Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 338 (2018) 134–161
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

Generation and application of multivariate polynomial quadrature rules

John D. Jakeman[a,*], Akil Narayan[b]

[a] *Center for Computing Research, Sandia National Laboratories, Albuquerque, NM, USA*
[b] *Department of Mathematics, and Scientific Computing and Imaging (SCI) Institute, University of Utah, Salt Lake City, UT, USA*

Received 24 August 2017; received in revised form 30 January 2018; accepted 3 April 2018
Available online 25 April 2018

Abstract

The search for multivariate quadrature rules of minimal size with a specified polynomial accuracy has been the topic of many years of research. Finding such a rule allows accurate integration of moments, which play a central role in many aspects of scientific computing with complex models. The contribution of this paper is twofold. First, we provide novel mathematical analysis of the polynomial quadrature problem that provides a lower bound for the minimal possible number of nodes in a polynomial rule with specified accuracy. We give concrete but simplistic multivariate examples where a minimal quadrature rule can be designed that achieves this lower bound, along with situations that showcase when it is not possible to achieve this lower bound. Our second contribution is the formulation of an algorithm that is able to efficiently generate multivariate quadrature rules with positive weights on non-tensorial domains. Our tests show success of this procedure in up to 20 dimensions. We test our method on applications to dimension reduction and chemical kinetics problems, including comparisons against popular alternatives such as sparse grids, Monte Carlo and quasi Monte Carlo sequences, and Stroud rules. The quadrature rules computed in this paper outperform these alternatives in almost all scenarios.

Keywords: Multivariate integration; Moment-matching quadrature; Uncertainty quantification; Dimension reduction; Bayesian inference

1. Introduction

Let $D \subset \mathbb{R}^d$ be a domain with nonempty interior. Given a finite, positive measure μ on D and a μ-measurable function $f : D \to \mathbb{R}$, our main goal is computation of

$$\int_D f(x)\mathrm{d}\mu(x)$$

* Corresponding author.
E-mail address: jdjakem@sandia.gov (J.D. Jakeman).

https://doi.org/10.1016/j.cma.2018.04.009

The need to evaluate such integrals arise in many areas, including finance, stochastic programming, robust design and uncertainty quantification. Typically these integrals are approximated numerically by quadrature rules of the form

$$\int_D f(x)\mathrm{d}\mu(x) \approx \sum_{m=1}^{M} w_m f\left(x_m\right) \tag{1}$$

Examples of common quadrature rules for high-dimensional integration are Monte Carlo and quasi Monte Carlo methods [1–4], and Smolyak integration rules [5–7]. Quadrature rules are typically designed and constructed in deference to some notion of approximation optimality. The particular approximation optimality that we seek in this paper is based on polynomial approximation.

Our goal is to find a set of quadrature nodes $x_j \in D$, $j = 1, \ldots, M$, and a corresponding set of weights $w_j \in \mathbb{R}$ such that

$$\sum_{j=1}^{M} w_j p\left(x_j\right) = \int_D p(x)\mathrm{d}\mu(x), \qquad p \in P_\Lambda, \tag{2}$$

where $\Lambda \subset \mathbb{N}_0^d$ is a multi-index set of size N, and P_Λ is an N-dimensional polynomial space defined by Λ. (We make this precise later.) P_Λ can be a relatively "standard" space, such as the space of all d-variate polynomials up to a given finite degree, or more exotic spaces such as those defined by ℓ^p balls in index space, or hyperbolic cross spaces.

In this paper we will present a method for numerically generating polynomial based cubature rules. This paper provides two major contributions to the existing literature. Firstly we provide a lower bound on the number points that make up a polynomial quadrature rule satisfying (2). Our analysis is straightforward, but to the authors knowledge this is the first reported bound of its kind. Our second contribution is a numerical method for generating quadrature rules that are exact for a set of arbitrary polynomial moments. Our method has the following features:

- Positive quadrature rules are generated. (I.e., $w_m > 0$ for all m.)
- The algorithm applies to any measure μ for which moments are computable. Many existing quadrature methods only apply to tensor-product measures; for non-tensorial integrals this requires construction of mappings to transform integrals over non-tensorial domains to integrals over tensorial domains. Our method can construct quadrature rules for measures with, for example, non-linear dependence between variables, without the use of mappings or transformations.
- Analytical or sample-based moments may be used. In some settings it may be possible to compute moments of a measure exactly, but in other settings only samples from the measure are available. For example, one may wish to integrate a function using Markov Chain Monte Carlo-generated samples from a posterior of a Bayesian inference problem.
- A quadrature that is faithful to arbitrary sets of moments may be generated. Many quadrature methods are exact for certain polynomial spaces, for example total-degree or sparse grid spaces. However, some functions may be more accurately represented by alternative polynomial spaces, such as hyperbolic cross spaces. In these situations it may be more prudent to construct rules that can match a customized set of moments.
- Efficient integration of ridge functions is possible. Some high-dimensional functions can be represented by a small number of linear combinations of the input variables. In this case it is more efficient to integrate these functions over this lower-dimensional coordinate space. Such a dimension-reducing transformation typically induces a new measure on a non-tensorial space of lower-dimensional variables. For example, a high-dimensional uniform probability measure on a hypercube may be transformed into a non-uniform measure on a zonotope (a multivariate polygon).

Our algorithm falls into the class of moment-matching methods. There have been some recent attempts at generating quadrature using moment matching via optimization approaches. These methods frequently either start with a small candidate set and add points until moments are matched [8], or start with a large set of candidate points and reduce them until no more points can be removed without numerically violating the moment conditions [9–11]. These approaches sometimes go by other names, such as scenario generation or scenario reduction methods. Our numerical strategy is novel and our results show that it is rather effective, but we do not ameliorate persistent bottlenecks in computing multivariate quadrature rules: The difficulty of computing high-degree quadrature rules scales commensurately with the dimension of the polynomial space, and this dimension can generally grow

exponentially with dimension. Also, in high dimensions we require nonconvex optimization in high dimensions, and cannot guarantee that our procedure converges to a global minimum.

This paper presents a quadrature/scenario reduction moment matching method based upon the work in [9]. The method in [9] is comprised of two steps. The first step generates a positive quadrature rule with $M = N$ points (N is the dimension of P_Λ in (2)), the existence of which is guaranteed by Tchakaloff's theorem [12]. The second step uses this quadrature rule as an initial guess for a local gradient-based optimization procedure that searches for a quadrature rule with $M < N$ points.

The initial quadrature rule is generated by drawing a large number of points from the domain of integration and then solving an inequality-constrained linear program to find a quadrature rule with positive weights that matches all desired moments. Similar approaches can be found in [13,14]. The N points comprising this initial quadrature rule are then grouped into $M \leq N$ clusters. A new approximate quadrature rule is then formed by combining points and weights within a cluster into a single point and weight. Our numerical method differs from [9] in the following ways: (i) we use a different formulation of the linear program to generate the initial quadrature rule; (ii) we show numerically that this quadrature only needs to be solved with very limited accuracy, (iii) we present an automated way of selecting the clusters from the initial rule — in [9] no method for clustering points is presented; (iv) we provide extensive numerical testing of our method in a number of settings.

The theoretical contributions of this paper include a lower bound on the number of points in the final quadrature rule, and a lower bound for the size of minimal quadrature rules (i.e., minimal quadrature rules are those the smallest possible number of nodes for a specified accuracy). We also provide a simple means of testing whether the quadrature rules generated by any method are minimal.

The remainder of the paper is structured as following. In Section 2 we introduce some nomenclature, derive our lower bound, and present a means to verify if a quadrature rule is minimal. We also use these theoretical tools to analytically derive minimal rules for multivariate functions that are sums of univariate functions. In Section 4 we detail our algorithm for generating quadrature rules. We then present a wide range of numerical examples, in Section 5, which explore the properties of the quadrature rules that our numerical algorithm can generate.

1.1. Existing quadrature rules

In this section we focus our review on polynomial-based quadrature rules. Although other types of quadrature rules certainly exist, we only mention them briefly when a comparison against polynomial-based approaches is relevant.

For univariate functions Gaussian quadrature is one of the most commonly used approaches. Nodes associated to Gaussian quadrature rules are prescribed by roots of polynomials pairwise orthogonal with respect to the measure μ [15]. The resulting quadrature rule is always positive and the rules are optimal in the sense that given a Gaussian quadrature rule of degree of exactness k, no rule with fewer points can be used to exactly integrate all degree-k polynomials.

When integrating multivariate functions with respect to tensor-product measures on a hypercube accurate and efficient quadrature rules can be found by taking tensor-products of one-dimensional Gaussian quadrature rules. These rules will be optimal for functions that can be represented exactly by tensor-product polynomial spaces of degree p. The accuracy of tensor-product integration of (1) generally scales like $M^{-r/d}$, where r indicates the maximum order of continuous partial derivatives in any direction [16]. In practice the use of tensor-product quadrature rules is limited to a small number of dimensions, say 3–4, because the number of the points in the rule grows exponentially with the dimension. Finally, if μ and/or D are non-tensorial, but may be smoothly mapped to a tensorial domain $\tilde{D}$ and measure $\widetilde{\mu}$, then a common strategy is to generate a quadrature rule on $(\tilde{D}, \widetilde{\mu})$ and map it back to (D, μ) for use there. This, of course, requires the ability to map (D, μ) to a tensorial configuration, which can be challenging in multiple dimensions.

Sparse grid quadrature methods have been successfully used as an alternative to tensor-product quadrature for multivariate functions [5–7]. Sparse grid quadrature delays the curse of dimensionality by focusing on integrating polynomial spaces that have high-degree univariate terms but low-degree interaction terms. For sparse grids consisting of univariate quadrature rules with $O(2^l)$ points at level l, the number of points in the level-l sparse grid scales like $O(2^l l^{d-1})$ [5]. For some functions of interest sparse grids can achieve a similar error to that obtained when using tensor product quadrature rules of level l; however unlike tensor-product rules the quadrature weights will not all be positive.

High-dimensional cubature rules can often be more effective than sparse grid rules when integrating functions that are well represented by total-degree polynomials. These rules have positive weights and typically consist of a very small number of points. However such highly effective cubature rules are difficult to construct and are have only been derived for a specific set of measures, integration domains and polynomial degree of exactness [17–20].

Polynomial based quadrature rules are useful when the integrand f has high-regularity. However when the function has less regularity, such as piecewise continuity, alternative rules based upon other basis functions, such as piecewise polynomials may be more effective. For example, composite quadrature rules are popular: Simpson's rule for univariate functions; sparse grids, based upon piecewise polynomials, for multivariate functions [21]; and the numerous adaptive versions of these methods e.g. [22]

When the above quadrature methods become intractable due to the curse of dimensionality, Monte Carlo (MC) and quasi Monte Carlo (QMC) approaches can be effective. These approximations produce convergence rates of $O(M^{-\frac{1}{2}})$ and $O(\log(M)^d M^{-1})$, respectively [23]. MC points are random, selected as independent and identically-distributed realizations of a random variable, and QMC points are deterministically generated as sequences that minimize discrepancy.

2. Minimality in multivariate polynomial quadrature

A quadrature rule is minimal if no other rule exists with a smaller number of nodes can have identical (or greater) accuracy. The goal of this section is to mathematically codify relationships between minimality, the number of quadrature points M and the dimension N of the polynomial space P_Λ. In particular, fixing Λ, we provide a theoretical lower bound for M and show that achieving the lower bound is a sufficient condition for minimality of a quadrature rule, but minimal quadrature rules can have size exceeding this lower bound (see Section 3).

2.1. Notation

With $d \geq 1$ fixed, we consider a positive measure μ on $\mathbb{R}^d$ with support $D = \text{supp } \mu$. This support may be unbounded. The $L^2_\mu(D)$ inner product and norm are defined as

$$\langle f, g\rangle_\mu := \int_D f(x)g(x)\mathrm{d}\mu(x), \qquad \|f\|^2_\mu = \langle f, f\rangle_\mu, \qquad L^2_\mu(D) = \left\{f : D \to \mathbb{R} \mid \|f\|^2_\mu < \infty\right\}$$

To prevent degeneracy of polynomials with respect to μ and to ensure finite moments of μ we assume

$$0 < \|q\|_\mu < \infty, \tag{3}$$

for all nonzero algebraic polynomials $q(x)$. One can guarantee the lower inequality if, for example, there is any open Euclidean ball in D inside which μ has a positive density function.

Let the coordinate representation of a point x in $\mathbb{R}^d$ be $x = \left(x^{(1)}, \ldots, x^{(d)}\right) \in \mathbb{R}^d$, then for a multi-index $\alpha \in \mathbb{N}_0^d$ with coordinates $\alpha = \left(\alpha^{(1)}, \ldots, \alpha^{(d)}\right)$, we have $\alpha! = \prod_{j=1}^d \alpha^{(j)}!$ and $x^\alpha = \prod_{j=1}^d \left[x^{(j)}\right]^{\alpha^{(j)}}$.

If $\alpha, \beta \in \mathbb{N}_0^d$ are any two multi-indices and $k \in \mathbb{R}$, we define $\alpha + \beta$, $k\alpha$, and $\lfloor k\alpha \rfloor$ component wise. The partial ordering $\alpha \leq \beta$ is true if all the component wise conditions are true. Letting $\Lambda \subset \mathbb{N}_0^d$ denote a multi-index set of finite size N, we define the following standard properties and operations on multi-index sets:

Definition 2.1. Let Λ and Θ be two multi-index sets, and let $k \in [0, \infty)$.

(Minkowski addition) The sum of two multi-index sets is

$$\Lambda + \Theta = \{\alpha + \beta \mid \alpha \in \Lambda,\ \beta \in \Theta\}$$

(Scalar multiplication) The expression $k\Lambda$ is defined as

$$k\Lambda = \{k\alpha \mid \alpha \in \Lambda\}.$$

Note that this need not be a set of multi-indices.

(Downward closed) Λ is a downward closed set if $\alpha \in \Lambda$ implies that $\beta \in \Lambda$ for all $\beta \leq \alpha$.

(Downward closure) For any finite Λ, $\bar{\Lambda}$ is the smallest downward closed set containing Λ,

$$\bar{\Lambda} = \left\{\alpha \in \mathbb{N}_0^d \mid \alpha \leq \beta \text{ for some } \beta \in \Lambda\right\}.$$

(Convexity) Λ is convex if for every $\alpha, \beta \in \Lambda$ and for all $\lambda \in [0,1]$ such that $\lambda\,\alpha + (1-\lambda)\beta \in \mathbb{N}_0^d$, then $\lambda\alpha + (1-\lambda)\beta \in \Lambda$.

