Generating nested quadrature rules with positive weights based on arbitrary sample sets

Van Den Bos, Laurent; Sanderse, Benjamin; Bierbooms, Wim; Van Bussel, Gerard

Citation (APA)
Van Den Bos, L., Sanderse, B., Bierbooms, W., & Van Bussel, G. (2020). Generating nested quadrature rules with positive weights based on arbitrary sample sets. *SIAM-ASA Journal on Uncertainty Quantification*, *8*(1), 139-169. https://doi.org/10.1137/18M1213373

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Generating Nested Quadrature Rules with Positive Weights based on Arbitrary Sample Sets[*]

Laurent van den Bos[†‡], Benjamin Sanderse[†], Wim Bierbooms[‡], and Gerard van Bussel[‡]

Abstract. For the purpose of uncertainty propagation a new quadrature rule technique is proposed that has positive weights, has high degree, and is constructed using only samples that describe the probability distribution of the uncertain parameters. Moreover, nodes can be added to the quadrature rule, resulting in a sequence of nested rules. The rule is constructed by iterating over the samples of the distribution and exploiting the null space of the Vandermonde system that describes the nodes and weights, in order to select which samples will be used as nodes in the quadrature rule. The main novelty of the quadrature rule is that it can be constructed using any number of dimensions, using any basis, in any space, and using any distribution. It is demonstrated both theoretically and numerically that the rule always has positive weights and therefore has high convergence rates for sufficiently smooth functions. The convergence properties are demonstrated by approximating the integral of the Genz test functions. The applicability of the quadrature rule to complex uncertainty propagation cases is demonstrated by determining the statistics of the flow over an airfoil governed by the Euler equations, including the case of dependent uncertain input parameters. The new quadrature rule significantly outperforms classical sparse grid methods.

1. Introduction. The problem of uncertainty propagation is considered, where the interest is in the effect of uncertainties in model inputs on model predictions. The distribution of the quantity of interest is assessed nonintrusively, i.e., by means of collocation. Problems of this form occur often in engineering applications if boundary or initial conditions are not known precisely. The canonical approach is first to identify uncertain input parameters, second to define a distribution on these parameters, and finally to determine statistics of the quantity of interest [42, 18, 23]. These statistics are defined as integrals and various techniques exist to approximate these. However, in practice it often occurs that the distribution of the uncertain parameters is known only through a sequence or collection of samples and that the distribution is possibly correlated, e.g., the distribution is inferred through Bayesian analysis. The goal

---

[†]Centrum Wiskunde & Informatica, P.O. Box 94079, 1090 GB, Amsterdam, the Netherlands (l.m.m.van.den.bos@cwi.nl, b.sanderse@cwi.nl).
[‡]Delft University of Technology, P.O. Box 5, 2600 AA, Delft, the Netherlands (l.m.m.vandenbos-1@tudelft.nl, w.a.a.m.bierbooms@tudelft.nl, g.j.w.vanbussel@tudelft.nl).

of this work is to construct collocation methods that are accurate for determining integrals when only samples of the distribution are known.

Several approaches exist to tackle problems of this type. In many cases the well-known and straightforward Monte Carlo approach is not applicable due to its low convergence rate of $1/\sqrt{N}$ (with N the number of model evaluations) and instead collocation techniques based on polynomial approximation can be constructed to alleviate this for reasonably small dimensionality. Often these techniques are based on knowledge about the input distribution, for example, its moments. A popular technique to choose evaluation nodes is the sparse grid technique [35, 26], which has been extended to a more general, correlated setting (mostly in a Bayesian setting, e.g., [31, 43, 4, 10]), provided that high order statistics of the distribution are known exactly. Other collocation techniques that can be applied to the setting in this work are techniques to consider the collocation problem as a minimization problem of an integration error [34, 16], to construct nested rules based on interpolatory Leja sequences [19, 24, 37], or to apply standard quadrature techniques after decorrelation of the distribution [25, 9]. All these approaches provide high order convergence, but require that the input distribution is explicitly known.

