Computers and Mathematics with Applications 59 (2010) 663–676

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

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

A numerical algorithm for the construction of efficient quadrature rules in two and higher dimensions

Hong Xiao [a,*], Zydrunas Gimbutas [b]

[a] Department of Mathematics, University of California, Davis, CA 95616, United States

[b] Courant Institute of Mathematical Sciences, New York University, New York, NY 10012, United States

ARTICLE INFO

Article history:
Received 4 July 2009
Accepted 22 October 2009

Keywords:
Multivariate integration
Gaussian quadrature
Triangle
Square
Cube
Point elimination method
Least squares Newton's method

ABSTRACT

We present a numerical algorithm for the construction of efficient, high-order quadratures in two and higher dimensions. Quadrature rules constructed via this algorithm possess positive weights and interior nodes, resembling the Gaussian quadratures in one dimension. In addition, rules can be generated with varying degrees of symmetry, adaptable to individual domains. We illustrate the performance of our method with numerical examples, and report quadrature rules for polynomials on triangles, squares, and cubes, up to degree 50. These formulae are near optimal in the number of nodes used, and many of them appear to be new.

1. Introduction

Numerical integration is a well-studied area in numerical analysis, and classical tools such as Gaussian quadratures are widely used in many fields of science and engineering. Gaussian quadratures have several desirable properties such as positivity of weights, symmetry, and are optimal for integrating polynomials in one dimension: an n-point Gaussian rule is exact for all polynomials of degrees up to $2n - 1$, and no n-point rule is exact for all polynomials of degree $2n$. Since each quadrature node in one dimension contributes two parameters (one for the coordinate, and one for the weight), it is expected that $2n$ functions require an n-point quadrature rule to integrate exactly. Indeed, it can be shown that, in one dimension, Gaussian quadratures can be generalized to a broad range of functions including polynomials, smooth functions and functions with end-point singularities, and effective numerical algorithms have been developed for the construction of these quadratures (see, for example, [1–3]).

The situation in higher dimensions is considerably more complex. While the interval is the only connected compact subset of $\mathbb{R}^1$, regions of $\mathbb{R}^n (n > 1)$ come in an infinite variety of shapes, each with its own symmetries. It is expected that efficient quadratures in $\mathbb{R}^n$ should be studied separately for each individual region. Indeed, extensive effort has been devoted to the study of quadratures on several two and three-dimensional domains including triangles, squares, cubes, tetrahedra, and a number of formulae have been found for integrating polynomials accurately up to moderate degrees (see, for example, [4–8]). However, the theory of numerical integration in dimensions greater than one is far from complete. For instance, the minimum number of nodes required for integrating a given number of functions on a given domain is yet unknown. Furthermore, previous methods largely rely on the detailed analysis of symmetric properties of associated integration domains, and are, therefore, limited in its applicability to each particular geometric shape.

* Corresponding author.
E-mail addresses: xiaoh@math.ucdavis.edu (H. Xiao), gimbutas@cims.nyu.edu (Z. Gimbutas).

0898-1221/$ – see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2009.10.027

In this paper, we present a numerical algorithm for the construction of quadrature rules on regular regions in two and higher dimensions. This method combines a node elimination scheme and the least squares Newton's method, and is effective in generating quadratures of moderate to high orders (up to 50). The resulting formulae have positive weights and interior nodes, resembling the Gaussian quadratures in one dimension. In addition, our algorithm is applicable to the construction of quadratures of all types of symmetries, with the degree of symmetry being a user-defined feature. We illustrate our method with numerical examples for the triangle, square, and cube. Many of the reported formulae use a lower number of nodes than any formula previously published.

The paper is organized as follows. We introduce mathematical and numerical preliminaries in Section 2, and develop analytical apparatus to be used by the algorithm in Section 3. We describe in detail the algorithm in Section 4, and present numerical results in Section 5. Finally, Section 6 contains conclusions and discussions.

2. Mathematical and numerical preliminaries

In this section, we collect elementary mathematical and numerical facts relevant to the development of the algorithm of this paper.

2.1. Quadratures

A quadrature in $\mathbb{R}^d$ is a formula of the form

$$\sum_{i=1}^{n} w_i f(\mathbf{x}_i) \approx \int_{\Omega} \omega(\mathbf{x}) f(\mathbf{x}) \mathrm{d}\mathbf{x}, \tag{1}$$

where Ω is the integration region in $\mathbb{R}^d$, f is an integrand defined on Ω, and ω is the weight function. The points $\mathbf{x}_i \in \mathbb{R}^d$ are typically called quadrature nodes, and w_i quadrature weights.

Typically, quadratures are designed such that (1) is exact for a pre-selected set of functions $\{\phi_1, \ldots, \phi_m\}$. Thus, an n-point quadrature with nodes $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ and weights $w_1, w_2, \ldots, w_n$ that integrates exactly $\phi_1, \ldots, \phi_m$ must satisfy the system of equations

$$\begin{pmatrix} \int_{\Omega} \omega(\mathbf{x})\phi_1(\mathbf{x})\mathrm{d}\mathbf{x} \\ \int_{\Omega} \omega(\mathbf{x})\phi_2(\mathbf{x})\mathrm{d}\mathbf{x} \\ \vdots \\ \int_{\Omega} \omega(\mathbf{x})\phi_m(\mathbf{x})\mathrm{d}\mathbf{x} \end{pmatrix} = \begin{pmatrix} \phi_1(\mathbf{x}_1) & \phi_1(\mathbf{x}_2) & \ldots & \phi_1(\mathbf{x}_n) \\ \phi_2(\mathbf{x}_1) & \phi_2(\mathbf{x}_2) & \ldots & \phi_2(\mathbf{x}_n) \\ \vdots & & & \\ \phi_m(\mathbf{x}_1) & \phi_m(\mathbf{x}_2) & \ldots & \phi_m(\mathbf{x}_n) \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix}. \tag{2}$$

Clearly, given the weight function ω and the integration region Ω, the number n of quadrature nodes depends upon the number m of the functions to be integrated. Therefore, a classical problem in numerical integration is to determine the minimum number n of nodes in (2) and the corresponding coordinates and weights. Quadratures that achieve the minimum n are considered *optimal*, and in one dimension, the optimal quadratures are the so-called generalized Gaussian quadratures which use exactly $\lceil m/2 \rceil$ nodes to integrate $\phi_1, \ldots, \phi_m$ (with $\lceil x \rceil$ denoting the smallest integer that is greater or equal to x) [2,3].

The existence of generalized Gaussian quadratures in one dimension seems to suggest that the optimal quadratures in R^d should involve $\lceil m/(d+1) \rceil$ nodes for integrating m functions, since each d-dimensional node contributes $d+1$ parameters. However, no such theorem has been established for d greater than 1, to the best of the authors knowledge. In order to differentiate quadratures that integrate the same set of functions with different number of nodes, we introduce the term *efficiency* in this paper, which is defined as

$$E = \frac{m/(d+1)}{n}, \tag{3}$$

where n is the number of nodes, m is the number of functions, and d is the dimension of the Euclidean space. Obviously, each generalized Gaussian quadrature in one dimension has $E = 1$. In the remainder of the paper, we call quadratures with value E close to 1 *efficient*.

2.2. Tensor product rules

Tensor product rules are a class of two- and higher-dimensional quadratures that are constructed based on one-dimensional quadratures. In this section, we outline a possible construction of such rules, as well as identify their deficiencies. For simplicity, we assume that the functions to be integrated are polynomials and that the integration region is

the standard square S in $\mathbb{R}^2$ with boundaries $x = \pm 1$ and $y = \pm 1$. The construction of tensor product rules on other regions in $\mathbb{R}^d$ (e.g. rectangles, cubes, triangles, and tetrahedra, etc.) and for other separable functions is similar.