With our notation, scalar multiplication is not consistent with Minkowski addition. In particular we have $2\Theta \subseteq \Theta + \Theta$ in general. If Λ is downward closed, then $\bar{\Lambda} = \Lambda$.

The polynomial space P_Λ is defined by a given multi-index Λ:

$$P_\Lambda = \operatorname{span}\{x^\alpha \mid \alpha \in \Lambda\}, \quad |\Lambda| = N,$$

and P_Λ has dimension N in $L^2_\mu(D)$ under the assumption (3). Note that we make no particular assumptions on the structure of Λ. I.e., we do not assume Λ is downward closed, but much of our theory and all our numerical examples use downward closed index sets.

On $\mathbb{N}_0^d$, we will make use of the ℓ^p quasinorm $\|\cdot\|_p$ for $0 \le p \le \infty$, and the associated ball $B_p(r)$ of radius $r \ge 0$ to define index sets. These sets are defined by

$$B_p(r) = \left\{\alpha \in \mathbb{N}_0^d \mid \|\alpha\|_p \le r\right\}.$$

The ℓ^p quasinorms are defined for $p = 0$, $0 < p < \infty$, and $p = \infty$ by, respectively,

$$\|\alpha\|_0 = \sum_{j=1}^d \mathbb{1}_{\alpha_j \neq 0}, \qquad \|\alpha\|_p^p = \sum_{j=1}^d \alpha_j^p, \qquad \|\alpha\|_\infty = \max_{1\le j\le d} \alpha_j.$$

The index sets $B_p(r)$ are all downward closed, and are convex if $p \ge 1$. The set $B_0(r)$ equals $\mathbb{N}_0^d$ when $r \ge d$.

2.2. Minimal quadrature

With M fixed, the theoretical and computational tractability of computing a solution to (2) depends on Λ, D, and μ. In particular, it is unreasonable to expect that Λ can be arbitrarily large; if this were true, then μ can be approximated to arbitrary accuracy by a sum of M Dirac delta distributions, which would allow us to violate the lower inequality in (3). There is a strong heuristic that motivates the possible size of Λ: The set $\{x_1, \ldots, x_M\}$ represents Md degrees of freedom, and varying w_j $(j = 1, \ldots, M)$ represents an additional M degrees of freedom. For an N-dimensional space P_Λ, (2) can be ensured with N constraints. Thus we expect for general (μ, D) that Λ (and thus N) must be small enough to satisfy

$$|\Lambda| = N \le (d+1)M. \tag{4}$$

We will show that this heuristic does not always produce a faithful bound on sizes of quadrature rules; we provide instead a strict lower bound on the number of points M in a quadrature rule for a given Λ. To proceed we require the notion of 'half-sets'.

Definition 2.2. Let $\Lambda \in \mathbb{N}_0^d$ be a finite, nontrivial, downward-closed set. A multi-index set Θ is

1. a half-set for Λ if $\Theta + \Theta \subseteq \Lambda$
2. a maximal half-set for Λ if it is a half-set of maximal size. I.e, if $|\Theta| = L$, with

$$L = L(\Lambda) = \max\{|\Theta| \mid \Theta \text{ a multi-index set satisfying } \Theta + \Theta \subseteq \Lambda\} \tag{5}$$

We call L the *maximal half-set size* of Λ.

Recall that $2\Theta \subseteq \Theta + \Theta$ so that the terminology "half" should not be conflated with the operation of halving each index in an index set. If Λ is not downward closed, it may not have any half sets. However, all nontrivial downward-closed sets have at least one nontrivial half-set (the zero set $\{0\}$ is one such half set). Thus maximal half sets always exist in this case, but they are not necessarily unique. An example in $d = 2$, depicted in Fig. 1, illustrates this non-uniqueness. Specifically let

$$\Lambda = B_0(1) \cap B_1(3) = \{(0,0),\ (0,1),\ (0,2),\ (0,3),\ (1,0),\ (2,0),\ (3,0)\},$$
$$\Theta_1 = \{(0,0),\ (1,0)\}, \quad \Theta_2 = \{(0,0),\ (0,1)\}.$$

For this example $L(\Lambda) = 2$ and both Θ_1 and Θ_2 are maximal half sets for Λ.

If Λ is both downward-closed and convex, then its maximal half-set is unique and easily computed.

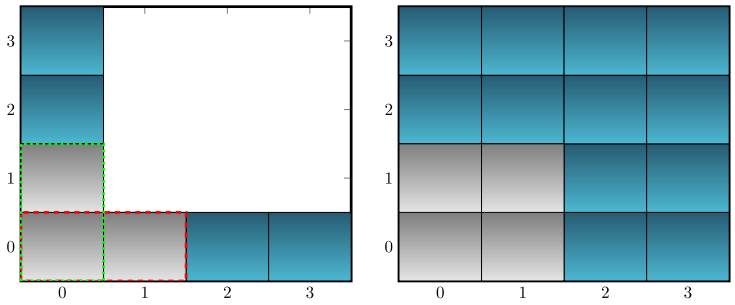

Fig. 1. (Left) Index set $\Lambda = B_0(1) \cap B_1(3)$ (blue and gray) and half sets $\Theta_j = \{0, e_j\}$ (gray), for $j = 1$ (red dashed line) and $j = 2$ (green dotted line), where e_j is the cardinal unit vector in the jth direction in $\mathbb{N}_0^2$. (Right) Index set $\Lambda = B_\infty(2)$ (blue and gray) and unique half set $\Theta = \{(0,0),(0,1),(1,0),(1,1)\}$ (gray). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Theorem 2.1. *Let Λ be convex and downward-closed. Then its maximal half-set Θ is unique, given by $\Theta = \left\lfloor \frac{1}{2}\Lambda \right\rfloor$.*

Proof. Let Θ be any half-set for Λ. Then for any $\theta \in \Theta$, we have $2\theta \in \Lambda$, so that $\theta \in \left\lfloor \frac{1}{2}\Lambda \right\rfloor$. Thus $\Theta \subseteq \left\lfloor \frac{1}{2}\Lambda \right\rfloor$, showing that any half-set must be contained in $\left\lfloor \frac{1}{2}\Lambda \right\rfloor$.

Now let $\theta_1, \theta_2 \in \left\lfloor \frac{1}{2}\Lambda \right\rfloor$. Then $2\theta_1, 2\theta_2 \in \bar{\Lambda} = \Lambda$. Thus,

$$\theta_1 + \theta_2 = \frac{1}{2}(2\theta_1) + \frac{1}{2}(2\theta_2) \in \Lambda,$$

where the set inclusion holds since $\theta_1 + \theta_2 \in \mathbb{N}_0^d$, and since Λ is convex. Thus, $\left\lfloor \frac{1}{2}\Lambda \right\rfloor$ is itself a half-set; by the previous observation that it also dominates any half-set, then it must be the unique largest (maximal) half-set. □

We can now state one of the main results of this section: The number $L(\Lambda)$ in (5) is a lower bound on the size of any quadrature rule satisfying (2).

Theorem 2.2. *Let Λ be a finite downward-closed set, and suppose that an M-point quadrature rule $\{x_j, w_j\}_{n=1}^N$ exists satisfying (2). Then $M \geq L(\Lambda)$, with L the maximal half-set size defined in (5).*

Proof. Let Θ be any set satisfying $\Theta + \Theta \subseteq \Lambda$ and $|\Theta| = L$, with L defined in (5). We choose any size-L μ-orthonormal basis for P_Θ:

$$P_\Theta = \operatorname{span}\{q_j\}_{j=1}^L, \quad \int q_j(x) q_k(x) \mathrm{d}\mu(x) = \delta_{k,j}$$

Note that q_j and q_k have monomial expansions

$$q_j(x) = \sum_{\alpha \in \Theta} c_\alpha x^\alpha, \; q_k(x) = \sum_{\alpha \in \Theta} d_\alpha x^\alpha,$$

for fixed $j, k = 1, \ldots, N$ and some constants c_α and d_α. This implies

$$q_j(x) q_k(x) = \sum_{\alpha,\beta \in \Theta} c_\alpha d_\beta x^{\alpha+\beta} = \sum_{\alpha \in (\Theta+\Theta)} f_\alpha x^\alpha.$$

Since Θ is a half set for Λ, then $q_j q_k \in P_\Lambda$ and is therefore exactly integrated by the quadrature rule (2).

Then consider the $L \times L$ matrix $\boldsymbol{G}$ defined as

$$\boldsymbol{G} = \boldsymbol{V}^T \boldsymbol{W} \boldsymbol{V}, \qquad (G)_{j,k} = \sum_{m=1}^M w_m q_j(x_m) q_k(x_m)$$

$$(V)_{j,k} = q_k(x_j), \qquad (W)_{j,k} = w_j \delta_{j,k}. \tag{6}$$

The matrices $\boldsymbol{V}$ and $\boldsymbol{W}$ are $M \times L$ and $M \times M$, respectively. Since the quadrature rule exactly integrates $q_j q_k$, then $\boldsymbol{G} = \boldsymbol{I}$, the $L \times L$ identity matrix. Thus, the matrix product $\boldsymbol{V}^T \boldsymbol{W} \boldsymbol{V}$ has rank L. If $M < L$ then, e.g., $\operatorname{rank}(\boldsymbol{W}) < L$ and so it is not possible that $\operatorname{rank}\left(\boldsymbol{V}^T \boldsymbol{W} \boldsymbol{V}\right) = L$. □

This result shows that if $M < L(\Lambda)$, then an M-point quadrature rule satisfying (2) cannot exist. This is nontrivial information. As an example, let $d = 2$, and consider

$$\Lambda = B_\infty(2) = \{(0,0),(0,1),(0,2),(1,0),(1,1),(1,2),(2,0),(2,1),(2,2)\},$$

corresponding to the tensor-product space of degree 2. Since $|\Lambda| = 9$, the heuristic (4) suggests that it is possible to find a rule with only 3 points. However, let $\Theta = \{(0,0),(0,1),(1,0),(1,1)\}$, such that $\Theta + \Theta = \Lambda$ and $|\Theta| = 4$. Thus, no 3-point rule that is accurate on P_Λ can exist. The index set Λ and the half set Θ for this example are shown in Fig. 1, right.

Another observation from Theorem 2.2 is that we may justifiably call an M-point quadrature rule minimal if $M = L$, in the sense that one cannot achieve the same accuracy with a smaller number of nodes. A quadrature rule is minimal if no other rule with a smaller number of nodes can have identical (or greater) accuracy.

Definition 2.3. Let Λ be a finite, downward-closed multi-index set. An M-point quadrature rule is *minimal* for Λ if (i) it satisfies (2), and (ii) any other quadrature rule with $\widetilde{M}$ points satisfying (2) has size obeying $\widetilde{M} \geq M$.

Since Λ is finite, then a minimal quadrature rule always exists. (E.g., the interpolatory quadrature rule over any set of $|\Lambda|$ abscissae for which the interpolation problem from P_Λ is unisolvent has finite size N.) Minimal quadrature rules by the definition above are not necessarily unique, and among several minimal quadrature rules some may have greater accuracy than others. (See Section 3.3.) Our definition does not guarantee existence of minimal quadrature rules, but Theorem 2.2 gives us an easy method to check if a quadrature rule is minimal.

Corollary 2.1. *Let Λ be a downward-closed multi-index set. An M-point quadrature rule is minimal for Λ if it satisfies (2) and $M = L(\Lambda)$, with L defined in (5).*

The condition $M = L(\Lambda)$ is a stronger condition than minimality, and Section 3.4 gives a concrete example of (Λ, D, μ) where a minimal quadrature rule must have $M > L(\Lambda)$. Thus, rules achieving the lower bound $M = L(\Lambda)$ are minimal, but minimal rules need not satisfy this lower bound.

The weights for minimal quadrature rules with $M = L(\Lambda)$ have a precise behavior. Under the assumption (3), we can find an orthonormal basis q_j for P_Λ:

$$\langle q_j, q_k \rangle_\mu = \delta_{j,k}, \quad P_\Lambda = \operatorname{span}\{q_j\}_{j=1}^N$$

Given this orthonormal basis, define

$$\lambda_\Lambda(x) = \frac{1}{\sum_{j=1}^N q_j^2(x)}.$$

The quantity λ_Λ depends on D, μ, and P_Λ, but not on the particular basis q_j we have chosen: Through a unitary transform we may map $\{q_j\}_{j=1}^N$ into any other orthonormal basis for P_Λ, but this transformation leaves the quantity above unchanged. The weights for a lower-bound-achieving quadrature rule on Λ are evaluations of λ_Θ, where Θ is a maximal half-set for Λ.

Theorem 2.3. *Let Λ be a finite downward-closed set, and let an M-point quadrature rule $\{x_m, w_m\}_{m=1}^M$ be minimal for Λ with $M = L(\Lambda)$. Then*

$$w_j = \lambda_\Theta(x_j), \quad j = 1, \ldots, M, \tag{7}$$

where Θ is a(ny) maximal half set for Λ.

Proof. Let $\ell_j(x)$, $j = 1, \ldots, M$, be the cardinal Lagrange interpolants from the space P_Θ on the nodes $\{x_j\}_{j=1}^M$.[1] These cardinal interpolants satisfy $\ell_j(x_k) = \delta_{j,k}$, span P_Θ, and are pairwise orthogonal in L^2_μ by (2) and the fact that $\Theta + \Theta \subseteq \Lambda$. The weights w_j must be positive since

$$w_j = \sum_{m=1}^M w_m \ell_j^2(x_m) = \int \ell_j^2(x) \mathrm{d}\mu(x) > 0, \tag{8}$$

[1] The interpolation problem of x_j for any basis of P_Θ must be invertible since the Vandermonde-like matrix $\boldsymbol{V}$ in the proof of Theorem 2.2 must have full rank, be square since $M = L(\Lambda)$, and thus be invertible. Thus, the ℓ_j are well-defined.

where the second equality uses the fact that the quadrature rule is exact for $\ell_j^2 \in P_\Lambda$. The above relation also shows that $q_k(x) := \ell_k(x)/\sqrt{w_k}$, $k = 1, \ldots, M$, is an orthonormal basis for P_Θ. Then the $M \times M$ matrices $\boldsymbol{V}$ and $\boldsymbol{W}$ with entries defined by (6) satisfy $\boldsymbol{V}^T \boldsymbol{W} \boldsymbol{V} = \boldsymbol{I}$, due to the exactness of the quadrature rule for $q_j q_k \in P_\Lambda$. Thus, $\sqrt{\boldsymbol{W}}\boldsymbol{V}$ is an orthogonal matrix, so that

$$\begin{aligned}\sqrt{\boldsymbol{W}}\boldsymbol{V}\boldsymbol{V}^T\sqrt{\boldsymbol{W}^T} &= \boldsymbol{I}\\ \boldsymbol{V}\boldsymbol{V}^T &= \boldsymbol{W}^{-1}\end{aligned}$$

Taking the diagonal components of the left- and right-hand side shows that $w_j = \left(\sum_{k=1}^N (V)_{j,k}^2\right)^{-1}$. □

2.3. Relation to lower bounds for total degree spaces

Our condition $M \geq L(\Lambda)$ from Theorem 2.2 is a lower bound for the possible size of a quadrature rule with prescribed accuracy, and is a notable generalization of existing lower bounds. For example, the following well-known result can be derived as a special case of Theorems 2.2 and 2.1.