On the other hand, procedures that directly construct collocation sequences on samples without using the input distribution directly have seen an increase in popularity, possibly due to the recent growth of data sets. A recent example is the clustering approach proposed in [8]. Another technique is based on polynomial approximation directly based on data [27] or iteratively with a focus on large data sets [33, 41]. These approaches do not require stringent assumptions on the input distribution, but often do not provide high order convergence.

In this article, we propose a novel nested quadrature rule that has positive weights. There are various existing approaches to construct quadrature rules with positive weights. Examples include numerical optimization techniques [17, 16, 30], where oftentimes the nodes and weights are determined by minimization of the quadrature rule error. A different technique that is closely related to the approach discussed in this article is subsampling [29, 32, 36, 39], where the quadrature rule is constructed by subsampling from a larger set of nodes. Subsampling has also been used in a randomized setting [41], i.e., by randomly removing nodes from a large tensor grid, or to deduce a proof for Tchakaloff's theorem [1, 5].

The quadrature rule proposed in this work is called the *implicit quadrature rule*, because it is constructed using solely samples from the distribution. The nodes of the rule form a subset of the samples and the accompanying weights are obtained by smartly exploiting the null space of the linear system governing the quadrature weights. Using a sample set limits the accuracy of the rule to the accuracy of the sample set, but an arbitrarily sized sample set can be used without additional model evaluations. The computational cost of our proposed algorithm scales (at least) linearly in the number of samples and for each sample the null space of a Vandermonde matrix has to be determined (whose number of rows equals the number of the nodes of the quadrature rule). The main advantage of using a sample set is that the proposed quadrature rule can be applied to virtually any number of dimensions, basis, space, or distribution without affecting the computational cost of our approach. Moreover it can be extended to obtain a sequence of nested distributions, allowing for refinements that reuse existing (costly) model evaluations.

This article is set up as follows. In section 2 the nomenclature and properties of quadrature rules that are relevant for this article are discussed. In section 3 the implicit quadrature rule is introduced and its mathematical properties are discussed. The accuracy of the quadrature rule is demonstrated by integration of the Genz test functions and by determining the statistical properties of the output of a stochastic partial differential equation (PDE) modeling the flow over an airfoil. The numerical results of these test cases are discussed in section 4 and conclusions are drawn in section 5.

2. Preliminaries. The quantity of interest is modeled as a function $u \colon \Omega \to \mathbb{R}$, where Ω is a domain in $\mathbb{R}^d$ (with $d = 1, 2, 3, \dots$). The parameters $x \in \Omega$ are uncertain and their distribution is characterized by an arbitrarily large set of samples, denoted by $Y_K := \{y_0, \dots, y_K\} \subset \Omega$ (with $K \in \mathbb{N}$). In other words, the parameters x have the following discrete distribution:

$$\rho_K(x) = \frac{1}{K+1} \sum_{k=0}^{K} \delta(\|x - y_k\|),$$

where δ denotes the usual Dirac delta function and $\|\cdot\|$ denotes any norm (the only necessary property is that $\|a\| = 0$ if and only if $a \equiv 0$). The function u is not known explicitly but can be determined for specific values of $x \in \Omega$ (e.g., it is the solution of a system of PDEs). The goal is to determine statistical moments of $u(x)$, e.g., to accurately determine

$$(2.1) \qquad \mathcal{I}^{(K)}u := \int_{\Omega} u(x)\, \rho_K(x)\, \mathrm{d}x = \frac{1}{K+1} \sum_{k=0}^{K} u(y_k),$$

where higher moments can be determined by replacing $u(x)$ with $u(x)^j$ for given j. Notice that if y_k are samples drawn from a known (possibly continuous) distribution ρ, (2.1) approximates an integral weighted with this distribution, i.e.,

$$(2.2) \qquad \mathcal{I}^{(K)}u = \int_{\Omega} u(x)\, \rho_K(x)\, \mathrm{d}x \approx \mathcal{I}u := \int_{\Omega} u(x)\, \rho(x)\, \mathrm{d}x.$$

We will assume throughout this work that a large number of samples can be determined fast and efficiently or is provided beforehand. There exist various methods to construct samples from well known distributions (such as the Gaussian, Beta, and Gamma distribution) [6], from general distributions by means of acceptance rejection approaches, or from unscaled probability density functions by means of Markov chain Monte Carlo methods [22, 13]. An example of acceptance rejection sampling that we will use throughout this text to visualize our methods is depicted in Figure 1.

