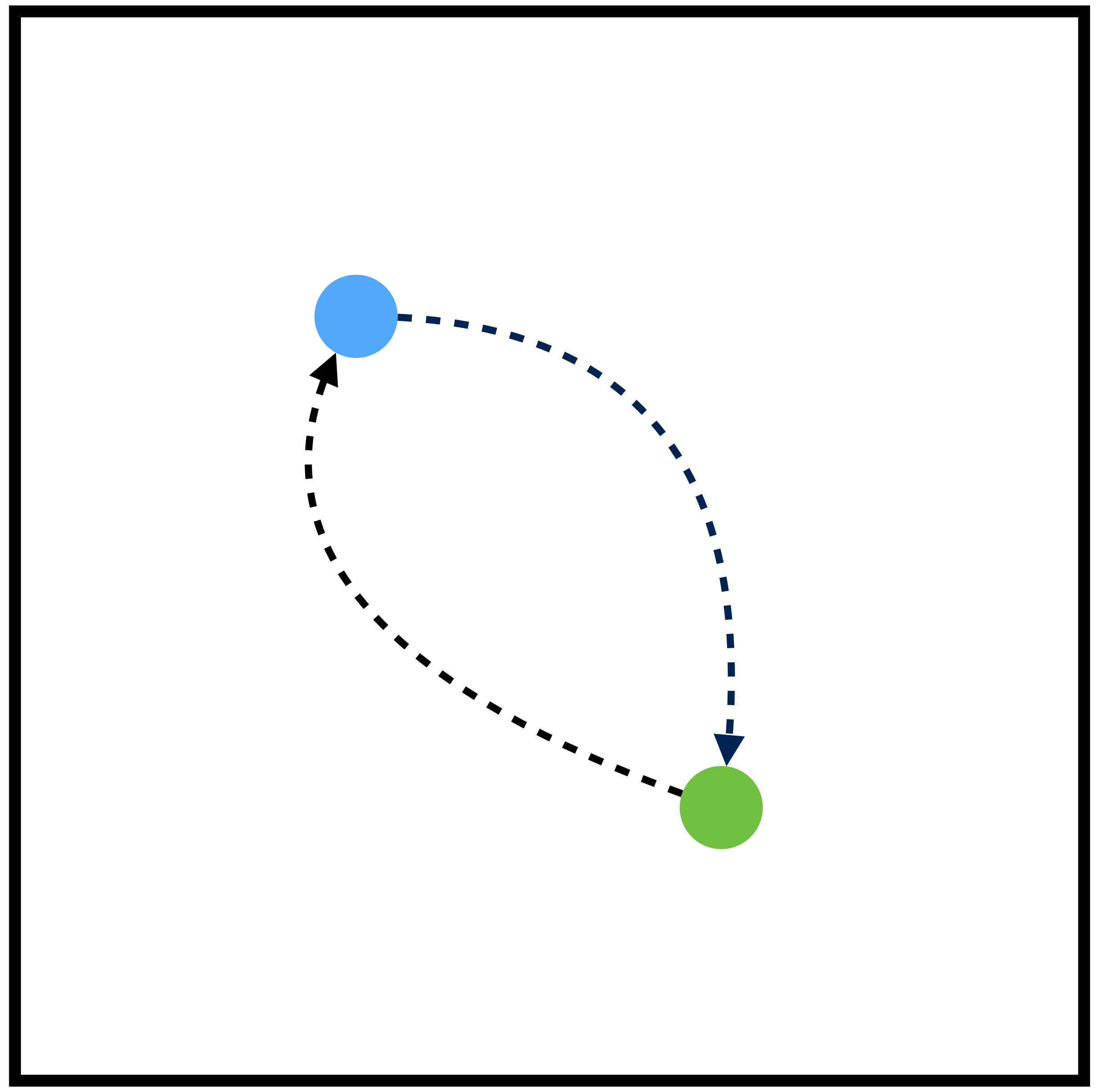
Unisolvency

- 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.

n	Tri	Quad	Tet	Pri	Pyr	Hex
20	—	12	3	35	34	2
40	7	36	13	260	161	6
80	—	121	50	2380	946	56
160	27	441	308	29330	6391	462

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.