As is well known, monomials of the form $x^i y^j$ form a basis for polynomials in $\mathbb{R}^2$. Suppose that we are to integrate on the square all monomials $x^i y^j$ up to the degree $2p-1$ (i.e., $0 \le i+j \le 2p-1$). Defining $t_1, t_2, \ldots, t_p$ and $v_1, v_2, \ldots, v_p$ as the nodes and weights of the p-point Gaussian quadrature on $[-1, 1]$, we know that $(t_1, v_1), (t_2, v_2), \ldots, (t_p, v_p)$ integrates all monomials t^j for all j less than $2p$. Let

$$\{q_{i,j} \in S | q_{i,j} = (t_i, t_j), i = 1 \ldots, p, j = 1, \ldots, p\}$$

be the p^2 grid points in S and let

$$\{w_{i,j} | w_{i,j} = v_i v_j, i = 1 \ldots, p, j = 1, \ldots, p\}$$

be the corresponding weights, we can easily verify that $(q_{1,1}, w_{1,1}), (q_{1,2}, w_{1,2}), \ldots, (q_{p,p}, w_{p,p})$ form a quadrature that integrates all monomials $x^i y^j$ for $0 \le i+j < 2p$ on S. This p^2-point quadrature is a typical tensor product rule on S of order $2p-1$.

Although tensor product rules are relatively easy to construct, they are suboptimal in terms of efficiency. For example, the p^2-point tensor product rule constructed above yields

$$E = \frac{p(2p+1)/3}{p^2} = \frac{2p+1}{3p}, \tag{4}$$

since there are $p\,(2p+1)$ monomials in the set. Clearly, the efficiency of a two-dimensional tensor product rule tends to $2/3$ as p increases. Similar analysis show that the efficiency of tensor product rules in three dimensions tends to $1/3$, with E becomes even lower in higher dimensions.

2.3. Least squares Newton's method

Newton's method is a classical tool for solving equations of the type

$$\mathbf{F}(\mathbf{x}) \equiv \begin{pmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_n(\mathbf{x}) \end{pmatrix} = \mathbf{0}, \tag{5}$$

where $\mathbf{F} : \mathbb{R}^m \to \mathbb{R}^n$ is typically nonlinear. Newton's method approximates the solution of (5) via a sequence of vectors $\mathbf{x}_k$ given by the formula

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{J}^\dagger(\mathbf{x}_k)\mathbf{F}(\mathbf{x}_k), \tag{6}$$

where $k > 0$, $\mathbf{x}_0$ is an initial estimation of the solution, $\mathbf{J} : \mathbb{R}^m \to \mathbf{R}^n$ is the Jacobian (or the Fréchet derivative) of $\mathbf{F}$, given by the formula

$$\mathbf{J}(\mathbf{x}) = \begin{pmatrix} \dfrac{\partial f_1}{\partial x_1}(\mathbf{x}) & \ldots & \dfrac{\partial f_1}{\partial x_m}(\mathbf{x}) \\ \vdots & & \\ \dfrac{\partial f_n}{\partial x_1}(\mathbf{x}) & \ldots & \dfrac{\partial f_n}{\partial x_m}(\mathbf{x}) \end{pmatrix}, \tag{7}$$

and $\mathbf{J}^\dagger(\mathbf{x})$ is the pseudo-inverse of $\mathbf{J}$ at $\mathbf{x}$. When $m \ne n$, (6) is referred to as the *least squares* Newton's method.

It is well known that Newton's method has second-order convergence under certain conditions. In Theorem 2.1 below, we summarize this fact under the constraint that is relevant to our paper. We refer the reader to [9] for additional details.

Theorem 2.1. *Suppose that $\mathbf{F} : \mathbb{R}^m \to \mathbb{R}^n$ $(m \ge n)$ is a function and $\boldsymbol{\xi} \in \mathbb{R}^m$ satisfies the condition*

$$\mathbf{F}(\boldsymbol{\xi}) = \mathbf{0}. \tag{8}$$

Furthermore, suppose that $\mathbf{F}$ is continuously differentiable in an ϵ-neighborhood $n_\epsilon(\boldsymbol{\xi}) = \{\mathbf{x} | \|\mathbf{x} - \xi\| < \epsilon\}$ of $\boldsymbol{\xi}$, and that $\mathbf{J}(\mathbf{x})$ defined by (7) has full row rank for all $\mathbf{x}$ in $n_\epsilon(\boldsymbol{\xi})$. Let $\mathbf{x}_k$ be defined by (6), and let h_n be given by the formula

$$h_n = \mathbf{x_n} - \boldsymbol{\xi}. \tag{9}$$

Then, there exist $\epsilon > \tilde{\epsilon} > 0$, $\delta > 0$ and integer $N_{\tilde{\epsilon},\delta} \ge 0$, such that if $\|\mathbf{x_0} - \boldsymbol{\xi}\| < \tilde{\epsilon}$, then

$$\|h_{n+1}\| \le \delta \|h_n\|^2 < \|h_n\|$$

is true for all $k \ge N_{\tilde{\epsilon},\delta}$.

2.4. Singular Value Decomposition

In this section, we summarize the Singular Value Decomposition method (or SVD) (see, for example, [10]), which will be used in Section 3.3 for the computation of basis functions.

Theorem 2.2. *Suppose that A is an $n \times m$ matrix. Then there exists a matrix factorization*

$$A = USV^*,$$

such that U is a unitary $n \times p$ matrix, V is a unitary $m \times p$ matrix, and $S = [s_{ij}]$ is a diagonal real matrix of size $p \times p$ such that $s_{ii} \geq s_{i+1,i+1} \geq 0$ for all $i = 1, \ldots, p-1$.

The diagonal entries s_{ii} of S are called the singular values of A, and the columns of U and V are referred to as the left and right singular vectors of A, respectively.

3. Analytical apparatus

In this section, we provide the analytical apparatus relevant to the construction of the quadratures of this paper. We first study the action of certain orthogonal transformations on point sets in $\mathbb{R}^d$ in Section 3.1, and analyze quadratures that are invariant under such transformations in Section 3.2. We then introduce formulae for the construction of symmetric and non-symmetric basis functions in R^d in Section 3.3. Finally, we introduce a pivoted Gram–Schmidt procedure to be used in Section 4 for the selection of an initial quadrature.

3.1. Symmetric point sets and S-generators

As is well known, the symmetry group G of $\Omega \subset \mathbb{R}^d$ is the set of all orthogonal linear transformations of $\mathbb{R}^d$ that map Ω onto itself. A subset $S \subseteq G$ is called a symmetry subgroup of Ω if S is itself a group. We say that two points $\mathbf{x}_1, \mathbf{x}_2$ in Ω are S-symmetrically related if there exists an element $s \in S$ such that $\mathbf{x}_2 = s\mathbf{x}_1$. We call the set of all points that are S-symmetrically related with $\mathbf{x}$ the orbit of $\mathbf{x}$, and denote it by $\mathcal{O}^S_{\mathbf{x}}$.

If for each point $\mathbf{x}$ in $\mathcal{X} \subset \Omega$, all S-symmetrically related points are also in $\mathcal{X}$, then we say that $\mathcal{X}$ is S-symmetric (or invariant with respect to S). It can be easily seen that any S-symmetric set $\mathcal{X}$ is a union of distinct orbits. Suppose that the set $\mathcal{B}$ contains only one and only point from each orbit contained in $\mathcal{X}$. Then we say that $\mathcal{B}$ is an S-generator of $\mathcal{X}$. Thus, all S-symmetric sets $\mathcal{X}$ can be expressed as $\mathcal{X} = \bigcup_{\mathbf{x}\in\mathcal{B}} \mathcal{O}^S_{\mathbf{x}}$, where $\mathcal{O}^S_{\mathbf{x}_i} \bigcap \mathcal{O}^S_{\mathbf{x}_j} = \emptyset$ for all $\mathbf{x}_i \in \mathcal{B}$, $\mathbf{x}_j \in \mathcal{B}$ and $\mathbf{x}_i \neq \mathbf{x}_j$. Obviously, many possible $\mathcal{B}$'s may exist for an arbitrary $\mathcal{X}$. For example, $\mathcal{X}$ may be partitioned geometrically based on the location of its elements within Ω.

Example 3.1. Consider the standard triangle T with the vertices

$$\left(-1, -1/\sqrt{3}\right), \left(0, 2/\sqrt{3}\right) \text{ and } \left(1, -1/\sqrt{3}\right). \tag{10}$$

The symmetry group of T is a dihedral group (often named D_3) which consists of six elements: reflection about the three medians (denoted by A, B, C), and rotations about the center by 0, $2\pi/3$ and $4\pi/3$ radians (denoted by E, D, F, respectively, among which E is the identity transformation). The symmetry group D_3 mentioned above has six subgroups, which are $S_1 = \{E\}$, $S_2 = \{E, A\}$, $S_3 = \{E, B\}$, $S_4 = \{E, C\}$, $S_5 = \{E, D, F\}$, and $S_6 = \{E, A, B, C, D, F\} = D_3$, respectively. The smaller triangles T_6 (bounded by the bottom edge, the vertical median and the median bisecting the angle at the lower-right vertex of T, see Fig. 1), T_3, and T_2 are examples of the S_6-, S_5-, and S_2-generators of T, respectively.