If $K + 1$ samples $Y_K = \{y_0, \dots, y_K\}$ are given, (2.1) could naively be evaluated by determining $u(y_k)$ for all k. However, it is well known that such an approximation is very computationally costly in many practical problems. Instead we approximate the moments by means of a quadrature rule, i.e., the goal is to determine a finite number of nodes, denoted by the indexed set $X_N = \{x_0, \dots, x_N\} \subset \Omega$, and weights, denoted by $W_N = \{w_0, \dots, w_N\} \subset \mathbb{R}$ such that

$$\mathcal{I}^{(K)}u \approx \sum_{k=0}^{N} u(x_k)w_k =: \mathcal{A}_N^{(K)}u.$$
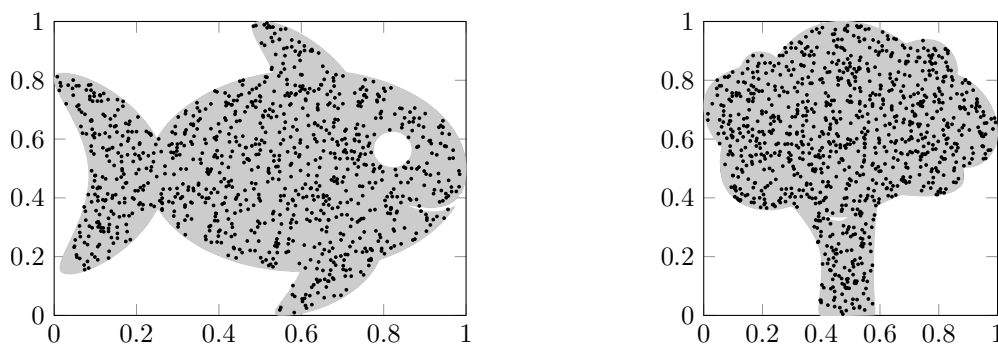
Figure 1. *The example used throughout this work: a uniform distribution restricted to the gray sets (with 1000 samples dotted in black).*

The operator $\mathcal{A}_N^{(K)}$ is the quadrature rule operator using the nodal set X_N. We omit the number of samples K from the notation if it is clear from the context.

Three properties are relevant in deriving quadrature rules: accuracy, positivity, and nesting. These properties are briefly discussed in subsections 2.1 to 2.3. The terms *nodes* and *samples* are sometimes used interchangeably in a quadrature rule setting. This is not the case in this article: samples are elements from sample sets statistically describing a distribution (called Y_K), whereas nodes are the collocation points from a quadrature rule (called X_N).

2.1. Accuracy. We enforce that the quadrature rule is accurate on a finite-dimensional function space, denoted by $\Phi_D := \mathrm{span}\{\varphi_0, \ldots, \varphi_D\}$ throughout this article. Here $\varphi_0, \ldots, \varphi_D$ are basis polynomials with $\deg \varphi_j \leq \deg \varphi_k$ for $j \leq k$, such that $\Phi_D \subset \Phi_{D+1}$ for any D. In the univariate case, this is equivalent to enforcing that the quadrature rule has degree D. In the multivariate case, the quadrature rule has at least degree Q if $\dim \Phi_D \geq \binom{Q+d}{d}$. The operator $\mathcal{A}_N^{(K)}$ is linear, hence if $D = N$ and K is given, the weights can be determined from the nodes by solving the following linear system:

$$(2.3) \qquad \mathcal{A}_N^{(K)} \varphi_j = \mathcal{I}^{(K)} \varphi_j \quad \text{for } j = 0, \ldots, D.$$

In the univariate case, this linear system is nonsingular if all nodes are distinct. This does not hold in general in the multivariate case or if $D \neq N$.