Corollary 2.2. *Let $\Lambda = B_1(k)$ be the total degree index set of degree k. Then any M-point quadrature rule satisfying (2) must satisfy*

$$M \geq |B_1(\lfloor k/2 \rfloor)|\,.$$

The above is a result that has appeared in the past in various incarnations, and with various restrictions on D, μ, k, and d [24–27]. Our result matches all these statements: In order to integrate polynomials up to degree k with a quadrature rule, at least $\dim P_\Theta$ points are required, with $\Theta = B_1(\lfloor k/2 \rfloor)$. Other than our assumptions in Section 2.1, our result requires no further properties about D or μ, and does not require the quadrature rule to be an interior rule (i.e., to have all nodes inside D).

However our results, Theorems 2.2 and 2.3, and Corollary 2.1, apply to general downward-closed index sets Λ that can be significantly more general than total degree spaces. For a general discussion about existing lower bounds on quadrature rule sizes we refer to [19], and to [28] for a compilation of many existing rules.

3. Examples: Quadrature rule minimality and the lower bound

To provide further insight into our lower bound characterization, Theorem 2.2, we investigate the consequences of the above theoretical results through some examples. This section consists of four examples: (i) a univariate example of a minimal rule achieving the lower bound, (ii) a multivariate minimal rule achieving the lower bound, (iii) many multivariate minimal rules achieving the lower bound, (iv) a multivariate situation where a minimal rule cannot achieve the lower bound.

3.1. Univariate rules

The example in this section discusses the facts (i) that minimal quadrature rules in one dimension are Gauss quadrature rules, and (ii) a great many minimal rules in one dimension achieve the lower bound, and some are capable of accurately integrating more polynomials than the others.

Let $d = 1$ with $\Lambda = \{0, \ldots, N-1\}$. The maximal half-set for Λ is $\Theta = \left\{0, \ldots, \left\lfloor \frac{N-1}{2} \right\rfloor\right\}$, and thus

$$L(\Lambda) = \left\lfloor \frac{N-1}{2} \right\rfloor + 1$$

We consider two cases, that of even N, and of odd N.

Suppose N is even. Then a quadrature rule achieving our lower bound has $M = |\Theta| = \frac{N}{2}$ abscissae, and exactly integrates $M = 2N$ (linearly independent) polynomials. I.e., the N-point rule exactly integrates polynomials up to degree $2M - 1$. It is well-known that this quadrature rule is unique: it is the μ-Gauss–Christoffel quadrature rule [15, chapter 3].

With $\left\{q_j\right\}_{j=0}^\infty$ a family of $L_\mu^2(D)$-orthonormal polynomials, with $\deg q_j = j$, then the abscissae $x_1, \ldots, x_M$ are the zeros of q_M, and the weights are given by (7). This quadrature rule can be efficiently computed from eigenvalues

and eigenvectors of the symmetric tridiagonal Jacobi matrix associated with μ [29]. Thus, this quadrature rule can be computed without resorting to optimization routines.

Now suppose N is odd. Then a quadrature rule achieving our lower bound has $M = L(\Lambda) = \frac{N+1}{2}$ abscissae, and exactly integrates $2M-1$ polynomials. I.e., the M-point rule exactly integrates polynomials up to degree $2M-2$. Clearly a Gauss–Christoffel rule is a minimal rule achieving our lower bound in this case. However, by our definition there are an (uncountably) *infinite* number of such lower-bound-achieving rules in this case [15]. In particular, for arbitrary $c \in \mathbb{R}$, the zero set of the polynomial

$$q_M(x) - cq_{M-1}(x), \tag{9}$$

corresponds to the abscissae of a quadrature rule achieving the lower bound, with the weights given by (7). Like the Gauss–Christoffel case, computation of these quadrature rules can be accomplished via eigendecompositions of symmetric tridiagonal matrices [30].

In classical scientific computing scenarios, sets of M-point rules with polynomial accuracy of degree $2M-2$ have been called Gauss–Radau rules, and are traditionally associated to situations when $\mathrm{supp}\mu$ is an interval and one of the quadrature abscissae is collocated at one of the endpoints of this interval [31].

3.2. Multivariate quadrature achieving the lower bound

This section furnishes an example where a multivariate quadrature rule achieving our lower bound can be explicitly constructed.

Consider a tensorial domain and probability measure, i.e., suppose

$$\mu = \times_{j=1}^{d} \mu_j,\ D = \otimes_{j=1}^{d} \mathrm{supp}\mu_j$$

where μ_j are univariate probability measures. We let $\Lambda = B_0(1) \cap B_1(n)$ for any $n \geq 2$. With e_j the cardinal j'th unit vector in $\mathbb{N}_0^d$, then Λ is the set of indices of the form qe_j for $0 \leq q \leq n$; the size of Λ is $N = dn + 1$. (See Fig. 1 for a visualization of these sets with $d = 2$, $n = 3$.) Note that this Λ would arise in situations where one seeks to approximate an unknown multivariate function as a sum of univariate functions; this is rarely a reasonable assumption in practice. However, in this case we can explicitly construct optimal quadrature rules.

The heuristic (4) suggests that we can construct a rule satisfying (2) if we use

$$M \geq n\left(\frac{d}{d+1}\right) + \frac{1}{d+1}$$

nodes, which is approximately n nodes. However, we can achieve this with fewer nodes, only $M = \lfloor n/2 \rfloor + 1$, independent of d. Here the heuristic (4) is too pessimistic when $d \geq 2$. Associated to each μ_j, we need the corresponding system of orthonormal polynomials $q_{n,j}$, $n \geq 0$, and the univariate λ function. For each $j = 1, \ldots, d$, let $q_{n,j}(x)$, $n = 0, 1, \ldots$, denote a $L^2_{\mu_j}$-orthonormal polynomial family with $\deg q_{n,j} = n$. Define

$$\lambda_{n,j}(\cdot) = \frac{1}{\sum_{i=0}^{n} q_{i,j}^2(\cdot)}.$$

Note that $\lambda_{0,j} = 1/q_{0,j}^2 \equiv 1$ for all j since each μ_j is a probability measure.

Consider the index sets $\Theta_j = \left\{0, e_j, 2e_j, \ldots, \lfloor n/2 \rfloor e_j\right\}$, for $j = 1, \ldots, d$, where 0 is the origin in $\mathbb{N}_0^d$. Each Θ_j is a maximal half-set for Λ, and L in (5) is given by $L = \lfloor n/2 \rfloor + 1$. The index set Λ and Θ_j, $j = 1, 2$ for $n = 3$ are shown in Fig. 1, left.

To construct a quadrature rule achieving the lower bound, $M = L$ nodes, we note that the weights w_j, $j = 1, \ldots, N$ must be given by (7), which holds for *any* maximal half index set Θ. I.e., it must simultaneously hold for *all* Θ_j. Thus,

$$w_m = \lambda_{\Theta_j}(x_m) = \left[\sum_{i=0}^{\lfloor n/2 \rfloor} q_{i,j}^2\left(x_m^{(j)}\right) \prod_{\substack{s=1,\ldots,d \\ s \neq j}} q_{0,s}^2\right]^{-1} = \lambda_{\lfloor n/2 \rfloor, j}\left(x_m^{(j)}\right), \tag{10}$$

for $j = 1, \ldots, d$. This implies in particular that the coordinates $x_m^{(j)}$ for node m must satisfy

$$\lambda_{\lfloor n/2 \rfloor,1}\left(x_m^{(1)}\right) = \lambda_{\lfloor n/2 \rfloor,2}\left(x_m^{(2)}\right) = \cdots = \lambda_{\lfloor n/2 \rfloor,d}\left(x_m^{(d)}\right).$$

We can satisfy this condition in certain cases. Suppose $\mu_1 = \mu_2 = \cdots = \mu_d$; then $\lambda_{n,1} = \lambda_{n,2} = \cdots = \lambda_{n,d}$ and so we can satisfy (10) by setting $x_m^{(1)} = x_m^{(2)} = \cdots = x_m^{(d)}$ for all $m = 1, \ldots, M$. Thus nodes for a lower-bound-achieving quadrature rule nodes could lie in $\mathbb{R}^d$ along the graph of the line defined by

$$x^{(1)} = x^{(2)} = \cdots = x^{(d)}.$$

In order to satisfy the integration condition (2) we need to distribute the nodes in an appropriate way. Having effectively reduced the problem to one dimension, this is easily done: we choose a Gauss-type quadrature rule as described in the previous section. Let the jth coordinate of the quadrature rule be the M-point Gauss quadrature nodes for μ_j, i.e.,

$$\left\{x_1^{(j)}, x_2^{(j)}, \ldots, x_M^{(j)}\right\} = q_{M,j}^{-1}(0).$$

This then uniquely defines $x_1, \ldots, x_M$, and w_m is likewise uniquely defined since we have satisfied (10).

Thus a "diagonal" Gauss quadrature rule, $M = \lfloor n/2 \rfloor + 1$, with equal coordinate values for each abscissa, is a minimal rule in this case that achieves the lower bound $M = L$.

3.3. Multivariate quadrature: non-uniqueness of rules achieving the lower bound

The previous example with an additional assumption allows to construct 2^{d-1} distinct minimal quadrature rules achieving the lower bound. Again take $\Lambda = B_0(1) \cap B_1(n)$ for $n \geq 2$, and let $\mu = \times_{j=1}^d \mu_j$ with identical univariate measures $\mu_j = \mu_k$.

To this add the assumption that μ_j is a symmetric measure; i.e., for any set μ_j-measurable $A \subset \mathbb{R}$, then $\mu_j(A) = \mu_j(-A)$. In this case the univariate orthogonal polynomials $q_{k,j}$ are even (odd) functions if k is even (odd). Thus, the set $q_{k,j}^{-1}(0)$ is symmetric around 0, and $\lambda_{M,j}$ is always an even function. With x_m the lower-bound-achieving set of nodes defined in the previous section, let

$$y_m^{(j)} = \sigma_j x_m^{(j)}, \sigma_j \in \{-1, +1\},$$

for a fixed but arbitrary sign train $\sigma_1, \ldots, \sigma_d$. Using the above properties, one can show that the nodes $\{y_1, \ldots, y_M\}$ and the weights $w_1, \ldots, w_M$ define a quadrature rule satisfying (2), and of course have the same number of nodes as the minimal rule from the previous section.

By varying the σ_j, we can create 2^{d-1} unique distributions of nodes, thus showing that at least this many minimal rules exist that achieve the lower bound.

3.4. Multivariate quadrature: minimal rule sizes can exceed the lower bound

We again use the setup of Section 3.2, but this time to illustrate that it is possible for minimal quadrature rule sizes to exceed our lower bound. We consider $d = 2$, and take $\Lambda = B_0(1) \cap B_1(3)$, and let $\mu = \mu_1 \times \mu_2$ for two univariate probability measures μ_1 and μ_2. We will show that a quadrature rule with $M = L(\Lambda)$ cannot exist. In contrast to previous sections, we let the μ_1 and μ_2 measures be different:

$$\mathrm{d}\mu_1(t) = \frac{1}{2}\mathrm{d}t, \qquad \text{supp } \mu_1 = [-1, 1]$$
$$\mathrm{d}\mu_2(t) = \frac{3}{4}(1 - t^2)\mathrm{d}t, \qquad \text{supp } \mu_2 = [-1, 1]$$

With our choice of Λ, we have $L(\Lambda) = 2$ with two maximal half-sets:

$$\Theta_1 = \{(0, 0), (1, 0)\}, \qquad \Theta_2 = \{(0, 0), (0, 1)\}$$

In this simple case the explicit μ_1- and μ_2-orthonormal polynomial families have constant and linear polynomials of the form

$$q_{0,1}(t) = 1, \qquad q_{1,1}(t) = \sqrt{3}t,$$
$$q_{0,2}(t) = 1, \qquad q_{1,2}(t) = \frac{\sqrt{5}}{2}t$$

so that associated to Θ_1 and Θ_2, respectively, we have the functions

$$\lambda_{1,1}(t) = \frac{1}{1+3t^2}, \qquad \lambda_{1,2}(t) = \frac{4}{4+5t^2}.$$

Since by (10) we require $w_m = \lambda_{1,1}\left(x_m^{(1)}\right) = \lambda_{1,2}\left(x_m^{(2)}\right)$, this implies that

$$3\left(x_m^{(1)}\right)^2 = \frac{5}{4}\left(x_m^{(2)}\right)^2, \quad m = 1, 2 \tag{11a}$$

However, the condition (2) also implies for our choice of Λ that

$$\int_{-1}^{1} p(t)\frac{1}{2}\mathrm{d}t = \sum_{m=1}^{2} w_m p\left(x_m^{(1)}\right), \quad p \in \operatorname{span}\left\{1, t, t^2, t^3\right\},$$

$$\int_{-1}^{1} p(t)\frac{3}{4}(1-t^2)\mathrm{d}t = \sum_{m=1}^{2} w_m p\left(x_m^{(2)}\right), \quad p \in \operatorname{span}\left\{1, t, t^2, t^3\right\}.$$

The conditions above imply that $\left\{x_m^{(1)}\right\}_{m=1}^2$ and $\left\{x_m^{(2)}\right\}_{m=1}^2$ be nodes for the 2-point μ_1- and μ_2-Gauss quadrature rules, respectively, which are both unique. Thus, $\left\{x_m^{(1)}\right\}_{m=1}^2 = \left\{\pm\sqrt{3}/3\right\}$, and $\left\{x_m^{(2)}\right\}_{m=1}^2 = \left\{\pm 1/\sqrt{5}\right\}$. Thus, we have the equality

$$3\left(x_m^{(1)}\right)^2 = 5\left(x_m^{(2)}\right)^2, \quad m = 1, 2 \tag{11b}$$

We arrive at a contradiction: simultaneous satisfaction of (11a) and (11b) implies all coordinates of the abscissae are 0, which cannot satisfy (2). Thus, a quadrature rule satisfying (2) cannot have $M = L(\Lambda)$ abscissae, and so any minimal rule has size M exceeding the lower bound, i.e., $M > L$.