3.2. S-symmetric quadratures

The primary purpose of this section is Theorem 3.2, which shows that an S-symmetric quadrature is completely determined by its configuration on any S-generator of the integration region.

Suppose that $\Omega \subseteq \mathbb{R}^d$ is the integration region, G is its symmetry group, and S is a symmetry subgroup of G. We say that a p-point quadrature whose nodes have coordinates $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_p\}$ and weights $\mathcal{W} = \{w_1, \ldots, w_p\}$ is *S-symmetric* (or invariant under the action of S) if $\mathcal{X}$ is an S-symmetric point set, and all points in $\mathcal{X}$ that are S-symmetrically related (see Section 3.1) have the same weight.

Clearly, an S-symmetric quadrature $(\mathcal{X}, \mathcal{W})$ is completely determined by its nodes $\mathcal{X}_{\mathcal{B}}$ that lie in a single S-generator $\mathcal{B}$ of Ω. In other words, the construction of an S-symmetric quadrature on Ω is equivalent to the design of a corresponding quadrature on $\mathcal{B}$ such that $\mathcal{X}_{\mathcal{B}} = \mathcal{X} \bigcap \mathcal{B} = \{\mathbf{x}_i, \ldots, \mathbf{x}_n\}$ and $\mathcal{W}_{\mathcal{B}} = \{w_1, \ldots, w_n\}$ satisfy the Eq. (2) on $\mathcal{B}$. Theorem 3.2 below summarizes this fact; the corresponding S-symmetric quadrature $\mathcal{X}$ on the original integration region Ω is then obtained via the formula

$$\mathcal{X} = \bigcup_{\mathbf{x}_i \in \mathcal{X}_{\mathcal{B}}} \mathcal{O}^S_{\mathbf{x}_i}, \tag{11}$$

with nodes belonging to the same orbit $\mathcal{O}^S_{\mathbf{x}_i}$ assuming the same weight w_i.

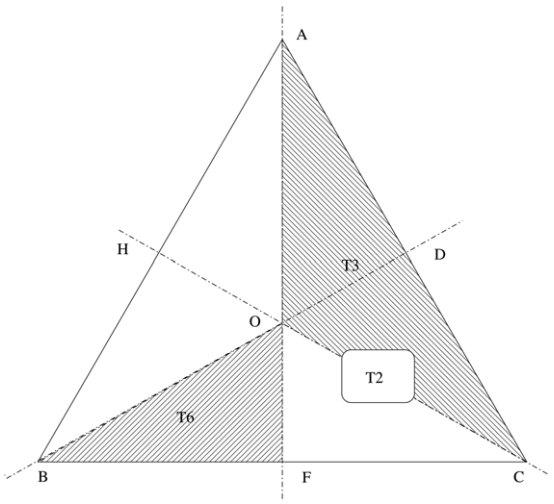

Fig. 1. The standard equilateral triangle T and some of its symmetric generators: an S_2-generator T_2 (i.e., triangle ACF), an S_5-generator T_3 (i.e., triangle AOC), and an S_6-generator T_6 (i.e., triangle BOF).

Theorem 3.2. *Suppose that $\mathcal{X}_{\mathcal{B}} = \{\mathbf{x}_i, \ldots, \mathbf{x}_n\}$ and $\mathcal{W}_{\mathcal{B}} = \{w_1, \ldots, w_n\}$ is an n-point quadrature on $\mathcal{B}$, an S-generator of Ω, such that*

$$\int_{\mathcal{B}} \phi_j = \sum_{\mathbf{x}_i \in \mathcal{X}_{\mathcal{B}}} w_i \, \phi_j(\mathbf{x}_i), \tag{12}$$

is accurate for all ϕ_j ($j = 1, 2, \ldots, m$) defined on Ω. Then, the S-symmetric p-point quadrature satisfies the equation

$$\int_{\Omega} \phi_j = \sum_{\mathbf{x}_i \in \mathcal{X}} w_i \, \phi_j(\mathbf{x}_i), \tag{13}$$

for all $j = 1, \ldots, m$ if the nodes $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_p\}$ are given by the formula

$$\mathcal{X} = \bigcup_{\mathbf{x}_i \in \mathcal{X}_{\mathcal{B}}} \mathcal{O}^S_{\mathbf{x}_i}, \tag{14}$$

and all weights belonging to nodes in the same orbit are equal.

Proof. Let ϕ be an arbitrary function in $\{\phi_j\}$ ($j = 1, 2, \ldots, m$) defined on Ω. Due to properties of the group action of S, we have

$$\int_{\Omega} \phi = \sum_{g \in S} \int_{g(\mathcal{B})} \phi = \sum_{g \in S} \int_{\mathcal{B}} g^{-1} \circ \phi = \sum_{g \in S} \frac{1}{\det(g)} \int_{\mathcal{B}} \phi, \tag{15}$$

with $\det(g)$ being the determinant of the orthogonal transformation g. On the other hand,

$$\sum_{\mathbf{x}_i \in \mathcal{X}} w_i \, \phi(\mathbf{x}_i) = \sum_{g \in S} \left(\sum_{\mathbf{x}_j \in \mathcal{X}_{\mathcal{B}}} w_j \, \phi(g(\mathbf{x}_j)) \right) = \sum_{g \in S} \left(\sum_{\mathbf{x}_j \in \mathcal{X}_{\mathcal{B}}} w_j \, (g^{-1} \circ \phi(\mathbf{x}_j)) \right), \tag{16}$$

which may be rewritten as

$$\sum_{g \in S} \left(\sum_{\mathbf{x}_j \in \mathcal{X}_{\mathcal{B}}} w_j \, (g^{-1} \circ \phi(\mathbf{x}_j)) \right) = \sum_{g \in S} \frac{1}{\det(g)} \left(\sum_{\mathbf{x}_j \in \mathcal{X}_{\mathcal{B}}} w_j \, \phi(\mathbf{x}_j) \right) = \sum_{g \in S} \frac{1}{\det(g)} \int_{\mathcal{B}} \phi. \quad \square \tag{17}$$

3.3. Orthogonal basis functions and symmetry

Although many classes of functions may be of interest, we focus, in this paper, on quadratures that integrate multivariate polynomials, and select $\phi_1, \ldots, \phi_m$ to be a basis for all polynomials up to a certain degree on $\Omega \subset \mathbb{R}^d$. The solution of (2) then gives a quadrature that integrates all polynomials accurately up to that degree on Ω.

A natural basis for polynomials on $\mathbb{R}^d$ is the set of monomials of the form $x_1^{m_1} \cdot x_2^{m_d} \cdots x_d^{m_d}$. However, monomials tend to be poorly conditioned even when the orders are moderate. Thus, orthogonal polynomial basis are often used.

For regions that are cartesian products of (orthogonal) intervals including rectangles, squares, cubes, orthogonal polynomial bases may be constructed via tensor product of one-dimensional orthogonal polynomials. For other simple two- and three-dimensional regions such as triangles and tetrahedra, explicit formulae for orthogonal polynomials are a classical research topic, with many forms available. For example,

$$K_{m,n}(u, v) = P_m\left(\frac{2u + v + 1}{1 - v}\right) \cdot \left(\frac{1 - v}{2}\right)^m \cdot P_n^{2m+1,0}(v) \tag{18}$$

($0 \le m, n \le d$ and $m + n \le d$) constitute a well-known orthogonal basis for all polynomials of degrees at most d on the right angle triangle T_R with vertices $(-1, -1)$, $(-1, 1)$ and $(1, -1)$ (see [11]), where $P_n^{\alpha,\beta}$ denotes the nth degree Jacobi polynomial corresponding to parameters α and β, and P_m denotes the mth degree Legendre polynomial. Via a change of coordinates, orthogonal bases may be obtained for related geometries.

Example 3.3. By changing coordinates

$$u = x - \frac{y}{\sqrt{3}} - \frac{1}{3}, \qquad v = \frac{2y}{\sqrt{3}} - \frac{1}{3}, \tag{19}$$

and substituting (19) into (18), we obtain

$$\bar{K}_{m,n}(x, y) = P_m\left(\frac{2u(x, y) + v(x, y) + 1}{1 - v(x, y)}\right) \cdot \left(\frac{1 - v(x, y)}{2}\right)^m \cdot P_n^{2m+1,0}(v(x, y)), \tag{20}$$

which form an orthogonal basis on the standard equilateral triangle T (see (10)).