The linear system (2.3) will be used often in this work to ensure the accuracy of the constructed quadrature rules. The matrix of this system is called the (multivariate) Vandermonde matrix V_N. If a basis $\varphi_0, \ldots, \varphi_D$ is given, the system of (2.3) can be alternatively written as

$$V_D(X_N)\mathbf{w} := \begin{pmatrix} \varphi_0(x_0) & \cdots & \varphi_0(x_N) \\ \vdots & \ddots & \vdots \\ \varphi_D(x_0) & \cdots & \varphi_D(x_N) \end{pmatrix} \begin{pmatrix} w_0 \\ \vdots \\ w_N \end{pmatrix} = \begin{pmatrix} \mu_0^{(K)} \\ \vdots \\ \mu_D^{(K)} \end{pmatrix}.$$

Here, $\mu_j^{(K)}$ are known as the (multivariate) raw moments of the samples Y_K, i.e.,

$$\mu_j^{(K)} := \frac{1}{K+1} \sum_{k=0}^{K} \varphi_j(y_k) = \mathcal{I}^{(K)} \varphi_j.$$

Throughout this article it is assumed that Φ_D is a polynomial space of minimal degree and that φ_k is a monomial for each k. Multivariate polynomials are sorted using the graded reverse lexicographic order. All methods discussed in this article can also be applied if the polynomials are sorted differently (i.e., a sparse or an orthonormal basis is considered) or if the basis under consideration is not polynomial at all (e.g., sinusoidal). The only imposed restriction is that φ_0 is the constant function.

The matrix $V_D(X_N)$ might become ill-conditioned if it is constructed using monomials even for small N. Since this matrix is used to construct quadrature rules in this article, this can limit the applicability of the methods discussed here. In this article, all quadrature rules have been constructed using (products of) Legendre polynomials, which resulted in a sufficiently well-conditioned matrix for moderately large N and D.

2.2. Positivity, stability, and convergence. Any constructed quadrature rule in this article has solely positive weights for two reasons: stability and convergence. We call such a quadrature rule simply a *positive* quadrature rule. Both stability and convergence follow from the fact that the induced ∞-norm of $\mathcal{A}_N^{(K)}$ (which is the condition number of $\mathcal{A}_N^{(K)}$ as $\mu_0^{(K)} = 1$) equals the sum of the absolute weights, i.e.,

$$\|\mathcal{A}_N^{(K)}\|_\infty := \sup_{\|u\|_\infty = 1} |\mathcal{A}_N^{(K)} u| = \sum_{k=0}^{N} |w_k|, \text{ with } \|u\|_\infty := \max_{x \in \Omega} |u(x)|.$$

This norm is minimal for quadrature rules with positive weights. In these cases, we have that for all K,

$$(2.4) \qquad \|\mathcal{A}_N^{(K)}\|_\infty = \sum_{k=0}^{N} |w_k| = \sum_{k=0}^{N} w_k = 1 = \mathcal{I}^{(K)} 1.$$

If a function u is perturbed by a numerical error ε, say, $\tilde{u} = u + \varepsilon$, this does not significantly affect $\mathcal{A}_N^{(K)} u$:

$$|\mathcal{A}_N^{(K)} u - \mathcal{A}_N^{(K)} \tilde{u}| \leq \|\mathcal{A}_N^{(K)}\|_\infty \|u - \tilde{u}\|_\infty = \varepsilon.$$

This demonstrates that a quadrature rule with positive weights is numerically stable, regardless of the nodal set under consideration.

Convergence can be demonstrated similarly. This can be observed by applying the Lebesgue inequality [2]. To this end, let q_D be the *best approximation* of u in Φ_D, i.e.,

$$q_D = \arg\min_{q \in \Phi_D} \|u - q\|_\infty.$$

Here, we assume without loss of generality that this best approximation exists. By using $\mathcal{A}_N^{(K)} q_D = \mathcal{I}^{(K)} q_D$, the Lebesgue inequality follows:

$$\begin{aligned}
|\mathcal{I}^{(K)} u - \mathcal{A}_N^{(K)} u| &= |\mathcal{I}^{(K)} u - \mathcal{I}^{(K)} q_D + \mathcal{I}^{(K)} q_D - \mathcal{A}_N^{(K)} u| \\
&= |\mathcal{I}^{(K)} u - \mathcal{I}^{(K)} q_D + \mathcal{A}_N^{(K)} q_D - \mathcal{A}_N^{(K)} u| \\
&\leq |\mathcal{I}^{(K)} u - \mathcal{I}^{(K)} q_D| + |\mathcal{A}_N^{(K)} q_D - \mathcal{A}_N^{(K)} u| \\
&= |\mathcal{I}^{(K)}(u - q_D)| + |\mathcal{A}_N^{(K)}(q_D - u)| \\
&\leq \|\mathcal{I}^{(K)}\|_\infty \|u - q_D\|_\infty + \|\mathcal{A}_N^{(K)}\|_\infty \|u - q_D\|_\infty.
\end{aligned}$$