4. Numerically generating multivariate quadrature rules

In this section we describe our proposed algorithm for generating multivariate quadrature rules; the algorithm generates rules having significantly fewer points than the number of moments being matched. We will refer to such rules as *reduced quadrature* rules. We will compare the number of nodes our rules can generate with the lower bound $L(\Lambda)$ defined in Section 2, along with the heuristic bound (4).

Our method is an adaptation of the method presented in [9]. The authors there showed that one can recover a univariate Gauss quadrature rule as the solution to an infinite-dimensional linear program (LP) over nonnegative measures. Let a finite index set Λ be given with $|\Lambda| = N$, and suppose that $\left\{p_j\right\}_{j=1}^N$ is any basis for P_Λ. In addition, let r be a polynomial on $\mathbb{R}^d$ such that $r \notin P_\Lambda$; we seek a positive Borel measure ν on $\mathbb{R}^d$ solving

$$\text{minimize} \int r(\mathbf{x})\mathrm{d}\nu(\mathbf{x}) \tag{12}$$

$$\text{subject to} \int p_j(\mathbf{x})\mathrm{d}\nu(\mathbf{x}) = \int p_j(\mathbf{x})\mathrm{d}\mu(\mathbf{x}),\ j = 1, \ldots, N_\Lambda \tag{13}$$

The restriction that ν is a positive measure will enter as a constraint into the optimization problem. A nontrivial result is that a positive measure solution to this problem exists with $|\operatorname{supp}\nu| = M \leq N$. Such a solution immediately yields a positive quadrature rule with nodes $\{\mathbf{x}_1, \ldots, \mathbf{x}_M\} = \operatorname{supp}\nu$ and weights given by $w_j = \nu(\mathbf{x}_j)$. Unfortunately, the above optimization problem is NP-hard, and so the authors in [9] propose a two step procedure for computing an approximate solution. The first step solves a finite-dimensional linear problem similar to (12) to find a K-point positive quadrature rule with $K \leq N$. In the second step, the $K \leq N$ points are clustered into M groups, where M is automatically chosen by the algorithm. The clusters are then used to form a set of M averaged points and weights, which form an approximate quadrature rule. This approximate quadrature rule is refined using a local gradient-based method to optimize a moment-matching objective. The precise algorithm is outlined in more detail in Appendix A.

In this paper we also adopt a similar two step procedure to compute reduced quadrature rules. We outline these two steps in detail in the following section. Pseudo code for generating reduced quadrature rules is presented in Algorithm 1 contained in Appendix B.

4.1. Generating an initial condition

Let an index set Λ be given with $|\Lambda| = N < \infty$, and suppose that $\left\{p_j\right\}_{j=1}^{n}$ is a basis for P_Λ. We seek to find a discrete measure $\nu = \sum_k v_k \delta_{x_k}$ by choosing a large candidates mesh, $X_S = \{x_k\}_{k=1}^{S}$, with $S \gg N$ points on D and solving the ℓ_1 minimization problem

$$\begin{aligned} &\text{minimize} \sum_{k=1}^{S} |v_k| \\ &\text{subject to} \sum_{k=1}^{S} v_k p_j(x_k) = \int p_j(x) \mathrm{d}\mu(x),\ j = 1, \ldots, N \\ &\qquad \text{and } v_k \geq 0, k = 1, \ldots, S \end{aligned} \tag{14}$$

The optimization is over the S scalars v_k. With $\boldsymbol{v} \in \mathbb{R}^S$ a vector containing the v_k, the non-zero coefficients then define a quadrature rule with $K = \|\boldsymbol{v}\|_0$ points, $K \leq N$. The points corresponding to the non-zero coefficients v_k are the quadrature points and the coefficients themselves are the weights.

This ℓ_1-minimization problem as well as the residual based linear program used by [9] become increasingly difficult to solve as the size of Λ and dimension d of the space increase. The ability to find a solution is highly sensitive to the candidate mesh. Through extensive experimentation we found that by solving (14) approximately via

$$\begin{aligned} &\text{minimize} \sum_{k=1}^{S} |\alpha_k| \\ &\text{subject to } \left|\sum_{k=1}^{S} \alpha_k p_j(x_k) - \int p_j(x) \mathrm{d}\mu(x)\right| < \epsilon,\ j = 1, \ldots, n \\ &\qquad \text{and } \alpha_k \geq 0, k = 1, \ldots, M \end{aligned} \tag{15}$$

for some $\epsilon > 0$, we were able to find solutions that, although possibly inaccurate with respect to the moment matching criterion, could be used as an initial condition for the local optimization to find an accurate reduced quadrature rule. This is discussed further in Section 5.1. We were not able to find such approximate initial conditions using the linear program used in [9]; we show results supporting this in Section 5.

We solved (15) using least angle regression with a LASSO modification [32] whilst enforcing positivity of the coefficients. This algorithm iteratively adds and removes positive weights α_k until $\epsilon = 0$, or until no new weights can be added without violating the positivity constraints. This allows one to drive ϵ to small values without requiring an *a priori* estimate of ϵ.

In our examples, the candidate mesh X_S is selected by generating uniformly random samples over the integration domain D, regardless of the integration measure. Better sampling strategies for generating the candidate mesh undoubtedly exist, but these strategies will be D- and Λ-dependent, and are not the central focus of this paper. Our limited experimentation suggested that there was only marginal benefit from exploring this issue.

4.2. Finding a reduced quadrature rule

Once an initial condition has been found by solving (15) we then use a simple greedy clustering algorithm to generate an initial condition for a local gradient-based moment-matching optimization.

4.2.1. Clustering

The greedy clustering algorithm finds the point with the smallest weight and combines it with its nearest neighbor. These two points are then replaced by a point whose coordinates are a convex combination of the two clustered points, where the weights correspond to the v_k weights that are output from the LP algorithm. The newly clustered point set has one less point than before. This process is repeated until a desired number of points M is reached. At the termination of the clustering algorithm a set of points $\widehat{x}_m$ and weights $\widehat{w}_m$, defining an approximate quadrature

rule are returned. The algorithm pseudo-code describing this greedy clustering algorithm is given in Algorithm 2 in Appendix B.

The ability to find an accurate reduced quadrature rule is dependent on the choice of the number of specified clusters M. As stated in Section 2, there is a strong heuristic that couples the dimension d, the number of matched moments N, and the number of points M. For general μ and given $N = |\Lambda|$ we set the number of clusters to be

$$M = \frac{N}{d+1}. \tag{16}$$

As shown in Section 2 this heuristic will not always produce a quadrature rule that exactly integrates all specified moments. Moreover, the heuristic may overestimate the actual number of points in a quasi-optimal quadrature rule, as shown in Section 3.2.

It is tempting to set M to the lower bound value $L(\Lambda)$, defined in (5). However, a sobering result of our experimentation is that in all our experiments we were never able to numerically find a quadrature rule with fewer points than that specified by (16); therefore, we could not find any rules with M points where $L(\Lambda) \leq M < N/(d+1)$. However, we were able to identify situations, and numerically construct reduced quadrature rules, in which the heuristic underestimated the requisite number of points in a reduced quadrature rule. For example, Corollary 2.2 implies that if one wants to match the moments of a total degree basis of degree k (here with k even) one must use at least $\binom{k/2+d}{d}$ points. This lower bound is typically violated by the heuristic for low-degree k and high-dimension d. E.g. for $d = 10$ and $k = 2$ we have $M = 6$ using the heuristic yet the theoretical lower bound for M from Corollary 2.2 requires $M \geq L(\Lambda) = 11$. In this case our theoretical analysis sets a lower bound for M that is more informative than the heuristic (16).

4.2.2. Local optimization

The approximate quadrature rule $\widehat{x}_k$, $\widehat{w}_k$ generated by the clustering algorithm is used as an initial guess for the following local optimization

$$\text{minimize} \quad \sum_{j=1}^{N} \left(\int p_j(x) \mathrm{d}\mu(x) - \sum_{m=1}^{M} w_m p_j(x_m) \right)^2 \tag{17a}$$

$$\text{subject to } x_m \in D \text{ and } w_m \geq 0, \quad m = 1, \ldots, M \tag{17b}$$

The objective f defined by (17a) is a polynomial and its gradient can easily be computed

$$\frac{df}{dx_k^{(s)}} = -\sum_{i=0}^{n-1} \left[w_k \frac{dp_i(x_k)}{dx_k^{(s)}} \left(q(p_i) - \sum_{j=1}^{M} w_j p_i(x_j) \right) \right] \tag{18a}$$

$$\frac{df}{dw_k} = -\sum_{i=0}^{n-1} \left[p_i(x_k) \left(q(p_i) - \sum_{j=1}^{M} w_j p_i(x_j) \right) \right], \tag{18b}$$

for $s = 1, \ldots, d$ and $m = 1, \ldots, M$, and where $q(p_i) = \int p_i(x) \mathrm{d}\mu(x)$.

We use a gradient-based nonlinear least squares method to solve the local optimization problem. Defining the optimization tolerance $\tau = 10^{-10}$, the procedure exits when either $|f_i - f_{i-1}| < \tau f_i$ or $\|g_s\|_\infty < \tau$, where f_i and f_{i-1} are the values of the objective at steps i and $i-1$ respectively, and g_s is the value of the gradient scaled to respect the constraints (17b).

This local optimization procedure in conjunction with the cluster based initial guess can frequently find a quadrature rule of size M as determined by the degree of freedom heuristic (16). However in some cases a quadrature rule with M points cannot be found for very high accuracy requirements (in all experiments we say that a quadrature rule is found if the iterations yield $|f_i| < 10^{-8}$). In these situations one might be able to find an M point rule using another initial condition. (Recall the initial condition provided by ℓ_1-minimization is found using a randomized candidate mesh.) However we found it more effective to simply increment the size of the desired quadrature rule to $M+1$. This can be done repeatedly until a quadrature rule with the desired accuracy is reached. While one fears that M may be incremented by a large number using this procedure, we found that no more than a total of 10 of increments ($M \to M+10$) were ever needed. This is described in more detail in Section 5.1.

Note that both generating the initial condition using (15) and solving the local optimization problem (17a) involve matching moments. We assume in this paper that moments are available and given; in our examples we compute these moments analytically (or to machine precision with high-order quadrature), unless otherwise specified.

5. Numerical results

In this section we will explore the numerical properties of the multivariate quadrature algorithm we have proposed in Section 4. First we will numerically compare the performance of our algorithm with other popular quadrature strategies for tensor product measures. We will then demonstrate the flexibility our approach for computing quadrature rules for non-tensor-product measures, for which many existing approaches are not directly applicable. Finally we will investigate the utility of our reduced quadrature rules for high-dimensional integration by leveraging selected moment matching and dimension reduction.

Our tests will compare the method in this paper (Section 4), which we call REDUCED QUADRATURE, to other standard quadrature methods. We summarize these methods below.

- MONTE CARLO — The integration rule
$$\int_D f(x)\mathrm{d}\mu(x) \approx \frac{1}{M}\sum_{m=1}^{M} f(X_m),$$
where X_m are independent and identically-distributed samples from the probability measure μ.
- SPARSE GRID — The integration rule for μ the uniform measure over $[-1, 1]^d$ given by a multivariate sparse grid rule generated from a univariate Clenshaw–Curtis quadrature rule [5].
- CUBATURE — Stroud cubature rules of degree 2, 3, and 5 [18]. These rules again integrate on $[-1, 1]^d$ with respect to the uniform measure.
- SOBOL — A quasi-Monte Carlo Sobol sequence [4] for approximating integrals on $[-1, 1]^d$ using the uniform measure.
- REDUCED QUADRATURE — The method in this paper, described in Section 4.
- ℓ_1 QUADRATURE — The "initial guess" for the REDUCED QUADRATURE algorithm, using the solution of the LP algorithm summarized in Section 4.1.

The SPARSE GRID, SOBOL, and CUBATURE methods are directly applicable (i.e., without mapping) only for integrating over $[-1, 1]^d$ with the uniform measure. When μ has a density $w(x)$ with support $D \subseteq [-1, 1]^d$, we will use these methods to evaluate $\int_D f(x)\mathrm{d}\mu(x)$ by integrating $f(x)w(x)$ with respect to the uniform measure on $[-1, 1]^d$, where we assign $w = 0$ on $[-1, 1]^d \setminus D$.

5.1. Tensor product measures

Our reduced quadrature approach can generate quadrature rules for non-tensor-product measures but in this section we investigate the performance of our algorithm in the more standard setting of tensor-product measures.

5.1.1. Computational complexity

We begin by discussing the computational cost of computing our quadrature rules. Specifically we compute quadrature rules for the uniform probability measure on $D = [-1, 1]^d$ in up to 10 dimensions for all total-degree spaces with degree at most 20 or with subspace dimension at most 3003. In all cases we were able to generate an efficient quadrature rule using our approach for which the number of quadrature points was equal to or only slightly larger (< 10 points) than the number of points suggested by the heuristic (16). The number of points in each computed quadrature rule, the number of moments matched $|\Lambda|$, the lower bound $L(\Lambda)$ on quadrature rule sizes, the number of points $\lceil |\Lambda|/(d+1) \rceil$ suggested by the heuristic (16), and the number of iterations taken by non-linear least squares algorithm, is presented in Table 1.

The final two columns of Table 1, the number of points in the reduced quadrature rules and the number of iterations used by the local optimization are given as ranges because the final result is sensitive to the random candidate sets used to generate the initial condition. The ranges presented in the table represent the minimum and maximum number of points and iterations used to generate the largest quadrature rules for each dimension for 10 different initial candidate

Table 1
Results for computing REDUCED QUADRATURE rules for total-degree spaces for the uniform measure on $[-1, 1]^d$. Tabulated are the number of moments matched ($|\Lambda|$), the theoretical lower bound on the quadrature rule size from (5), the number of the quadrature points given by the counting heuristic (16), the number of reduced quadrature points found, and the number of iterations required in the optimization. The reduced quadrature algorithm output is random since the initial candidate grid is a random set. Thus, the final two columns give a range of results over 10 runs of the algorithm.