Remark 3.4. Orthogonal bases for functions other than the polynomials may be constructed via Gram–Schmidt orthogonalization or Singular Value Decomposition, either numerically or symbolically.

3.4. A pivoted Gram–Schmidt algorithm

In this section, we present a pivoted Gram–Schmidt algorithm, which will be used in Section 4 for the selection of an initial quadrature. Unlike the classical Gram–Schmidt algorithm, which returns an orthonormal basis of the span of a given vector set with each basis vector generally not a member of the original set, the Pivoted Gram–Schmidt algorithm returns *a subset* of the original vectors to form the basis. In this section, we assume that $\pi = \{q_1, \ldots, q_m\} \subset \mathbb{R}^n$ is a set of m column n-vectors, and that m is no less than n. Denoting by k the dimension of the span of π and by $\|\cdot\|$ the l_2-norm of a vector, we outline the algorithm below.

Algorithm 3.5 (*Pivoted Gram–Schmidt*). Given $\pi = \{q_1, \ldots, q_m\} \subset \mathbb{R}^n$, the following algorithm computes the dimension k of π, and returns the indices $p_1, p_2, \ldots, p_k$ and an ordered sequence $Q = \{q_{p_1}, q_{p_2}, \ldots, q_{p_k}\}$ of vectors such that $q_{p_j} \in \pi$ for all j, that q_{p_1} has the largest 2-norm among all vectors in π, and that $q_{p_{j+1}}$ has the largest projection in the orthogonal complement of *span* $(q_{p_1}, \ldots, q_{p_j})$ for all $1 \le j \le k - 1$.

$p_i = i$ for $1 \le i \le m$
for $i = 1 : m$ **do**
  Determine μ with $i \le \mu \le m$ so $\|q_\mu\| = max_{i \le j \le m}\|q_j\|$
  if $\|q_\mu\| \ne 0$ **then**
    $q_\mu \leftrightarrow q_i$; $p_\mu \leftrightarrow p_i$
    for $j = i + 1 : m$ **do**
      $q_j = q_j - (q_j^T q_i)q_i/\|q_i\|$
    end for
  end if
end for
$k = i - 1$

The complexity of the algorithm is $O(nmk)$. In our implementation, a re-orthogonalization step is used to ensure numerical stability.

4. Construction of quadratures of varying symmetries

In this section, we present a numerical algorithm for the construction of a class of quadratures that use as few number of nodes as possible to integrate a given set of functions on domains in two and higher dimensions. The algorithm is a combination of the node elimination algorithm, and the least squares Newton's method. A primary advantage of our algorithm is its flexibility in handling higher-dimensional quadratures and their symmetries,with a majority of the quadratures obtained having desirable properties such as positivity of weights and interior nodes.

4.1. Node elimination algorithm

Observation 4.1. *Suppose that $\Omega \in \mathbb{R}^d$ is an integration region in d dimensions and that $\phi_1, \ldots, \phi_m$ are functions defined on Ω. Suppose further that $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ and $w_1, \ldots, w_n$ are nodes and weights of an n-point quadrature for integrating $\phi_1, \ldots, \phi_m$, and that the nth weight $w_n \sim \varepsilon$ is small. Then, $\mathbf{x}_1, w_1, \ldots, \mathbf{x}_{n-1}, w_{n-1}, \mathbf{x}_n$ and $w_n = 0$ may be considered as an approximate solution to (2) with precision $O(\varepsilon)$, as long as ϕ_j's are relatively smooth and are normalized properly.*

The implication of the above observation is that the nodes $\mathbf{x}_1, \ldots \mathbf{x}_{n-1}$ and the weights $w_1 \ldots, w_{n-1}$ may be considered as an $(n-1)$-point quadrature that integrates the requested functions *approximately* (to precision $O(\varepsilon)$). Using these values as an initial approximation for the least squares Newton's method for solving (2), we will obtain an *accurate* $(n-1)$-point quadrature, provided that the approximate solution is close enough to the true solution for the Newton's method to converge. The process of repeated elimination of "unnecessary" nodes in order to obtain a sequence of quadratures with fewer and fewer nodes is described in the algorithm follows.

Algorithm 4.2 (*Node Elimination*). Given a set of m functions $\phi_1, \ldots, \phi_m$ and an n-point quadrature with nodes and weight $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and $w_1, \ldots, w_n$, respectively, the following algorithm computes a k-point quadrature $\mathbf{x}_1, \ldots, \mathbf{x}_k, w_1, \ldots, w_k$ for integrating $\phi_1, \ldots, \phi_m$ where k is no greater than n.

1: **for** $k = n : 1$ **do**
2: **reorder** $\mathbf{x}_j$ and w_j for $j = 1 : k$ in increasing order of the "significance index" s_j
3: **for** $i = 1 : k$ **do**
4: **run** least squares Newton's method to eliminate ith node ($\mathbf{x}_i$ and w_i)
5: **if** Newton's method succeeds **then**
6: **save** new nodes, weights to $\mathbf{x}_1, \ldots, \mathbf{x}_{k-1}, w_1, \ldots, w_{k-1}$;
7: **goto** line 11
8: **end if**
9: **end for**
10: **return** $\mathbf{x}_1, \ldots, \mathbf{x}_k, w_1, \ldots, w_k$
11: **end for**

In our experiments, the "significance index" s_j (line 2 above) for each quadrature node is computed via either

$$s_j = w_j \sum_{i=1}^{m} \phi_i^2(\mathbf{x}_j), \quad j = 1, \ldots, k, \tag{21}$$

or

$$s_j = \sum_{i=1}^{m} \phi_i^2(\mathbf{x}_j), \quad j = 1, \ldots, k, \tag{22}$$

which essentially compute the (weighted) norms of the column vector $(\phi_1(\mathbf{x}_j), \ldots, \phi_m(\mathbf{x}_j))^T$ in (2). Our experience shows that the node elimination algorithm is insensitive to the exact form of s_j, and the elimination process typically continues successfully until the number of quadrature nodes is extremely close to the expected value $m/(d+1)$ (see Section 2.1).

Remark 4.3. Algorithm 4.2 is essentially a depth-first search algorithm with the "search" operation carried out by the least squares Newton's method, and the order of the search between sibling nodes determined by the reordering scheme. With each successful descent on the search tree via the execution of lines 6–7, a quadrature with one fewer nodes is created, and the search for a quadrature with still fewer nodes starts. Special considerations such as location of the nodes, positivity of weights, and symmetry may be incorporated into the reordering scheme as well. For instance, preference of nodes that lie within the boundary of the domain may be formulated as assigning a negative number to s_j if $\mathbf{x}_j$ is outside.

Remark 4.4. There is no guarantee that the Newton's method will continue to converge as the number of quadrature nodes decreases. This, in combination with the fact that the number of nodes required by the optimal quadrature in two and higher dimensions is generally unknown, complicates the issue of determining what the final quadrature should be. In our experiments, we have found that random restarts with different initial quadratures (see Section 4.2 below) sometimes help reduce the number of nodes used by the final quadrature. As can be seen in Section 5, our final quadratures typically use one or two nodes more than the expected optimum in two and three dimensions.

4.2. The initial quadrature

In this section, we present a method for the construction of the initial quadrature, which is needed to bootstrap the Newton's method in the node elimination algorithm. As is well known, Newton's method is sensitive to the initial approximation, and converges only when the approximation is close to a true solution. In this section, we show that the initial quadrature constructed using the pivoted Gram–Schmidt method (see Section 3.4) from certain tensor product rules provides a stable initial approximation for our algorithm.

As is seen in Section 2.2, a p^d-point tensor product rule with nodes $\mathbf{x}_1, \ldots, \mathbf{x}_n$ and weights $w_1, \ldots, w_n$ ($n = p^d$) on $\Omega \subset \mathbb{R}^d$ may be constructed from a p-point one-dimensional generalized Gaussian quadrature integrating $\Phi_1, \Phi_2, \ldots, \Phi_{2p}$ accurately. The resulting tensor product rule is accurate for all functions in the set

$$\pi = \{f_{i_1,i_2,\ldots,i_d}(x_1, x_2, \ldots, x_d) | f_{i_1,i_2,\ldots,i_d}(x_1, x_2, \ldots, x_d) = \Phi_{i_1}(x_1)\Phi_{i_2}(x_2)\cdots\Phi_{i_d}(x_d)\},$$

with $i_1, i_2, \ldots, i_d = 1, \ldots, 2p$.