If $w_k = |w_k|$, it holds that $\|\mathcal{A}_N^{(K)}\|_\infty = \|\mathcal{I}^{(K)}\|_\infty = 1$ (see (2.4)) and convergence follows readily if $\|u - q_D\|_\infty \to 0$ for $D \to \infty$, i.e.,

$$(2.5) \qquad |\mathcal{I}^{(K)} u - \mathcal{A}_N^{(K)} u| \leq 2 \inf_{q \in \Phi_D} \|u - q\|_\infty.$$

The rate of convergence depends on the specific characteristics of u: if the space Φ_D (here polynomials) is suitable for approximating u, the error of the quadrature rule will decay fast (e.g., exponentially fast if u is analytic). It is well known that u can be approximated well using a polynomial if among others u is absolute continuous in a closed and bounded set Ω, but various other results on this topic exist [14, 38, 2].

Notice that the error of the quadrature rule $\mathcal{A}_N^{(K)} u$ with respect to $\mathcal{I}^{(K)} u$ does not depend on the accuracy of the moments $\mu_j^{(K)}$, i.e., on whether the number of samples is large enough to resolve $\mu_j^{(K)}$ accurately. This can be seen as follows. Assume the samples Y_K are drawn from a distribution $\rho \colon \Omega \to \mathbb{R}$ and let $\mathcal{I}$ be the integral from (2.2) weighted with this distribution. Even though $|\mathcal{I}\varphi_j - \mathcal{I}^{(K)}\varphi_j|$ can become large for increasing j, the error of the quadrature rule is not necessarily large:

$$|\mathcal{I}u - \mathcal{A}_N^{(K)} u| \leq \underbrace{|\mathcal{I}u - \mathcal{I}^{(K)} u|}_{\text{Sampling error}} + \underbrace{|\mathcal{I}^{(K)} u - \mathcal{A}_N^{(K)} u|}_{\text{Quadrature error}}.$$

The error depends on two components. The sampling error describes whether the number of samples is large enough to approximate the integral of u (which is independent of φ_j), whereas the quadrature error describes whether the quadrature rule is accurate (which depends on φ_j, but not through the samples; see (2.5)). The quadrature error is conceptually different from the sampling error and often decreases much faster in N than the sampling error does in K. As we assume an arbitrarily sized sequence of samples is readily available to make the sampling error sufficiently small, this article will focus on the quadrature error.

2.3. Nesting. Nesting means that $X_{N_1} \subset X_{N_2}$ for some $N_1 < N_2$, i.e., the nodes of a smaller quadrature rule are contained in a larger quadrature rule. This allows for the reuse of model evaluations if the quadrature rule is refined by considering more nodes. We will call such a quadrature rule, with a little abuse of nomenclature, a *nested* quadrature rule (because strictly speaking it is a nested *sequence* of quadrature rules).

A nested quadrature rule has the additional favorable property that it can be used to provide an error estimate of the approximated integral. If the quadrature rule converges to the true integral, i.e., $\mathcal{A}_N u \to \mathcal{I} u$, then $|\mathcal{A}_{N_1} u - \mathcal{A}_{N_2} u|$ should converge to 0:

$$|\mathcal{A}_{N_1} u - \mathcal{A}_{N_2} u| \leq |\mathcal{A}_{N_1} u - \mathcal{I} u| + |\mathcal{A}_{N_2} u - \mathcal{I} u| \to 0, \text{ for } N_1, N_2 \to \infty.$$