Dimension	Degree	$\lvert\Lambda\rvert$	$L(\Lambda)$	$\lceil\lvert\Lambda\rvert/(d+1)\rceil$	No. Points	No. Iterations
2	20	231	66	77	77–79	262–854
3	20	1771	286	443	445–447	736–1459
4	13	2380	210	476	479–480	782–2033
5	10	3003	252	501	506–508	3181–4148
10	5	3003	66	273	273–274	511–2229

meshes. We observe only very minor variation in the number of points, but more sensitivity in the number of iterations is observed.

The number of iterations needed by the local optimization to compute the quadrature rules was always a reasonable number: at most a small multiple of the number of moments being matched. However, the largest rules did take several hours to compute on a single core machine due to the cost of running the optimizer (we used scipy.optimize.least_squares) and forming the Jacobian matrix of the objective.

Profiling the simulation revealed that long run times were due to the cost of evaluating the Jacobian (18) of the objective function, and the singular value decomposition repeatedly called by the optimizer. This run-time could probably be reduced by using a Krylov-based nonlinear least squares algorithm, and by computing the Jacobian in parallel. We expect that such improvements would allow one to construct such rules in higher dimensions in a reasonable amount of time; however, even our unoptimized code was able to compute such rules on a single core in a moderate number of dimensions.

5.1.2. Accuracy

To compare the performance of our reduced quadrature rules to existing algorithms consider the corner-peak function often used to test quadrature methods [33],

$$f_{\mathrm{CP}}(\mathbf{x}) = \left(1 + \sum_{i=1}^{d} c_i\, x_i\right)^{-(d+1)}, \quad \mathbf{x} \in D = [0,1]^d \tag{19}$$

The coefficients c_i can be used to control the variability over D and the effective dimensionality of this function. We generate each c_i randomly from the interval $[0, 1]$ and subsequently normalize them so that $\sum_{i=1}^{d} c_i = 1$. For X a uniform random variable on $D = [0, 1]^d$, the mean and variance of $f(X)$ can be computed analytically; these values correspond to computing integrals over $D = [0, 1]^d$ with μ the uniform measure. This positive function raises quickly to its peak value of 1 at $x = 0$ from its smallest value of 2^{-d-1} at the opposite corner of the domain, making this function difficult to integrate in high dimensions.

Fig. 2 plots the convergence of the error in the mean of the corner-peak function using the reduced quadrature rules generated for μ the uniform probability measure on $[0, 1]^d$ with $d = 2, 3, 4, 5, 10$. Because generating the initial condition requires randomly sampling over the domain of integration for a given degree, we generate 10 quadrature rules and plot both the median error and the minimum and maximum errors. There is sensitivity to the random candidate set used to generate the initial condition. However, for each of the 10 repetitions a quadrature rule was always found.

For a given number of samples the error in the reduced quadrature rule estimate of the mean is significantly lower than the error of a Clenshaw–Curtis-based sparse grid.[2] It is also more accurate than Sobol sequences up to 10 dimensions. Furthermore the reduced quadrature rule is competitive with the Stroud degree 2,3 and 5 cubature rules. However unlike the Stroud rules the polynomial exactness of the reduced quadrature rule is not restricted to low degrees (shown here) nor is it even restricted to total-degree spaces.

[2] We adopt the most common convention of using univariate Clenshaw–Curtis rules that grow exponentially with level l. Specifically, the number of points for a level-l univariate rule satisfies $m_l = 2^l + 1$.

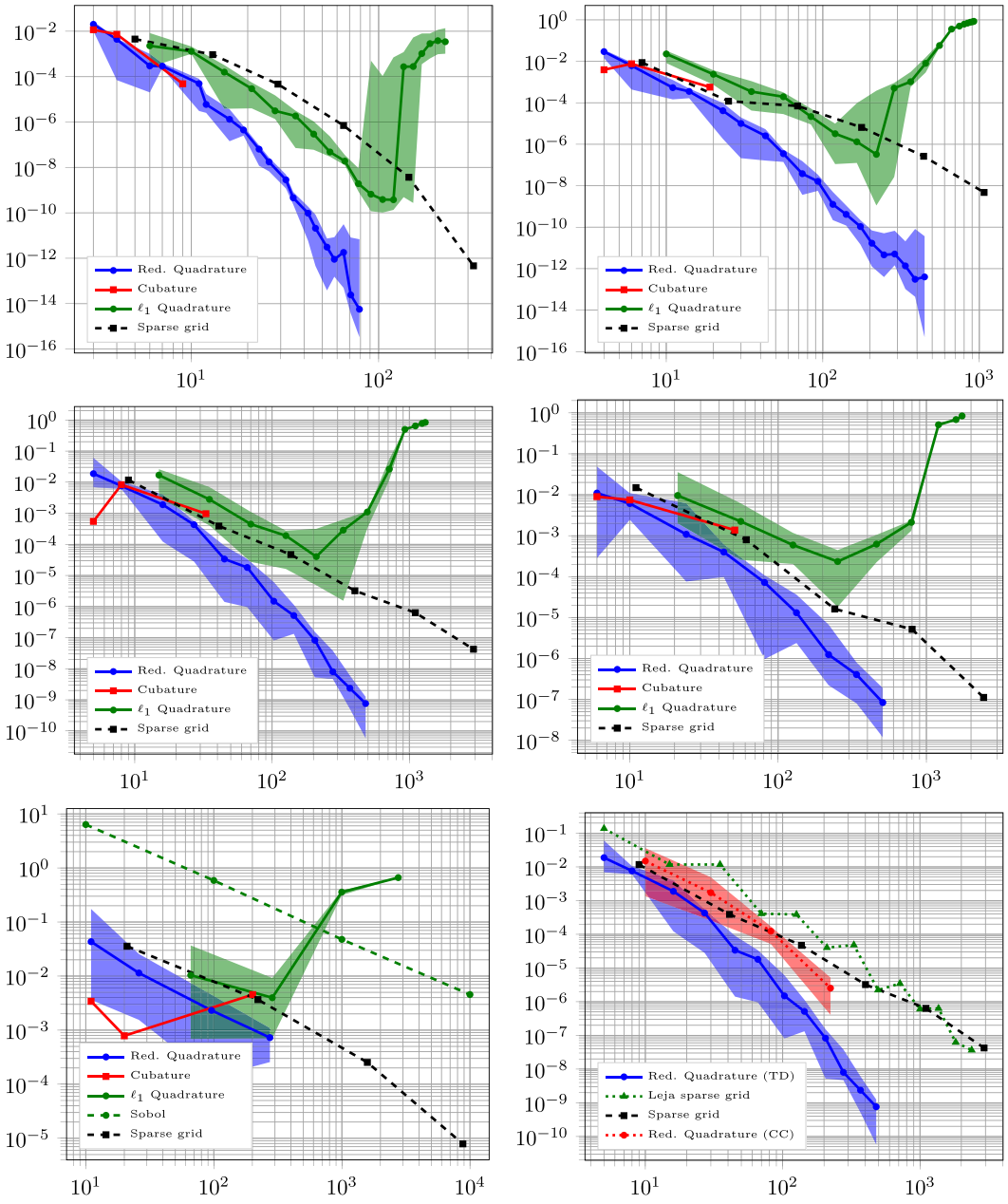

Fig. 2. Convergence of the error in the mean of the corner-peak function f_{CP} (25) computed using the reduced quadrature rule generated for the uniform measure. The horizontal axis in all plots indicates the number of quadrature points M used to approximate the integral. Convergence is shown for $d = 2, 3, 4, 5, 10$ which are respectively shown left to right starting from the top left. Solid lines represent the median error of the total-degree quadrature rules for 10 different initial conditions. Transparent bands represent the minimum and maximum errors of the same 10 repetitions. The bottom right depicts the convergence of the error in the reduced quadrature rules computed using total-degree polynomial indices (TD) and with a Clenshaw–Curtis indices (CC) in 4 dimensions ($d = 4$). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Reduced quadrature rules can be generated for arbitrary index sets, not just total degree spaces. In the bottom right plot of Fig. 2, we compare the accuracy of reduced cubature rules constructed using total-degree indices (TD) and with the index sets for which the Clenshaw–Curtis sparse grid are exact (CC). The latter index sets are hyperbolic-cross-like, and three-dimensional examples of 4th degree TD and CC index sets are shown in Fig. 3. The reduced quadrature rules based upon total-degree index sets out perform all other approaches. This result is expected because the integrand is smooth and sparse grid index sets are better suited to functions with less regularity. However even when using CC

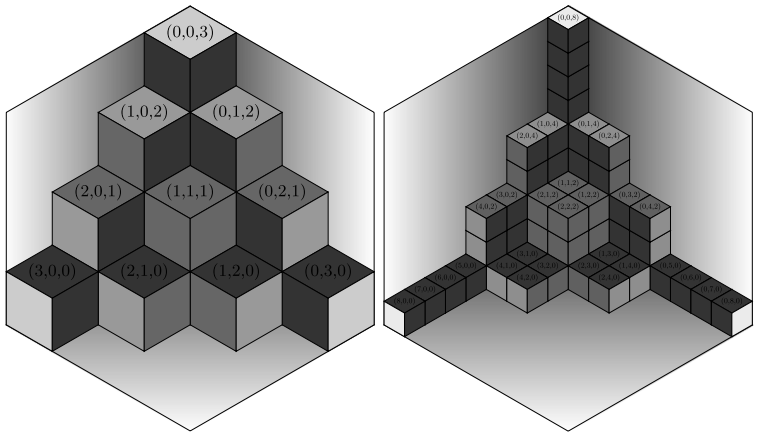

Fig. 3. Total-degree index (TD, left) and Clenshaw–Curtis index (CC, right) spaces of degree/level three in three dimensions. The four-dimensional versions of these index sets are used to define reduced quadrature rules in the bottom-right plot of Fig. 2.

index sets our reduced quadrature approach still outperforms sparse grid integration as the degree of the rule increases. The generation of reduced quadrature rules for other types of index sets is presented in Section 5.3.1.

In Fig. 2 we also compare sparse grids based on nested Leja sequences [34] which approximate integrals using total-degree index sets. Specifically the Leja sparse grid was constructed using univariate Leja sequences, for which the number of points m_l in a level l rule satisfies $m_l = l$. Approximating the integral with sparse grids, based upon total-degree polynomial spaces, does not improve the performance of the sparse grid. This is in contrast to the benefits observed when using total-degree reduced quadrature rules.

In Fig. 2 we also plot the error in the quadrature rule used as the initial condition for the local optimization used to generated the reduced rule. As expected, the error of the initial condition is larger than the final reduced rule for a given number of points. Moreover it becomes increasingly difficult to find an accurate initial condition as the degree increases. In two-dimensions, as the degree is increased past 13, the error in the initial condition begins to increase. Similar degradation in accuracy also occurs in the other higher dimensional examples. However, even when the accuracy of the initial condition still is extremely low, it still provides a good starting point for the local optimization, allowing us to accurately generate the reduced quadrature rule.

We emphasize here that there is a one-to-one correspondence between the data points in the convergence curves of the reduced quadrature rule and the ℓ_1 quadrature rules depicted in Fig. 2. (I.e., each dot on the blue curve corresponds to a dot on the green curve.) Each of the ℓ_1 quadrature rules are initial conditions used to find the corresponding reduced quadrature rules. The reduced quadrature curve does not extend to the larger samples size of the ℓ_1 quadrature rules because the reduced quadrature rules achieve the same polynomial accuracy with fewer points. This is the major advantage of our approach.

We also stress that, as pointed out in Section 5.1.1, constructing reduced quadrature rules (excluding function evaluations) is much more computationally expensive than constructing sparse grid quadrature rules. However in this paper we are targeting situations when function evaluations are significantly more expensive than the cost of building the sparse grid and one can only afford a limited number of evaluations, i.e. $\lesssim 1000$. This is commonly the case when the integrand involves the evaluation of high-fidelity physics-based simulations.

5.2. *Beyond tensor product measures*

Numerous methods have been developed for multivariate quadrature for tensor-product measures, however comparatively less attention has been given to computing quadrature rules for arbitrary measures, for example those that exhibit non-linear dependence between dimensions. In this setting Monte Carlo quadrature is the most popular method of choice.

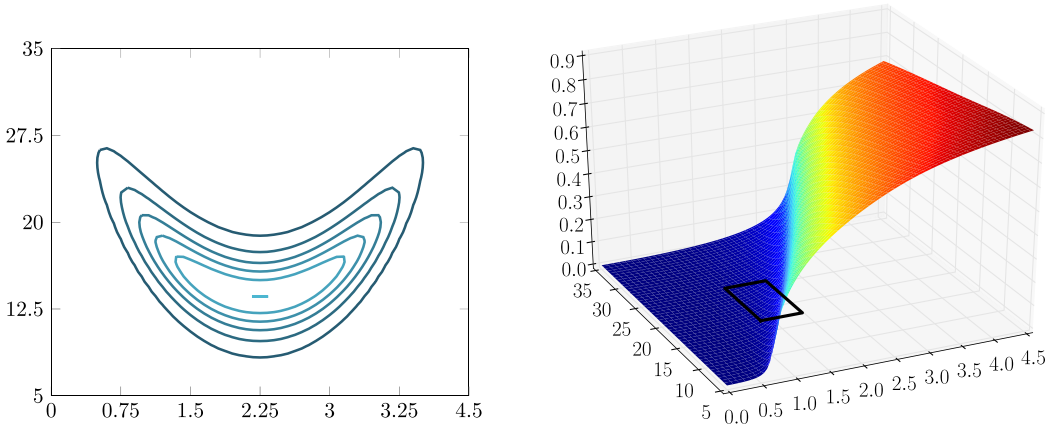

Fig. 4. (Left) Contour plot of the banana density (21) mapped to D_I. (Right) Response surface of the mass fraction u_1 at $t = 100$ predicted by the chemical reaction model. The response is plotted over the entire domain of problem I and the black box represents the smaller domain of problem II.