Suppose that $\phi_1, \ldots, \phi_m$ is a basis of π and that $p^d \gg m$. Defining an $m \times p^d$ matrix $A = (a_{j,k}) = \sqrt{w_k}\,\phi_j(\mathbf{x}_k)$ such that

$$A = \begin{bmatrix} \sqrt{w_1}\phi_1(\mathbf{x}_1) & \sqrt{w_2}\phi_1(\mathbf{x}_2) & \ldots & \sqrt{w_l}\phi_1(\mathbf{x}_l) & \ldots & \sqrt{w_{p^d}}\phi_1(\mathbf{x}_{p^d}) \\ \sqrt{w_1}\phi_2(\mathbf{x}_1) & \sqrt{w_2}\phi_2(\mathbf{x}_2) & \ldots & \sqrt{w_l}\phi_2(\mathbf{x}_l) & \ldots & \sqrt{w_{p^d}}\phi_2(\mathbf{x}_{p^d}) \\ \vdots & \vdots & & \vdots & & \vdots \\ \sqrt{w_1}\phi_m(\mathbf{x}_1) & \sqrt{w_2}\phi_m(\mathbf{x}_2) & \ldots & \sqrt{w_l}\phi_m(\mathbf{x}_l) & \ldots & \sqrt{w_{p^d}}\phi_m(\mathbf{x}_{p^d}), \end{bmatrix} \tag{23}$$

we can see that rows of A are orthonormal. Furthermore, there exist m columns of A indexed by $s_1, \ldots, s_m$ such that they form a basis for the column space of A. In other words, A_s, the m by m sub-matrix of A given by the formula

$$A_s = \begin{bmatrix} \sqrt{w_{s_1}}\phi_1(\mathbf{x}_{s_1}) & \sqrt{w_{s_2}}\phi_1(\mathbf{x}_{s_2}) & \ldots & \sqrt{w_{s_m}}\phi_1(\mathbf{x}_{s_m}) \\ \sqrt{w_{s_1}}\phi_2(\mathbf{x}_{s_1}) & \sqrt{w_{s_2}}\phi_2(\mathbf{x}_{s_2}) & \ldots & \sqrt{w_{s_m}}\phi_2(\mathbf{x}_{s_m}) \\ \vdots & \vdots & & \vdots \\ \sqrt{w_{s_1}}\phi_m(\mathbf{x}_{s_1}) & \sqrt{w_{s_2}}\phi_m(\mathbf{x}_{s_2}) & \ldots & \sqrt{w_{s_m}}\phi_m(\mathbf{x}_{s_m}) \end{bmatrix} \tag{24}$$

is invertible. In addition, it can be shown (see Theorem 4.5 below) that certain choices of $s_1, s_2, \ldots, s_m$ lead to quadratures with bounded weights. The reader is referred to [12] for a proof of this theorem, which was presented in a slightly different form.

Theorem 4.5. *Suppose that $\Omega \in \mathbb{R}^d$ contains at least m points, and that functions $\phi_1, \ldots, \phi_m$ are defined and integrable on Ω. Suppose further that A is defined by (23) and that $\mathbf{x}_{s_1}, \mathbf{x}_{s_2}, \ldots, \mathbf{x}_{s_m}$ are the m points selected by the pivoted Gram–Schmidt procedure given in Algorithm 3.5. Then, the system (2) has a unique solution $\mathbf{x}_{s_1}, \mathbf{x}_{s_2}, \ldots, \mathbf{x}_{s_m}, w_1, \ldots, w_m$, and there exists a positive constant $\varepsilon < 1$ such that w_k satisfies the condition*

$$|w_k| \leq 1 + \varepsilon \tag{25}$$

for all $k = 1, \ldots, m$.

Remark 4.6. The estimate of w_k in Theorem 4.5 is actually a pessimistic one. In our experiments, w_k's are not only bounded as indicated above but also significantly smaller. Nevertheless, a more detailed quantification of the magnitude of the w_k's is beyond the scope of this paper. The boundedness of w_k implies that constructed quadratures are numerically stable.

Remark 4.7. The user-defined parameter p, which determines the order of the one-dimensional generalized Gaussian quadrature, can be varied according to the specific needs of the application. In addition, we usually supplement the tensor product rules with additional points on the medians and at the center of the integration region to artificially introduce special nodes in the initial quadrature.

5. Numerical results

We have implemented in Fortran the numerical algorithms presented above, and obtained quadratures for a number of two- and three-dimensional regions to double precision accuracy. In this section, we present results related to integration of polynomials on the triangle, the square, and the cube.

We first present results for the standard triangle T in $\mathbb{R}^2$. As is shown in Example 3.1, the symmetry group G of T has six subgroups S_1 through S_6, each of which imposes a certain symmetry on its point sets. For example, point sets that are S_2- (or S_3-or S_4-) symmetric possess reflective symmetry, point sets that are S_5-symmetric possess rotational symmetry, and point sets that are S_6-symmetric possess both rotational and reflective (or total) symmetry. We show in Fig. 1 the S_6-, S_5-, and S_2-generators (the right triangle *BOF*, the equilateral triangle *AOC*, and the right triangle *ACF*, respectively) that we have used in our program; they are the domain Ω in the construction of totally symmetric, rotationally symmetric and reflectively symmetric quadratures on T, respectively.

A summary of the three types of quadratures is given in Table 1, where the degree (up to 50) of each quadrature is listed in columns labelled d, and the total numbers of quadrature nodes required by the final quadratures under each symmetry category are listed in columns labelled n_2 (reflectively symmetric), n_3 (rotationally symmetric), and n_6 (*totally symmetric*), respectively. All quadratures reported in this table have positive weights and interior (or boundary) nodes only. Quadratures of this type typically require more nodes than those that possess either negative weights or exterior nodes.

Table 1
A summary of the total number of nodes required by three types of symmetric quadrature rules on T: reflectively symmetric, rotationally symmetric, and totally symmetric (i.e., both rotationally and reflectively symmetric). For each degree d between 1 and 50, the total number of nodes on T for each of the three types is listed under n_2, n_3, and n_6, respectively. For degrees 11 through 30, the minimum numbers of nodes from previously known totally symmetric rules are listed in column p. Quadratures with underlines are apparently new.

d	n_2	n_3	n_6	d	n_2	n_3	n_6	p	d	n_2	n_3	n_6	p	d	n_6	d	n_6
1	1	1	1	11	27	27	28	28	21	85	87	87	93	31	181	41	309
2	3	3	3	12	32	33	33	33	22	95	93	96	100	32	193	42	324
3	4	6	6	13	35	36	37	37	23	102	102	103	106	33	204	43	339
4	6	6	6	14	41	42	42	42	24	110	111	112	118	34	214	44	354
5	7	7	7	15	48	46	49	54	25	119	118	120	126	35	228	45	370
6	11	12	12	16	52	52	55	58	26	127	129	130	138	36	243	46	385
7	13	12	15	17	58	58	60	61	27	137	136	141	145	37	252	47	399
8	16	16	16	18	65	66	67	73	28	148	147	150	154	38	267	48	423
9	19	19	19	19	71	72	73	73	29	157	156	159	166	39	282	49	435
10	24	24	25	20	79	78	79	85	30	168	168	171	175	40	295	50	453

Table 2
Efficiency of the quadrature rules summarized in Table 1. For each degree d between 1 and 50, efficiency of the quadratures of reflective symmetry, of rotational symmetry, and of total symmetry are listed under columns e_2, e_3, and e_6, respectively.