Therefore, the quantity $|\mathcal{A}_{N_1} u - \mathcal{A}_{N_2} u|$ can be used to estimate the accuracy of $A_{N_2} u$. If $X_{N_1} \subset X_{N_2}$, this error estimation can be calculated without any additional model evaluations.
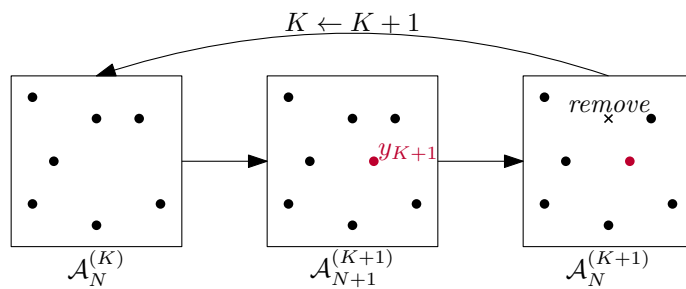
3. Implicit quadrature rule. The implicit quadrature rule is a quadrature rule that is constructed using an arbitrarily sized sequence of samples. The crucial equation in the method is (2.3), which can be written as

$$(3.1) \qquad \sum_{k=0}^{N} \varphi_j(x_k) w_k = \mu_j^{(K)}, \text{ with } \mu_j^{(K)} = \frac{1}{K+1} \sum_{k=0}^{K} \varphi_j(y_k), \text{ for } j = 0, \ldots, D.$$

Given a sequence of basis functions $\varphi_0, \ldots, \varphi_D$, the left-hand side of this equation only depends on the quadrature nodes X_N and weights W_N, whereas the right-hand side of the equation only depends on the samples Y_K. The goal is to determine, based on the $K + 1$ samples in the set Y_K, a subset of $N + 1$ samples that form the nodes X_N of a quadrature rule in such a way that (3.1) is satisfied and such that the corresponding weights are positive. The existence of such a subset is motivated by the Tchakaloff bound [5], which states that there exists a quadrature rule with positive weights with $N = D$ if Φ_D encompasses polynomials (as in this article).

The approach to determine the quadrature rule is to use an iterative algorithm: starting from an initial quadrature rule, the nodes and weights are changed iteratively while new samples y_k are added. Redundant nodes are removed while ensuring that the accuracy of the quadrature rule does not deteriorate. This iterative step, which is the key idea of the proposed algorithm, is sketched in Figure 2. By repeatedly applying this step, a quadrature rule that validates (3.1) is obtained.

Our algorithm is explained in the next two sections. First, in subsection 3.1 we propose a method for a slightly simpler problem: we fix D (or Φ_D) and determine at which nodes

Figure 2. *The implicit quadrature rule proposed in this work. Given a quadrature rule that integrates K samples ($\mathcal{A}_N^{(K)}$), a node y_{K+1} is added such that a rule is obtained of one more node ($\mathcal{A}_{N+1}^{(K+1)}$). Finally, one or more nodes are removed to obtain a quadrature rule of fewer nodes ($\mathcal{A}_N^{(K+1)}$).*

the model should be evaluated to integrate the sample moments while preserving positivity of weights. Second, in subsection 3.2 this method is extended to create sequences of *nested* quadrature rules with increasing D, increasing K, or both. In other words, given N model evaluations, we determine a subset of the samples such that (3.1) is satisfied *and* the provided N model evaluations are reused.

3.1. Fixed implicit rule. The goal is to construct a positive quadrature rule integrating all $\varphi \in \Phi_D$ exactly, where D is provided a priori. The quadrature rule will consist of (at most) $D + 1$ nodes in this case. Without loss of generality, it is assumed that $D < K$, i.e., the number of available samples is at least as large as the dimension of Φ_D.

The initial step is to consider Y_D and to construct the following quadrature rule for $N = D$:

$$X_N^{(D)} = Y_D = \{y_0, \ldots, y_D\},$$
$$W_N^{(D)} = \{1/(D+1), \ldots, 1/(D+1)\}.$$

The upper index describes the set of samples used for the construction, in this case Y_D, and the lower index describes the number of nodes of the quadrature rule (i.e., $x_0, \ldots, x_N$). This initial rule simply approximates the moments by means of Monte Carlo and it is obvious that (3.1) holds for $K = D$.