5.2.1. *Chemical reaction model*

The reduced quadrature method we have developed can generate quadrature rules to integrate over arbitrary measures, assuming the moments are known. Consider the following model of competing species absorbing onto a surface out of a gas phase [35]

$$\begin{aligned}
\frac{du_1}{dt} &= x_1 z - c u_1 - 4 d u_1 u_2 \\
\frac{du_2}{dt} &= 2 x_2 z^2 - 4 d u_1 u_2 \\
\frac{du_3}{dt} &= e z - f u_3 \\
z &= u_1 + u_2 + u_3, \quad u_1(0) = u_2(0) = u_3(0)
\end{aligned} \tag{20}$$

The constants c, d, e, and f are fixed at the nominal values $c = 0.04$, $d = 1.0$, $e = 0.36$, and $f = 0.016$. The parameters x_1 and x_2 will vary over a domain D endowed with a non-tensor-product probability measure μ. Viewing $X = (X_1, X_2) \in \mathbb{R}^2$ as a random variable with probability density $\mathrm{d}\mu(x)$, we are interested in computing the mean of the mass fraction of the third species $u_1(t = 100)$ at $t = 100$ s.

We will consider two test problems, problem I and problem II, each defined by its own domain D and measure μ. We therefore have two rectangular domains D_{I} and D_{II} with measures μ_{I} and μ_{II}, respectively. The domains and the measures are defined via affine mapping of a canonical domain and measure:

$$\mathrm{d}\mu(x) = C \exp(-(\frac{1}{10}x_1^4 + \frac{1}{2}(2x_2 - x_1^2)^2)), \quad x \in D = [-3, 3] \times [-2, 6], \tag{21}$$

where C is a constant chosen to normalize μ as a probability measure. This density is called a "banana density", and is a truncated non-linear transformation of a bivariate standard Normal Gaussian distribution. We define $(D_{\mathrm{I}}, \mu_{\mathrm{I}})$ and $(D_{\mathrm{II}}, \mu_{\mathrm{II}})$ as the result of affinely mapping D to the domains

$$D_{\mathrm{I}} = [0, 4.5] \times [5, 35], \qquad D_{\mathrm{II}} = [1.28, 1.92] \times [16.6, 24.9].$$

The response surface of the mass fraction over the integration domain of problem I is shown in the right of Fig. 4. The response has a strong non-linearity which makes it ideal for testing high-order polynomial quadrature. For comparison the integration domain of problem II is also depicted in the right of Fig. 4. The domain of problem II is smaller and thus the non-linearity of the response is weaker, consequently integrating the mean of problem II is easier than integrating the mean of problem II for polynomial quadrature methods.

Fig. 5 compares the convergence of the error in the mean of the mass fraction of the chemical reaction model computed using the reduced quadrature rule, with the estimates of the mean computed using Monte Carlo sampling

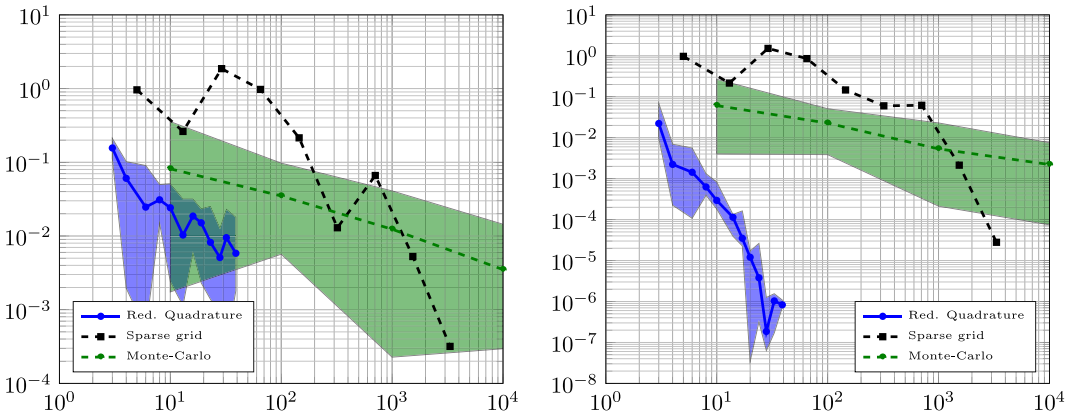

Fig. 5. Convergence of the error in the mean of the mass fraction of the chemical reaction model computed using the reduced quadrature rule generated for the banana-density over (right) D_{II} and (left) D_{I}. Error is compared to popular alternative quadrature methods.

and Clenshaw–Curtis based sparse grids. Unlike previous examples the mean cannot be computed analytically so instead we compute the mean using 10^6 samples from a 2-dimensional Sobol sequence. Sparse grids can only be constructed for tensor-product measures, so here we investigate performance of sparse grids by including the probability density in the integrand and integrating with respect to the uniform measure. This is the most common strategy to tackle a non-tensor-product integration problem using a tensor-product quadrature rule.

For the more challenging integral defined over D_{I}, the reduced quadrature error out-performs Monte Carlo and sparse grid quadrature, however the difference is more pronounced when integrating over D_{II}. The apparent slower rate of convergence of reduced quadrature for problem I is because a high-polynomial degree is needed to accurately approximate the steep response surface features over this domain. The performance of the reduced quadrature method is related to how well the integrand can be approximated by a polynomial.

5.2.2. *Sample-based moments*

Note that both generating the initial condition using (15) and solving the local optimization problem (17a) involve matching moments. Throughout this paper we have computed the moments of the polynomials we are matching analytically (or to machine precision with high-order quadrature). However situations may arise when one only has samples from the probability density of a set of random variables. For example Bayesian inference [36] is often used to infer densities of random variables conditional on available observational data. The resulting so called posterior densities are almost never tensor-product densities. Moreover it is difficult to compute analytical expressions for the posterior density and so Markov Chain Monte Carlo (MCMC) sampling is often used to draw a set of random samples from the posterior density.

Consider a model $f_{\mathrm{p}}(x) : \mathbb{R}^d \to \mathbb{R}^1$ parameterized by d variables $x = (x_1, \ldots, x_d) \in D \subset \mathbb{R}^d$, predicting an unobservable quantity of interest (QoI). There is a true underlying value of x that is unknown. In the example (20), f_p is the mass fraction u_1 of the chemical reaction model. We wish to quantify the effect of the uncertain variables on the model prediction, and we use Bayesian inference to accomplish this.

In the standard inverse problems setup, we have no direct measurements of f_p but we can make observations y_o of other quantities which we can use to constrain estimates of uncertainty in the unobservable QoI. To make this precise, let $f_{\mathrm{o}}(x) : \mathbb{R}^d \to \mathbb{R}^{n_o}$ be an observable quantity, parameterized by the same d random variables x, which predicts a set of n_o observable quantities. Bayes rule can be used to define the posterior density for the model parameters x given observational data y_o:

$$\pi(x|y_o) = \frac{\pi(y_o|x)\pi(x)}{\int_D \pi(y_o|x)\pi(x)dx}, \tag{22}$$

where any prior knowledge on the model parameters is captured through the prior density $\pi(x)$. The function $\pi(y_0|x)$ is the likelihood function and dictates an assumed model-versus-data misfit.

The construction of the posterior is often not the end goal of an analysis. Instead, one is often interested in statistics on the unobservable QoI. Here we will focus on moments of the data informed predictive distribution, for example

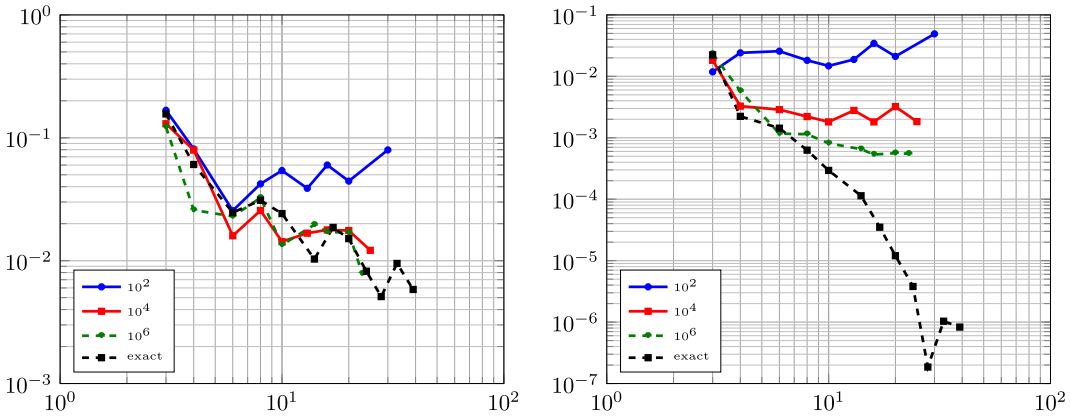

Fig. 6. Convergence of the error in the mean of the mass fraction of the chemical reaction model computed using the reduced quadrature rule generated for the banana-density using approximate moments computed with S Monte Carlo samples. The x axis in the plots indicates the number of points in a reduced quadrature rule generated from approximate moments. The numbers in the legend indicate the number of Monte Carlo samples used to approximate moments. (Right) Convergence over D_{I} and (Left) D_{I}. The "exact" lines are reproductions of the reduced quadrature (blue) lines in Fig. 5. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the mean prediction

$$m_p = \int_D f_{\mathrm{p}}(x)\pi(x|y_o)\mathrm{d}x. \tag{23}$$

In practice the posterior of the chemical model parameters may be obtained by observational data y from chemical reaction experiments and an observational model f_{o} that is used to numerically simulate those experiments. However for ease of discussion we will assume the posterior distribution (22) is given by the banana-type density defined over D_{I} or D_{II}. The banana density (21) has been used previously to test Markov Chain Monte Carlo (MCMC) methods [37] and thus is a good test case for Bayesian inference problems.

As in the previous section we will use the reduced quadrature method to compute m_p, however here we will investigate the performance of the quadrature rules that are generated from *approximate* moments. The approximate moments are those computed via Monte Carlo sampling from $\pi(x|y_o)$. These approximate moments are used to generate reduced quadrature rules, i.e. they are used as inputs when solving (15) and (17a). Fig. 6 illustrates the effect of using a finite number of samples to approximate the polynomial moments. The left of the figure depicts the convergence of the error in the mean of the mass fraction for the banana density defined over D_{I}. The right shows errors for the banana density defined over D_{II}. The right-hand figure illustrates that the accuracy of the quadrature rule is limited by the accuracy of the moments used to generate the quadrature rule. Once the error in the approximation of the mean reaches the accuracy of the moments, the error stops decreasing when the number of quadrature points is increased. The error saturation point can be roughly estimated by the typical Monte Carlo error which scales like $S^{-1/2}$, where S is the number of samples used to estimate the polynomial moments. The saturation of error present in the right-hand figure is not as evident in the left-hand figure, and this is because the error of the quadrature rules using exact moments is greater than the error in the Monte Carlo estimate of the moments using based upon 10^4 and 10^6 samples.

Our Monte Carlo samples from $\pi(x|y_o)$ were generated using rejection sampling, which is an exact sampler [38]. While MCMC is the tool of choice for sampling from high-dimensional non-standard distributions, it is an approximate sampler. If we had generated samples using MCMC, then our approximate moments contain two error sources: that from finite sample size, and that from approximate sampling. For this reason, we opted to use the exact rejection sampling technique. However, we expect that the results in Fig. 6 would look similar when using MCMC samples.

5.3. High-dimensional quadrature

The cost of computing integrals with total-degree quadrature rules increases exponentially with dimension for a fixed degree. This is because the dimension of the polynomial space $N = |\Lambda| = |B_1(k)|$ grows exponentially with d

for k fixed. This cost can be ameliorated if one is willing to consider subspaces that are more exotic than total-degree spaces. In this section we show how our reduced quadrature rules can generate quadrature for non-standard subspaces, taking advantage of certain structures that may exist in an integrand. Specifically we will demonstrate the utility of reduced quadrature rules for integrating high-dimensional functions that can be approximated by (i) low-order ANOVA expansions and (ii) by a function of a small number of linear combinations of the function variables (such functions are often referred to as ridge functions).

5.3.1. ANOVA decompositions

Dimension-wise decompositions of the input–output relationship have arisen as a successful means of delaying or even breaking the curse of dimensionality for certain applications [39–42]. Such an approach represents the model outputs as a high-dimensional function $f(x)$ dependent on the model inputs $x = (x_1, \ldots, x_d)$. One useful dimension-wise decomposition is the ANOVA decomposition

$$f(x_1, \ldots, x_d) = f_0 + \sum_{n=1}^{d} f_i(x_i) + \sum_{i,j=1}^{d} f_{i,j}(x_i, x_j) + \cdots + f_{i,\ldots,d}(x_i, \ldots, x_d) \tag{24}$$

which separates the function into a sum of subproblems. The first term is the zero-th order effect which is a constant throughout the d-dimensional variable space. The $f_i(x_i)$ terms are the first-order terms which represent the effect of each variable acting independently of all others. The $f_{i,j}(x_i, x_j)$ terms are the second-order terms which are the contributions of x_i and x_j acting together, and so on. In practice only a small number of interactions contribute significantly to the system response and consequently only a low-order approximation is needed to represent the input–output mapping accurately [43].

In this section we show that reduced quadrature rules can be constructed for functions admitting ANOVA type structure. Specifically consider the following modified version of the corner peak function (19)

$$f_{\text{MCP}} = \sum_{i=1}^{d-1} (1 + c_i x_i + c_{i+1} x_{i+1})^{-3}, \quad x \in D = [0, 1]^d \tag{25}$$

with $d = 20$. This function has at most second order ANOVA terms, and can be integrated exactly using the uniform probability measure on D.

We can create a quadrature rule ideally suited to integrating functions that have at most second-order ANOVA terms. Specifically we need only customize the index set Λ that is input to Algorithm 1. To emphasize a second-order ANOVA approximation, we compute moments of the form

$$\int_D p_\alpha(x)\, d\mu(x), \quad \forall \alpha \in \Lambda = \{\alpha \mid \|\alpha\|_0 \leq 2 \text{ and } \|\alpha\|_1 \leq k\}$$

for some degree k.

In Fig. 7 we compare the error in the mean computed using our reduced quadrature method, with the errors in the estimates of the mean computed using popular high-dimensional integration methods, specifically Clenshaw–Curtis sparse grids, Quasi Monte Carlo integration based upon Sobol sequences and low-degree (Stroud) cubature rules. By tailoring the reduced quadrature rule to the low-order ANOVA structure of the integrand, the error in the estimate of the mean is orders of magnitude smaller than the estimates computed using the alternative methods.

Note that sparse grids are a form of an anchored ANOVA expansion [39] and delay the curse of dimensionality by assigning decreasing number of samples to resolving higher order ANOVA terms. Hence, tailoring a sparse grid to the exact ANOVA structure of the function is possible.