d	e_2	e_3	e_6	d	e_2	e_3	e_6	d	e_6	d	e_6
1	1.000	1.000	1.000	16	0.981	0.981	0.927	31	0.972	46	0.977
2	0.667	0.667	0.667	17	0.983	0.983	0.950	32	0.969	47	0.983
3	0.833	0.556	0.556	18	0.974	0.960	0.945	33	0.972	48	0.965
4	0.833	0.833	0.833	19	0.986	0.972	0.959	34	0.981	49	0.977
5	1.000	1.000	1.000	20	0.975	0.987	0.975	35	0.974	50	0.976
6	0.849	0.778	0.778	21	0.992	0.969	0.969	36	0.964		
7	0.923	1.000	0.800	22	0.968	0.989	0.958	37	0.980		
8	0.938	0.938	0.938	23	0.980	0.980	0.971	38	0.974		
9	0.965	0.965	0.965	24	0.985	0.976	0.967	39	0.969		
10	0.917	0.917	0.880	25	0.983	0.992	0.975	40	0.973		
11	0.963	0.963	0.929	26	0.992	0.977	0.969	41	0.973		
12	0.948	0.919	0.919	27	0.988	0.995	0.960	42	0.974		
13	1.000	0.972	0.946	28	0.980	0.986	0.967	43	0.973		
14	0.976	0.952	0.952	29	0.987	0.994	0.975	44	0.974		
15	0.944	0.986	0.925	30	0.984	0.984	0.967	45	0.975		

The quadratures that are totally symmetric on the triangle have been studied extensively, and results for relatively high orders are available (see, for example, [5,8]). In the column p of Table 1, we summarize quadratures of this type that are previously known for degrees 11 through 30. As can be seen, most of our quadratures (denoted by underlines) between degrees 15 and 30 use at least one node fewer than those previously published, with some saving as many as eight nodes. For degrees higher than 30, the authors have failed to find comparable rules (i.e., rules that are totally symmetric, and have only positive weights with interior or boundary points). Finally, in Table 2, we calculate the efficiency E (to three digits) of the quadratures listed in Table 1. These results show that our rules have relatively high values of E, with E above 0.90 for all quadratures of degrees 10 and above, and with some as high as 0.99. A comparison of the quadratures that integrate the exact same set of polynomials but have different degrees of symmetry indicates that rules designed with fewer symmetric constraints tend to require fewer nodes in general.

In Table 3, we summarize results for non-symmetric quadratures on the standard 2 by 2 square in $\mathbb{R}^2$ up to degree 24. These quadratures are designed without any symmetry considerations. As before, the degree of each quadrature is listed under the column d. For each degree, the expected minimum number of nodes $\lceil m/(d+1)\rceil$ (see Section 2.1) is listed under the column *theo*, and the total number of nodes of the final quadrature we constructed is listed under n. As in the case for the triangle, all quadratures reported in this table have positive weights and interior (or boundary) nodes only. Although the formulae are designed to be non-symmetric, many of the final quadratures show a certain degree of symmetry, primarily due to the inherent symmetry of the integration region. As can be seen, the efficiency of these quadratures is, again, high. Typically, our rules use at most one additional node than what is expected. In several cases (noted by asterisks), our quadratures use even fewer nodes than expected.

A summary of the results for non-symmetric quadratures on the standard 2 by 2 by 2 cube in $\mathbb{R}^3$ up to degree 15 is included in Table 4. Quadratures reported in this table are constructed with no symmetric requirements imposed, and use interior (or boundary) nodes only. The degree of each quadrature is listed in row d, and the expected number of nodes is listed in row *theo*. The total number of quadrature nodes needed by the final quadrature computed by our algorithm for each degree is listed in row n. Except for the final rule for degree 14, which contains one negative weight, all other quadratures listed here possess positive weights only. Again, although the formulae are designed to be non-symmetric, many of the final quadratures show a certain degree of symmetry. As can be seen, the efficiency of these quadratures is relatively high with most rules using at most two nodes more than expected. For degrees 7, 9, and 11, the quadratures that we have constructed

Table 3
A summary of the results for non-symmetric quadratures on the square for degrees up to 24. For each degree listed under *d*, the expected number of nodes is listed under *theo*, and the total number of nodes of the final quadrature is listed under *n*. Asterisks denote quadratures that use fewer nodes (*n*) than expected (*theo*).

d	*theo*	*n*	*d*	*theo*	*n*
1	1	1	13	35	35
2	2	3	14	40	40
3	4	4	15*	46	45
4	5	6	16	51	52
5	7	7	17	57	57
6	10	10	18	64	64
7	12	12	19	70	71
8	15	16	20	77	78
9*	19	18	21	85	85
10	22	22	22	92	93
11	26	27	23	100	101
12	31	31	24	109	109

Table 4
A summary of the results for non-symmetric quadratures on the cube for degrees up to 15. For each degree listed under *d*, the expected number of nodes is listed under *theo*, and the total number of nodes of the final quadrature is listed under *n*. The quadrature with 173 nodes has one negative weight. Asterisks denote quadratures that use fewer nodes (*n*) than expected (*theo*).

d	1	2	3	4	5*	6	7	8	9*	10	11*	12	13	14	15
theo	1	3	5	9	14	21	30	42	55	72	91	114	140	170	204
n	1	4	6	10	13	22	28*	42	54	74	90	116	141	173*n*	206

Table 5
A 13-point non-symmetric quadrature of degree 5 on the standard cube.

x	*y*	*z*	Weight
0.0000000000000000	0.0000000000000000	0.0000000000000000	1.6842105263157894
0.8154162179225410	0.7011590353030981	−0.3318720618376825	0.5580769304856384
−0.8154162179225410	−0.7011590353030981	0.3318720618376825	0.5580769304856384
0.3354972826749644	0.8137174023181656	0.7014073204272666	0.5595478790816522
−0.3354972826749644	−0.8137174023181656	−0.7014073204272666	0.5595478790816522
−0.7204688849769894	0.3858783453882358	0.7737500597784259	0.5832800882215720
0.7204688849769894	−0.3858783453882358	−0.7737500597784259	0.5832800882215720
0.8786392187683029	−0.0478015313848859	0.7016942379029152	1.3916991226476483
−0.8786392187683029	0.0478015313848859	−0.7016942379029152	1.3916991226476483
0.6649493200879981	−0.8971504092175344	0.1401078571029694	0.4898898395473991
−0.6649493200879981	0.8971504092175344	−0.1401078571029694	0.4898898395473991
−0.0103664220183517	−0.6588164031477778	0.9124254221056255	0.4750600560080839
0.0103664220183518	0.6588164031477778	−0.9124254221056255	0.4750600560080839

use one or two nodes fewer than that is expected by the theoretical argument. It should be noted that the constructed quadratures for the cube use a significantly lower number of nodes than corresponding tensor product rules. For example, a tensor product rule of degree 15 needs at least 512 nodes, while our rule uses just over 200 nodes.

A sample 13-point quadrature of degree 5 on the cube is listed in Table 5, and sample quadratures for the triangle of degrees 10, 30, and 50 are tabulated in Tables 6–8, respectively. As is mentioned in Section 3.1, all symmetric quadratures are uniquely determined by their configuration on its symmetric generators. Therefore, only nodes that lie within the corresponding *S*-generators are listed in Tables 6–8; all remaining nodes may be generated via actions of the group elements in S_6, S_2 and S_5, respectively (see Example 3.1).

Finally, we show in Figs. 2 and 3 two sample nodal configurations, with Fig. 2 showing the 453 node quadrature listed in Table 8, and Fig. 3 depicting a 109 node quadrature of degree 24 on the square. Both figures show a certain resemblance of those of Gaussian quadratures in one dimension, with nodes distributed more densely near the boundaries of the integration region than in the center.

6. Conclusions

We present a numerical algorithm for the construction of highly efficient, high-order quadratures in two and higher dimensions. This method combines the least squares Newton's method and a node elimination scheme, and generates a sequence of quadratures that use fewer and fewer number of nodes. The resulting quadratures possess desirable properties such as positivity of weights, and interior nodes, resembling Gaussian quadratures in one dimension. Symmetry, a considerably richer subject in quadrature construction for higher-dimensional regions, is integrated naturally into the

Table 6
A reflectively symmetric quadrature of degree 10 with a total number of 24 nodes on T. Only the generating 14 nodes on either the generator triangle T_2 or the median are listed.

x	y	Weight
0.1551275866992061	0.3926587334507982	0.05567338455800167
0.9252314250365582	−0.5241973822167591	0.00896962036448401
0.1384253518864596	0.8231152771873217	0.02240118142259882
0.3609812845058779	0.0267765158868186	0.06089917117558221
0.5812005281770837	0.0484050181187623	0.03108755935951332
0.3458850891448130	0.4561437111102405	0.03146650526166210
0.4510519110813880	−0.3059662347573659	0.06246416621868210
0.6893778940829133	−0.5226600964300047	0.02582709676771077
0.7634762995170715	−0.3046970738749675	0.03178319217145733
0.2563805151012611	−0.5238003914950184	0.03640547863919981
0.0000000000000000	0.7059408419246426	0.03498602106696763
0.0000000000000000	1.0463216388186550	0.01202119786551837
0.0000000000000000	0.0240306520799389	0.07686209708916766
0.0000000000000000	−0.3100722765497620	0.07278083120266070

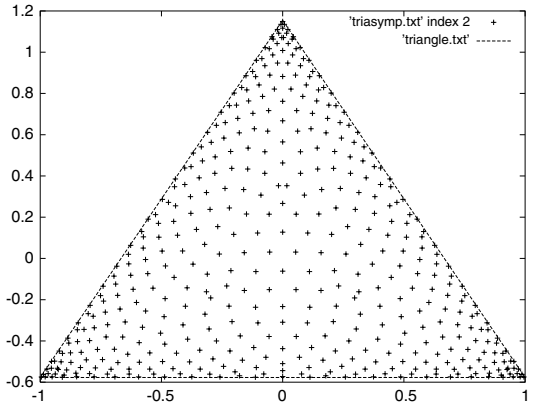

Fig. 2. A 453-point totally symmetric quadrature of degree 50 on the equilateral triangle T.

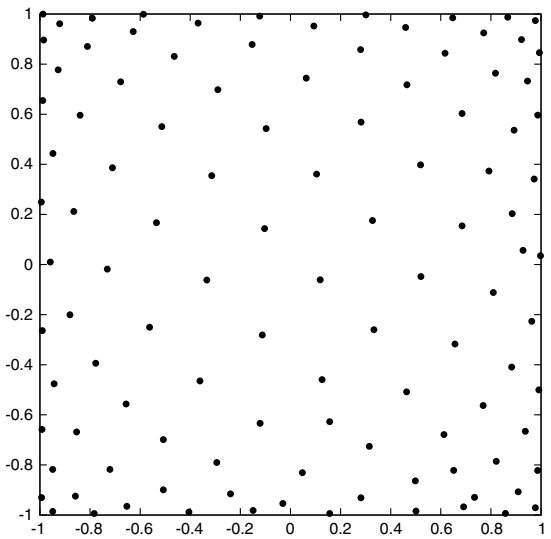

Fig. 3. A 109-point quadrature of degree 24 on the standard square.

Table 7
A rotationally symmetric quadrature of degree 30 with a total number of 168 nodes on T. Only the generating 56 nodes on either the generator triangle T_3 or the median are listed; the remaining 112 nodes may be obtained via rotations of these nodes by $2\pi/3$ and $4\pi/3$.

x	y	Weight
0.8272205847965533e−1	−0.2086440937022876e−1	0.8225724599112734e−2
0.6460592016667458e+0	−0.155719850752971e+0	0.3425246656362368e−2
0.5912391719771670e−1	0.1039914752424664e+1	0.6743805221766848e−3
0.7357619003526131e+0	−0.1341845875173481e+0	0.1261438394384131e−2
0.5761546120269558e+0	−0.513625080655618e−1	0.4918794158057610e−2
0.1994087632082849e+0	0.9377211520122531e−2	0.8224000212730418e−2
0.8765562348681995e+0	−0.455736738075614e+0	0.1937869814215937e−2
0.1386237601227312e+0	0.8232066932267047e+0	0.2818999309598719e−2
0.8771274146530530e+0	−0.378387545369664e+0	0.9936867294932528e−3
0.8367393599650243e+0	−0.3732813894029767e+0	0.2291488485753811e−2
0.9530921218500234e−1	0.1518657794513243e+0	0.9584455083925919e−2
0.3150312065314394e+0	0.2258787078498025e−1	0.8253156910035582e−2
0.6538072391711382e+0	0.5766254583042341e−2	0.1591561151450938e−2
0.7836086212486449e−1	0.9114490338273592e+0	0.2625273142248666e−2
0.5333723933200066e+0	−0.1420835800669319e+0	0.6576713150961314e−2
0.2113485001639861e+0	0.7021528320464188e+0	0.3441276278660254e−2
0.4201764494776592e−1	0.1007798049850320e+1	0.1802891190698994e−2
0.6955339994886459e+0	−0.1292191897482428e+0	0.3466499998769438e−2
0.4127226449687541e−1	0.8325715025940558e+0	0.4341724718219735e−2
0.1156522714200305e+0	0.7358780895538129e+0	0.4962835559191245e−2
0.4469844656691110e+0	−0.2382953838781011e−2	0.7398503863311476e−2
0.3529921026374059e+0	0.1578080210271638e+0	0.8023583585428551e−2
0.2460258923508316e+0	−0.1400047139425037e+0	0.9876856647926293e−2
0.9668893072398800e+0	−0.5375235554130982e+0	0.5589691156993659e−3
0.5330742228948119e+0	−0.2572390264377685e+0	0.6986365093669510e−2
0.6377880380051988e+0	−0.3605731102565961e+0	0.5630106441409166e−2
0.2985565943246146e+0	0.5504747584405043e+0	0.4032405218823953e−2
0.5627080639457788e+0	0.1630690759145039e+0	0.1820674630472769e−2
0.4861035354319526e+0	0.9943388269681425e−1	0.6268653378145729e−2
0.1241089996082145e+0	0.3341243654933032e+0	0.9280637865480141e−2
0.6642540651500375e+0	−0.2597847118974944e+0	0.5571424237523911e−2
0.5987333241573769e+0	0.3050210764280144e−1	0.4182326597878419e−2
0.2197532948080663e+0	0.1798196888453294e+0	0.9050760263478505e−2
0.2698852497770884e+0	0.6708183810109211e+0	0.1690915977597697e−2
0.3958421104006370e+0	0.3804317735351793e+0	0.4393772842367057e−2
0.9158780779011614e+0	−0.5245150065313966e+0	0.1507411066409142e−2
0.2558634471266385e+0	0.3251482772651994e+0	0.8206004882495090e−2
0.1838917766755212e+0	0.8193255390401215e+0	0.1507119774858036e−2
0.1974375794859356e+0	0.6013855940274377e+0	0.5765437083136701e−2
0.7584770267556704e+0	−0.2670670580928476e+0	0.3703392584101807e−2
0.3469765078485344e−1	0.4734263841039434e+0	0.8715934043390202e−2
0.4976940838283434e+0	0.2037394989534461e+0	0.4453198965263726e−2
0.3870703164854328e+0	0.2699318755098380e+0	0.6579627058432091e−2
0.9288658345029073e+0	−0.4725350228450654e+0	0.9516599016618329e−3
0.2889458004352184e+0	0.4416175187348937e+0	0.6362435785130976e−2
0.4645516229515598e+0	0.3330077787895797e+0	0.1913354619710829e−2
0.3650914806280184e+0	0.5055382016363809e+0	0.1857174714166686e−2
0.7473303039531310e−1	0.6275958705543054e+0	0.7084040821689384e−2
0.3924863580876331e+0	−0.1272914050208924e+0	0.8993525759622909e−2
0.1619393302508105e+0	0.4863704550920276e+0	0.7900965369909250e−2
0.1935821149964777e−1	0.1100542011523507e+1	0.6685162557274974e−3
0.7609746208061138e+0	−0.3765326388259521e+0	0.4547482851553974e−2
0.8060584904241093e+0	−0.2621917384930423e+0	0.1659224167519006e−2
0.1114826626855052e+0	0.9415786826470525e+0	0.1417990273592256e−2
0.8581107053895953e−3	0.1139838338986214e+1	0.1916747284979650e−3
0.2769231349292084e−2	0.9429639395524531e+0	0.3108419351734294e−2

algorithm, where the degree of symmetry of the quadrature is a user specified parameter. Using this algorithm, we have constructed quadratures for polynomials on the triangle, the square and the cube which are more efficient than any formulae previously known. Many of the rules we report here appear to be new. The quadratures obtained also have corresponding interpolation schemes, which will be reported separately.

Although our method has generated satisfactory results in two and three dimensions, the approach is experimental in nature. Theoretical analysis on the optimality of higher-dimensional quadratures would be highly desirable.