5.3.2. Ridge functions

In this section we show that our algorithm can be used to integrate high-dimensional ridge functions. Ridge functions are multivariate functions that can be expressed as a function of a small number of linear combinations of the input variables [44]. We define a ridge function to be a function of the form $f : \mathbb{R}^d \to \mathbb{R}$ that can be expressed as a function g of $s < d$ rotated variables,

$$f(y) = g(Ay), \quad A \in \mathbb{R}^{s \times d}.$$

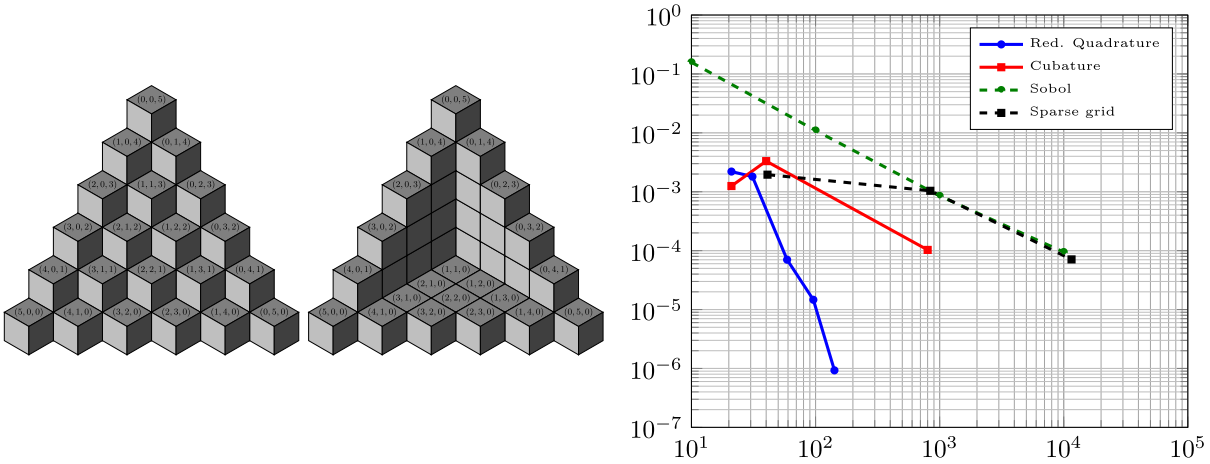

Fig. 7. (Left) Comparison of a three dimensional total degree index set of degree 5 with a 2nd-order ANOVA index set of degree 5. (Right) Convergence of the error in the mean of the modified corner-peak function f_{MCP} (25) computed using reduced quadrature rules for the uniform measure.

In applications it is common for d to be very large, but for f to be a(n approximate) ridge function with $s \ll d$. When integrating a ridge function one need not integrate in $\mathbb{R}^d$ but rather can focus on the more tractable problem of integrating in $\mathbb{R}^s$. The difficulty then becomes integrating with respect to the transformed measure of the high-dimensional integral in the lower-dimensional space, which is typically unknown and challenging to compute.

When d-dimensional space is a hypercube, then the corresponding s-dimensional domain of integration is a multivariate zonotope, i.e., a convex, centrally symmetric polytope that is the s-dimensional linear projection of a d-dimensional hypercube. The vertices of the zonotope are a subset of the vertices of the d-dimensional hypercube projected onto the s-dimensional space via the matrix A.

When computing moment-matching quadrature rules on zonotopes we must amend the local optimization problem to include linear inequality constraints, to enforce that the quadrature points chosen remain inside the zonotope. The inequality constraints of the zonotope are the same constraints that define the convex hull of the zonotope vertices.[3] Computing all the vertices of the zonotope $\{Av \mid v \in [-1,1]^d\}$ can be challenging: The number of vertices of the zonotope grows exponentially with the large dimension d. To address this issue we use a randomized algorithm [45] to find a subset of vertices of the zonotope. This algorithm produces a convex hull that is a good approximation of the true zonotope with high-probability. In all our testing we found that the approximation of the zonotope hull did not noticeably affect the accuracy of the quadrature rules we generated.

We assume that $y \in \mathbb{R}^d$ is the d-dimensional variable with a measure ν. Since d is large, ν is typically a tensor-product measure. In the projected space $x := Ay \in \mathbb{R}^s$, this induces a new measure μ on the zonotope D that is not of tensor-product form. With this setup, the μ-moments can be computed analytically by taking advantage of the relationship $x = Ay$. For further details see Appendix C.

Once a quadrature rule on a zonotope D is constructed, some further work is needed before it can be applied to integrate the function f on the original d-dimensional hypercube. Specifically we must transform the quadrature points $x \in D$ back into the hypercube. The inverse transformation of A is not unique, but if the function is (approximately) constant in the directions orthogonal to x, then any choice will do. In our example we set this transformation as $y = A^T x \in D$. The integral of the ridge function can then be approximated by

$$\int_{D_y} f(y)\, d\nu(y) \approx \sum_{i=1}^{M} f(A^T x_i) w_i$$

where (x_i, w_i) are a quadrature rule generated to integrate over the s-dimensional domain D with the non-tensor-product measure μ.

[3] Note that most non-linear least squares optimizers do not allow the specification of inequality constraints so to compute quadrature rules on a zonotope we used a sequential quadratic program.

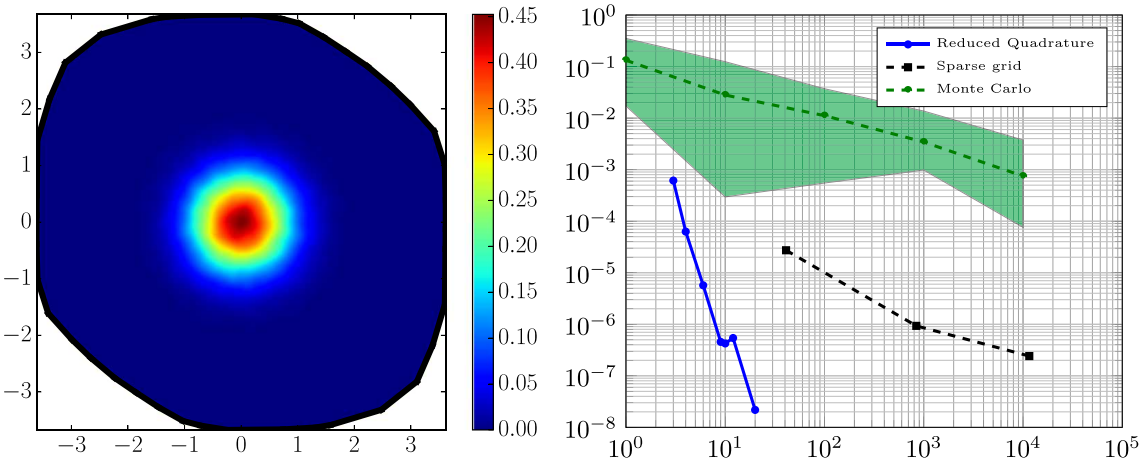

Fig. 8. (Left) The zonotope defining the domain of integration and the joint probability density of the two variables defining the two-dimensional ridge function of 20 uniform variables. The affine mapping described in Section 5.2 is used to center the mean of the zonotope density over the highly non-linear region of the chemical reaction model response surface shown in the right of Fig. 4. (Right) The convergence of the error in the mean value of the ridge function computed using reduced quadrature rules over the two-dimensional zonotope compared against more standard quadrature rules in the full 20-dimensional space.

In the following we will consider the integration of a high-dimensional ridge function with ν the uniform probability measure. We will again consider the integrating the moments mass fraction of the third species $u_1(t = 100)$ of the competing species model from Section 5.2. However now we set $x = Ay$, where y with $d = 20$ and $A \in \mathbb{R}^{2\times 20}$ is a randomly generated matrix with orthogonal rows. This makes the mass fraction a ridge function of two variables which are restricted to a zonotope within the domain D_I.

For a realization of A we plot the two resulting dimensional zonotope that defines the domain of integration of the variables x in Fig. 8 (left). The new probability density μ is depicted in the same figure. It is obvious that the transformed density μ is no longer uniform. In Fig. 8 (right) we plot the convergence of the error in the mean value of the ridge function computed using reduced quadrature rules. For a fixed number of function evaluations, the error for the reduced quadrature approach is orders of magnitudes smaller than the error obtained using Clenshaw–Curtis sparse grids and Sobol sequences in the 20 dimensional space.

6. Conclusions

In this paper we present a flexible numerical algorithm for generating polynomial quadrature rules. The algorithm employs optimization to find a quadrature rule that can exactly integrate, up to a specified optimization tolerance, a set of polynomial moments. We also provide novel lower bound for the number of nodes in a quadrature rule that is exact for a given set of polynomial moments. Quadrature rule achieving this lower bound are minimal rules. Intuition regarding this analysis is developed using a simple set of analytical multivariate examples that address existence and uniqueness of rules achieving the lower bound. In practice we often cannot computationally find an optimal quadrature rule. Typically the number of points M is only slightly larger (< 10 points) than the number of moments, N, divided by the dimension plus one, $d + 1$, i.e. $M \approx N/(d + 1)$. The algorithm we present is flexible in the sense that it can construct quadrature rules with positive weights: (i) for any measure for which moments can be computed; (ii) using analytic or sample based moments; (iii) for any set of moments from a downward-closed index space, e.g. total-degree or hyperbolic-cross polynomial spaces. We have shown that this algorithm can be used to efficiently integrate functions in a variety of settings: (i) total-degree integration on hypercubes; (ii) integration for non-tensor-product measures; (iii) high-order integration using approximate moments; (iv) ANOVA approximations; and (v) ridge function integration.

Acknowledgments

J.D. Jakeman's work was supported by DARPA EQUIPS. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly

owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

A. Narayan is partially supported by AFOSR FA9550-15-1-0467, DARPA EQUiPS N660011524053, and NSF DMS-1720416.

Appendix A. The LP algorithm

In this section we detail the reduced quadrature algorithm first presented in [9]. Let a finite index set Λ be given with $|\Lambda| = N$, and suppose that $\{p_j\}_{j=1}^N$ is a basis for P_Λ. In addition, let r be a polynomial on $\mathbb{R}^d$ such that $r \notin P_\Lambda$. We seek to find a positive Borel measure ν solving

$$\begin{aligned} &\text{minimize} \int r(x)\mathrm{d}\nu(x) \\ &\text{subject to} \int p_j(x)\mathrm{d}\nu(x) = \int p_j(x)\mathrm{d}\mu(x), \quad j = 1, \dots, N \end{aligned}$$

The authors in [9] propose the following procedure for approximating a solution:

(1) Choose a candidates mesh, $\{y_j\}_{j=1}^S$, of S points on D. Solve the much more tractable finite-dimensional linear problem

$$\begin{aligned} &\text{minimize} \sum_{k=1}^S v_k r(y_k) \\ &\text{subject to} \sum_{k=1}^S v_k p_j(y_k) = \int p_j(x)\mathrm{d}\mu(x),\ j = 1, \dots, N \\ &\quad \text{and } v_k \geq 0, k = 1, \dots, S \end{aligned}$$

The optimization is over the S scalars v_k. Let the solution to this problem be denoted v_k^*.

(2) Identify $M \leq N$ clusters from the solution above. Partition the index set $\{1, \dots, S\}$ into these M clusters, denoted C_j, $j = 1, \dots, M$. Construct M averaged points and weights from this clustering:

$$\widehat{x}_j = \frac{1}{\widehat{w}_j} \sum_{k \in C_j} v_k^* y_k, \quad \widehat{w}_j = \sum_{k \in C_j} v_k^*$$

Note that the size-M set $\{\widehat{x}_j, \widehat{w}_j\}_{j=1}^M$ is a positive quadrature rule, but it is no longer a solution to the optimization in the previous step.

(3) Solve the nonlinear optimization problem for the nodes $x_{(k)}$ and weights $w_{(k)}$, $k = 1, \dots, M$,

$$\begin{aligned} &\text{minimize} \sum_{j=1}^N \left(\int p_j(x)\mathrm{d}\mu(x) - \sum_{k=1}^M w_k p_j(x_{(k)}) \right)^2 \\ &\text{subject to } x_{(k)} \in \Gamma \text{ and } w_k \geq 0, k = 1, \dots, M \end{aligned}$$

using the initial guess $x_{(k)} \leftarrow \widehat{x}_k$ and $w_k \leftarrow \widehat{w}_k$.

We refer to the above method as the LP algorithm.

Appendix B. Reduced quadrature algorithms

This section presents pseudo code that outlines how to compute reduced quadrature rules as detailed in Section 4. Algorithm 1 presents the entire set of steps for computing reduced quadrature rules and Algorithm 2 details how the

clustering algorithm used to generate the initial condition for the local optimization used to compute the final reduced quadrature rule.

Algorithm 1: Reduced Quadrature Method

input : measure μ with and polynomial family p, index set Λ, quadrature tolerance ϵ
output: quadrature points $X = [x_1, \ldots, x_M]$ and weights $\mathbf{w} = (w_1, \ldots, w_M)^T$

With $N = |\Lambda|$, compute moments $\mathbf{m} = (m_1, \ldots, m_N)^T$,

$$m_i = \int_\Gamma p_\alpha(x)\, d\mu(x), \quad \forall \alpha \in \Lambda$$

Generate S candidate samples X_S over integration domain D;
Compute initial condition:

$$\min \|\mathbf{w}\|_1 \text{ s.t. } \mathbf{\Phi w} = \mathbf{m}, \quad \Phi_{ij} = p_\alpha(x_k),\ k \in [S],\ j \in [N]$$

Set minimum number of quadrature points

$$M = N/(d+1)$$

$i = 0$;
do
  Cluster initial condition into $M + i$ points and weights using Algorithm 2;
  Solve

  $$\text{minimize } \|\mathbf{\Phi w} - \mathbf{m}\|_2 \text{ s.t } x_k \in D,\ w_k > 0$$

  Set $i \leftarrow i + 1$
while $\|\mathbf{\Phi w} - \mathbf{m}\|_2 > \epsilon$;

Algorithm 2: Cluster Initial Quadrature Rule

input : Initial quadrature points $X = [x_1, \ldots, x_S]$ and weights $\mathbf{w} = (w_1, \ldots, w_S)^T$, number of clusters M
output: quadrature points $\widehat{X}$ and weights $\widehat{\mathbf{w}}$

$\widehat{X} = X$ and $\widehat{\mathbf{w}} = \mathbf{w}$;
while $S > M$ **do**
  $I = \arg\min_{k \in [N]} \widehat{w}_k$;
  $J = \arg\min_{k \in [N]} \|\widehat{x}_k - \widehat{x}_I\|_2$;
  Form new point as convex combination of x_I and x_J

  $$x^\star = \frac{1}{w^\star}(\widehat{w}_I \widehat{x}^I + \widehat{w}_J \widehat{x}^J), \qquad w^\star = \widehat{w}_I + \widehat{w}_J$$

  Set $\widehat{w}_I = w^\star$ and $\widehat{x}_I = x^\star$;
  Remove $\widehat{w}_J$ and $\widehat{x}_J$ from $\widehat{X}$ and $\widehat{\mathbf{w}}$;
  $S = S - 1$;
end

Appendix C. Ridge function quadrature

Consider a variable $y \in \mathbb{R}^d$ and a linear transformation $A \in \mathbb{R}^{s \times d} : \mathbb{R}^d \to \mathbb{R}^s$ which maps the variables y into a lower dimensional set of variables $x = Ay \in \mathbb{R}^s$, where $s \leq d$. When ν is a measure in $\mathbb{R}^d$, this transformation induces a measure μ in $\mathbb{R}^s$. In this section we describe how to compute the moments of the measure μ in terms of those for ν. Being able to compute such moments allows one to efficiently integrate ridge functions (see Section 5.3.2).