Table 8
A 453-point totally symmetric quadrature of degree 50 on the equilateral triangle T. Only the generating 84 nodes on either the S_6-generator T_6 or the median AF (see Fig. 1) are listed; the remaining nodes may be obtained via group action of S_6.

x	y	Weight
0.0000000000000000	0.2666577779779638	0.0021904046082402
−0.1124313701772454	−0.5757525283461596	0.0004571095646746
−0.9256882209861061	−0.5384865283035131	0.0003948881417073
−0.2969210611797281	−0.1914654083296075	0.0035934111950642
−0.7077745499764682	−0.5373048839112955	0.0014910742851108
−0.9090720752703281	−0.5758662830537718	0.0001796728471361
−0.1777622292035842	−0.5690214644458064	0.0010125558476824
−0.6335800721605949	−0.5366591781602346	0.0017069127169250
0.0000000000000000	1.1490520500034340	0.0000240796957403
−0.4863915034605701	−0.4155825311049862	0.0036927130629621
−0.6796097231116817	−0.5049563709243417	0.0022489391525853
−0.8098785802816203	−0.4918608972661665	0.0019524312574849
0.0000000000000000	−0.5438415613297470	0.0010866799154497
−0.1019729166748508	−0.3126684717312953	0.0055988347396351
−0.6390531355140903	−0.4645670361930981	0.0028924669217577
−0.5484885696606917	−0.5367126451543023	0.0020380484585475
−0.2874329156493009	−0.4872277196540656	0.0034261098541603
−0.2585357538056598	−0.5747616861659614	0.0006135145701974
−0.8529331149785352	−0.5564898875850632	0.0008654202070143
−0.0691175320963452	−0.4791693406122051	0.0036671947134331
0.0000000000000000	−0.3091843545841412	0.0028026432284100
−0.0561154156396137	−0.3784066674034268	0.0054349341984383
−0.2439741392754361	−0.5231979844239280	0.0028351957640421
−0.2732507888320513	−0.4002538935543359	0.0045746103628843
0.0000000000000000	−0.5100624385517909	0.0015952659260343
−0.1662720547195513	−0.4936068218349784	0.0035621046832988
−0.9378927147457812	−0.5600651349094425	0.0005077140943325
−0.1193050675401607	−0.4320087923635920	0.0045268556007289
−0.9015110856568858	−0.5656968032429866	0.0005764758688778
−0.3204953594372978	−0.2702437615902795	0.0057986412207422
−0.5910420443956919	−0.5034192605577299	0.0026551803851187
−0.2175146254032372	−0.4484982152285935	0.0042618143181478
−0.1580090779826294	−0.1554764764590674	0.0061410787717774
−0.3419299642831888	−0.4450361891196334	0.0041577749997957
−0.5034831466244999	−0.3621869683337191	0.0045714943165212
−0.7229666416869851	−0.5604125298441649	0.0011784033012900
−0.8341041276554126	−0.5289512235160428	0.0014219896538536
−0.4522096929495317	−0.5343810876810189	0.0023860085586204
−0.8909150874538825	−0.5407589630878091	0.0010070034152583
−0.4379356663313332	−0.4593112313803924	0.0037609206282914
−0.5479900871398830	−0.4581933023157792	0.0034952452756601
−0.2088545582629160	−0.3065381028776907	0.0058926392977540
−0.1965326162646397	−0.5500033153935602	0.0021761510975303
−0.3402642556071983	−0.5367398993353952	0.0025597503150258
−0.3033412114126202	−0.3419129259938245	0.0053360452230413
−0.4953529428186446	−0.4999352242884132	0.0030890849879669
−0.0549674992365703	−0.1511683414760968	0.0071328535685235
−0.7964198114498382	−0.5587073875420177	0.0010810483517258
−0.7725262143408313	−0.5309351704947155	0.0017544886347878
−0.2212226571925028	−0.2311104433727553	0.0061494904649466
0.0000000000000000	−0.4373547401936403	0.0023575565603961
−0.6331072992422112	−0.5606190171869835	0.0013783891229758
0.0000000000000000	0.7621026989631550	0.0018290788738495
−0.3840758657118736	−0.5018046405596390	0.0033652181243651
−0.5317084125284348	−0.5606640458276943	0.0015184413550438
−0.1751172055153042	−0.3756408245656322	0.0055587775921145
−0.3006341397609248	−0.5609358364702884	0.0017601404353050
−0.1086869220568949	−0.5289106688491748	0.0031341293249895
−0.4111397082995238	−0.3222783448212513	0.0056365508134121
0.0000000000000000	1.1168066855539390	0.0001973796391634
−0.6917837628935971	−0.5741103522607572	0.0005799858664965
0.0000000000000000	0.8822615489186989	0.0013607360815822
−0.9795356262898514	−0.5737698773740506	0.0001521977184470
−0.9509031506431555	−0.5738751156319078	0.0002400849357944
−0.1119510482835352	−0.2368429502002873	0.0067581925200909
−0.3921825162734547	−0.3911914123606699	0.0051393423392662
−0.4870384654434685	−0.5741255207957627	0.0006951734036825
−0.7432058368831903	−0.4903886875105679	0.0024993080895095

(*continued on next page*)

Table 8 (*continued*)

x	y	Weight
0.0000000000000000	0.5649064689901148	0.0026503828818360
−0.6939356748620450	−0.4403471863506789	0.0030717948054284
−0.5941680920408391	−0.5741825739834173	0.0006352303361032
0.0000000000000000	0.4631499494550457	0.0030076725635797
−0.5916869526417282	−0.4084779816408744	0.0042759500648291
−0.4222627055622458	−0.5595603013208083	0.0017369109423123
0.0000000000000000	−0.2332573579986706	0.0033989953518860
0.0000000000000000	0.6675997901497540	0.0022934322638873
−0.7777797118015541	−0.5739813147120367	0.0005221250706383
0.0000000000000000	−0.0618585703169292	0.0038540006507433
0.0000000000000000	1.0083409893504060	0.0007889526686353
−0.3747333122781531	−0.5738641446709915	0.0007706843098833
0.0000000000000000	−0.5727911592067401	0.0004930816413700
−0.0779889206854663	−0.5602379116800536	0.0018435234238175
0.0000000000000000	0.1276641504717413	0.0037992680605946
−0.8496556980932881	−0.5734944598728055	0.0004876127244490

Acknowledgements

The authors would like to thank Professor V. Rokhlin for his advice and support. The first author was supported in part by NSF under grant No. DMS-051609. The second author was supported by the AFOSR under MURI grant FA9550-06-1-0337 and in part by a research grant from Meyer Sound.

References

[1] H. Cheng, V. Rokhlin, N. Yarvin, Nonlinear optimization, quadrature, and interpolation, SIAM J. Optim. 9 (4) (1999) 901–923.
[2] J. Ma, V. Rokhlin, S. Wandzura, Generalized Gaussian quadrature of systems of arbitrary functions, SIAM J. Numer. Anal. 33 (3) (1996) 971–996.
[3] N. Yarvin, V. Rokhlin, Generalized Gaussian quadratures and singular value decompositions of integral operators, SIAM J. Sci. Comput. 20 (2) (1998) 669–718.
[4] J. Berntsen, T.O. Espelid, Degree 13 Symmetric Quadrature Rules for the Triangle, Reports in Informatics 44, Dept. of Informatics, University of Bergen, 1990.
[5] J.N. Lyness, D. Jespersen, Moderate degree symmetric quadrature rules for the triangle, J. Inst. Math. Appl. 15 (1975) 19–32.
[6] P. Silvester, Symmetric quadrature formulae for simplexes, Math. Comput. 24 (1970) 95–100.
[7] A.H. Stroud, Approximate Calculation of Multiple Integrals, Prentice-Hall, 1971.
[8] S. Wandzura, H. Xiao, Symmetric quadrature rules on a triangle, Comput. Math. Appl. 45 (2003) 1829–1840.
[9] J. Stoer, R. Bulirsch, Introduction To Numerical Analysis, 2nd ed., Springer-Verlag, 1992.
[10] V.H. Golub, C.H. Van Loan, Matrix Computations, Johns Hopkins University Press, Baltimore, 1983.
[11] T. Koornwinder, Two-variable analogues of the classical orthogonal polynomials, in: R.A. Askey (Ed.), in: Theory and Applications of Special Functions, Academic Press, 1975, pp. 435–495.
[12] P.G. Martinsson, V. Rokhlin, M. Tygert, On interpolation and integration in finite-dimensional spaces of bounded functions, in: Communications in Applied Mathematics and Computational Science, vol. 1, 2006.