One approach is to approximate the moments of μ via Monte Carlo sampling. That is, generate a set of samples $Y = \{y_i\}_{i=1}^S \subset \mathbb{R}^d$ from the measure ν, and then compute a set of samples $X = AY$ in the s-dimension space. Given

a multi-index set Λ with basis p_α, $\alpha \in \Lambda$, the μ-moments of p_α can then be computed approximately via

$$\frac{1}{M}\sum_{i=1}^{M} p_\alpha(x_i) = \frac{1}{M}\sum_{i=1}^{M} p_\alpha(Ay_i).$$

Such an approach is useful when one cannot directly evaluate $\mathrm{d}\mu(x)$ but rather only has samples from the measure. However the accuracy of a moment matching quadrature rule will be limited by the accuracy of the moments that are being matched. For moments evaluated using Monte Carlo sampling the error in these moments decays slowly at a rate proportional to $M^{-1/2}$ and the variance of the polynomial p_α.

However, when the higher-dimensional measure ν is a tensor-product measure, $\nu(y) = \prod_{i=1}^{d} \nu_i(y_i)$, with each $\nu_i(x_i)$ a univariate measure, then the moments of the lower-dimensional measure μ can be computed analytically using, for example, a monomial basis.

We need to compute the moments of a monomial basis of the variables x with respect to the measure $\mu(x)$. Computing an expression of the measure $\mu(x)$ in terms of ν is difficult in general. Instead, we leverage the following equality

$$\int_D P_\alpha(y)d\nu(y) = \int_D p_\alpha(x)d\mu(x), \quad P_\alpha(y) \coloneqq p_\alpha(Ax).$$

In particular, monomials in x can be expanded in terms of the variables y, e.g $x^p = (Ay)^p$, and the resulting polynomials P_α are just products of univariate integrals which can be computed analytically or to machine precision with univariate Gaussian quadrature.

For example let $y \in [-1,1]^3$, $\nu(y)$ be the uniform probability measure, and $x = Ay$, where $A \in \mathbb{R}^{2\times 3}$ then the moment of x_1x_2 is

$$\int_D x_1x_2 d\mu(x) = \int_{[-1,1]^3} (A_{11}y_1 + A_{12}y_2 + A_{13}y_3)(A_{21}y_1 + A_{22}y_2 + A_{23}y_3)d\nu(x).$$

The right-hand side high-dimensional integrand can be expanded into sums of products of univariate terms, and thus can be integrated with univariate quadrature.

Let A have rows $a_j^T \in \mathbb{R}^d$, $j = 1,\dots,s$, and for simplicity assume that ν is a measure on $[-1,1]^d$. For a general multi-index α, we have

$$\begin{aligned}\int_D p_\alpha(x)\mathrm{d}\mu(x) = \int_D x^\alpha \mathrm{d}\mu(x) &= \int_{[-1,1]^d} \prod_{j=1}^{s} \left(a_j^T y\right)^{\alpha^{(j)}} \mathrm{d}\nu(y) \\ &= \int_{[-1,1]^d} \prod_{j=1}^{s} \left[\sum_{|\beta|=\alpha^{(j)}} \binom{\alpha^{(j)}}{\beta} a_j^\beta y^\beta\right] \mathrm{d}\nu(y), \end{aligned} \tag{C.1}$$

where we have, for a generic $\beta \in \mathbb{N}_0^d$,

$$a_j^T = \begin{pmatrix} a_{j,1} & \dots & a_{j,d}\end{pmatrix}, \qquad a_j^\beta = \prod_{k=1}^{d} a_{j,k}^{\beta^{(k)}},$$

and the multinomial coefficients

$$\binom{\alpha^{(j)}}{\beta} \coloneqq \frac{\alpha^{(j)}!}{\beta!} = \binom{\alpha^{(j)}}{\beta^{(1)},\ \beta^{(2)},\ \dots,\ \beta^{(d)}}.$$

Inspection of (C.1) and using the tensor-product structure of ν, we see that this can be evaluated exactly via sums and products of univariate integrals of y^β. While exact, this approach becomes quite expensive for large $k \coloneqq \alpha^{(j)}$ (in which case there are $\binom{k+d-1}{k-1}$ summands under the product), or when s in large (in which case one must expand an s-fold product). Nevertheless, one can use this approach for relatively large s and d since the univariate integrands in (C.1) are very inexpensive to tabulate, and it is only processing the combination of them that is expensive.

References

[1] J. Halton, On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals, Numer. Math. 2 (1960) 84–90.
[2] J. Hammersley, D. Handscomb, Monte Carlo Methods, Chapman and Hall, New York, 1964.
[3] H. Niederreiter, Random Number Generation and Quasi-Monte Carlo Methods, SIAM, Philadelphia, 1992.
[4] M. Sobol', B. Shukhman, Integration with quasi random sequences: Numerical experience, Internat. J. Modern Phys. C 6 (2) (1995) 263–275.
[5] T. Gerstner, M. Griebel, Numerical integration using sparse grids, Numer. Algorithms 18 (3–4) (1998) 209–232.
[6] T. Gerstner, M. Griebel, Dimension-adaptive tensor-product quadrature, Computing 71 (1) (2003) 65–87. http://dx.doi.org/10.1007/s00607-003-0015-5.
[7] S. Smolyak, Quadrature and interpolation formulas for tensor products of certain classes of functions, Soviet Math. Dokl. 4 (1963) 240–243.
[8] S. Mehrotra, D. Papp, Generating moment matching scenarios using optimization techniques, SIAM J. Optim. 23 (2) (2013) 963–999.
[9] E.K. Ryu, S.P. Boyd, Extensions of Gauss quadrature via linear programming, Found. Comput. Math. 15 (4) (2014) 953–971. http://dx.doi.org/10.1007/s10208-014-9197-9.
[10] V. Keshavarzzadeh, R. Kirby, A. Narayan, Numerical integration in multiple dimensions with designed quadrature, SIAM J. Sci. Comput. (2017), in press. Arxiv: https://arxiv.org/abs/1804.06501.
[11] L. van den Bos, B. Koren, R. Dwight, Non-intrusive uncertainty quantification using reduced cubature rules, J. Comput. Phys. 332 (2017) 418–445. http://dx.doi.org/10.1016/j.jcp.2016.12.011. http://www.sciencedirect.com/science/article/pii/S0021999116306647.
[12] V. Tchakaloff, Formules de cubatures mécaniques á coefficients non négatifs, Bull. Sci. Math 81 (2) (1957) 123–134.
[13] M. Arnst, R. Ghanem, E. Phipps, J. Red-Horse, Measure transformation and efficient quadrature in reduced-dimensional stochastic modeling of coupled problems, Internat. J. Numer. Methods Engrg. 92 (12) (2012) 1044–1080. http://dx.doi.org/10.1002/nme.4368.
[14] P. Constantine, E. Phipps, T. Wildey, Efficient uncertainty propagation for network multiphysics systems, Internat. J. Numer. Methods Engrg. 99 (3) (2014) 183–202. http://dx.doi.org/10.1002/nme.4667.
[15] G. Szegö, Orthogonal Polynomials, fourth ed., American Mathematical Soc., 1975.
[16] E. Novak, K. Ritter, The curse of dimension and a universal method for numerical integration, in: G. Nürnberger, J.W. Schmidt, G. Walz (Eds.), Multivariate Approximation and Splines, Birkhäuser Basel, Basel, 1997, pp. 177–187. http://dx.doi.org/10.1007/978-3-0348-8871-4_15.
[17] P. Hammer, A. Stroud, Numerical evaluation of multiple integrals II, Math. Tables Aids Comput. 12 (1958) 272–280.
[18] A. Stroud, Approximate Calculation of Multiple Integrals, Prentice-Hall, Englewood Cliffs, N.J., 1971.
[19] R. Cools, Constructing cubature formulae: the science behind the art, Acta Numer. 6 (1997) 1–54. http://dx.doi.org/10.1017/S0962492900002701.
[20] D. Xiu, Numerical integration formulas of degree two, Appl. Numer. Math. 58 (10) (2008) 1515–1520. http://dx.doi.org/10.1016/j.apnum.2007.09.004. http://linkinghub.elsevier.com/retrieve/pii/S0168927407001420.
[21] H.-J. Bungartz, M. Griebel, Sparse grids, Acta Numer. 13 (2004) 147–269. http://dx.doi.org/10.1017/S0962492904000182.
[22] D. Pflüger, B. Peherstorfer, H.-J. Bungartz, Spatially adaptive sparse grids for high-dimensional data-driven problems, J. Complexity 26 (5) (2010) 508–522. http://dx.doi.org/10.1016/j.jco.2010.04.001. http://www.sciencedirect.com/science/article/pii/S0885064X10000257.
[23] R. Caflisch, Monte Carlo and quasi-Monte Carlo methods, Acta Numer. 7 (1998) 1–49.
[24] J. Radon, Zur mechanischen Kubatur, Monatshefte für Mathematik 52 (4) (1948) 286–300. http://dx.doi.org/10.1007/BF01525334.
[25] A.H. Stroud, Quadrature methods for functions of more than one variable, Ann. New York Acad. Sci. 86 (3) (1960) 776–791. http://dx.doi.org/10.1111/j.1749-6632.1960.tb42842.x.
[26] J.L. Gout, A. Guessab, Sur les formules de quadrature numérique á nombre minimal de noeuds d'intégration, Numer. Math. 49 (4) (1986) 439–455. http://dx.doi.org/10.1007/BF01389541.
[27] A. Guessab, Cubature formulae which are exact on spaces p, intermediate between p_k and q_k, Numer. Math. 49 (5) (1986) 561–576. http://dx.doi.org/10.1007/BF01389706.
[28] R. Cools, An encyclopaedia of cubature formulas, J. Complexity 19 (3) (2003) 445–453. http://dx.doi.org/10.1016/S0885-064X(03)00011-6.
[29] G.H. Golub, J.H. Welsch, Calculation of Gauss quadrature rules, Math. Comp. 23 (106) (1969) 221–230. http://dx.doi.org/10.2307/2004418. http://www.jstor.org/stable/2004418.
[30] A. Narayan, Polynomial approximations by sampling from the spectral distribution, 2017, (Preprint).
[31] G. Golub, Some modified matrix eigenvalue problems, SIAM Rev. 15 (2) (1973) 318–334. http://dx.doi.org/10.1137/1015032.
[32] R. Tibshirani, Regression shrinkage and selection via the Lasso, J. R. Stat. Soc. Ser. B Stat. Methodol. 58 (1) (1996) 267–288.
[33] A. Genz, A package for testing multiple integration subroutines, in: P. Keas, G. Fairweather (Eds.), Numerical Integration, D. Riedel, 1987, pp. 337–340.
[34] A. Narayan, J. Jakeman, Adaptive leja sparse grid constructions for stochastic collocation and high-dimensional approximation, SIAM J. Sci. Comput. 36 (6) (2014) A2952–A2983. http://dx.doi.org/10.1137/140966368.
[35] R.D. Vigil, F.T. Willmore, Oscillatory dynamics in a heterogeneous surface reaction: Breakdown of the mean-field approximation, Phys. Rev. E 54 (1996) 1225–1231. http://dx.doi.org/10.1103/PhysRevE.54.1225.
[36] A.M. Stuart, Inverse problems: A Bayesian perspective, Acta Numer. 19 (2010) 451–559. http://dx.doi.org/10.1017/S0962492910000061.
[37] M. Parno, Y. Marzouk, Transport map accelerated Markov chain Monte Carlo, 2014, ArXiv Preprint ArXiv:1412.5492.
[38] C. Robert, G. Casella, Monte Carlo Statistical Methods, second ed., Springer-Verlag New York, 2004. http://www.springer.com/us/book/9780387212395.
[39] M. Griebel, M. Holtz, Dimension-wise integration of high-dimensional functions with applications to finance, J. Complexity 26 (5) (2010) 455–489. SI: HDA 2009.

[40] J. Foo, G. Karniadakis, Multi-element probabilistic collocation method in high dimensions, J. Comput. Phys. 229 (5) (2010) 1536–1557. http://dx.doi.org/10.1016/j.jcp.2009.10.043.

[41] X. Ma, N. Zabaras, An adaptive hierarchical sparse grid collocation algorithm for the solution of stochastic differential equations, J. Comput. Phys. 228 (2009) 3084–3113.

[42] J. Jakeman, S. Roberts, Local and dimension adaptive stochastic collocation for uncertainty quantification, in: J. Garcke, M. Griebel (Eds.), Sparse Grids and Applications, in: Lecture Notes in Computational Science and Engineering, vol. 88, Springer Berlin Heidelberg, 2013, pp. 181–203. http://dx.doi.org/10.1007/978-3-642-31703-3_9.

[43] X. Wang, I. Sloan, Why are high-dimensional finance problems often of low effective dimension, SIAM J. Sci. Comput. 27 (2005) 159–183.

[44] A. Pinkus, Ridge functions, in: Cambridge Tracts in Mathematics, Cambridge University Press, 2015. http://dx.doi.org/10.1017/CBO9781316408124.

[45] K. Stinson, D.F. Gleich, P.G. Constantine, A randomized algorithm for enumerating zonotope vertices, 2016, ArXiv Preprint ArXiv:1602.06620.