The iterative procedure works as follows. Assume $X_N^{(K)}$, $W_N^{(K)}$ form the positive quadrature rule integrating all $\varphi \in \Phi_D$ exactly. This quadrature rule has the property that $\mathcal{A}_N^{(K)} \varphi_j = \mu_j^{(K)}$ for $j = 0, \ldots, D$. The goal is to construct a quadrature rule that also has this property, but with the moments $\mu_j^{(K+1)}$ as the right-hand side. To this end, let y_{K+1} be the next sample and straightforwardly determine $X_{N+1}^{(K+1)}$ and $W_{N+1}^{(K+1)}$ as follows:

(3.2)
$$X_{N+1}^{(K+1)} = X_N^{(K)} \cup \{y_{K+1}\},$$
$$W_{N+1}^{(K+1)} = \left(\left(\frac{K+1}{K+2} \right) \cdot W_N^{(K)} \right) \cup \left\{ \frac{1}{K+2} \right\},$$

i.e., y_{K+1} is "added" to $X_N^{(K)}$ (hence $x_{N+1} = y_{K+1}$) and the weights are changed such that the quadrature rule again integrates the sample moments. The latter can be seen as follows:

$$\sum_{k=0}^{N} \varphi_j(x_k) \frac{K+1}{K+2} w_k + \frac{1}{K+2} \varphi_j(x_{N+1}) = \frac{K+1}{K+2} \sum_{k=0}^{N} \varphi_j(x_k) w_k + \frac{1}{K+2} \varphi_j(x_{N+1})$$

$$= \frac{K+1}{K+2} \left(\frac{1}{K+1} \sum_{k=0}^{K} \varphi_j(y_k) \right) + \frac{1}{K+2} \varphi_j(y_{K+1})$$

$$= \frac{1}{K+2} \sum_{k=0}^{K+1} \varphi_j(y_k) = \mu_j^{(K+1)}.$$

Here, w_k are the weights from the *original* quadrature rule, i.e., $w_k \in W_N^{(K)}$. We will use v_k to denote the weights from the updated quadrature rule, i.e., $v_k \in W_{N+1}^{(K+1)}$.

If $W_N^{(K)}$ consists of positive weights, then so does $W_{N+1}^{(K+1)}$. The problem with this simple update is that, compared to the original nodal set, the quadrature rule now requires an additional node to integrate all $\varphi \in \Phi_D$ exactly, resulting in a total of $N+2$ nodes.

In order to construct a quadrature rule that requires only $N+1$ nodes (while preserving positive weights and integrating $\mu_j^{(K+1)}$ exactly), one node will be removed from the extended rule $X_{N+1}^{(K+1)}$, following the procedure outlined in [36]. The procedure has an insightful geometric interpretation, as it is based on Carathéodory's theorem and convex cones. In this article the linear algebra interpretation is used in order to facilitate the removal of multiple nodes later in this work.

The Vandermonde matrix of the extended quadrature rule, i.e., $V_D(X_{N+1}^{(K+1)})$, is as follows:

$$V_D(X_{N+1}^{(K+1)}) = \begin{pmatrix} \varphi_0(x_0) & \dots & \varphi_0(x_N) & \varphi_0(x_{N+1}) \\ \vdots & \ddots & \vdots & \vdots \\ \varphi_D(x_0) & \dots & \varphi_D(x_N) & \varphi_D(x_{N+1}) \end{pmatrix}.$$

This is a $(D+1) \times (N+2)$-matrix (with $N = D$), so at least one nontrivial null vector $\mathbf{c} = (c_0, \dots, c_{N+1})^{\mathrm{T}}$ of this matrix exists, i.e.,

$$\begin{pmatrix} \varphi_0(x_0) & \dots & \varphi_0(x_N) & \varphi_0(x_{N+1}) \\ \vdots & \ddots & \vdots & \vdots \\ \varphi_D(x_0) & \dots & \varphi_D(x_N) & \varphi_D(x_{N+1}) \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ c_N \\ c_{N+1} \end{pmatrix} = \mathbf{0}.$$

Any multiple of $\mathbf{c}$ is also a null vector. Hence it holds for any $\alpha \in \mathbb{R}$ that

$$\begin{pmatrix} \varphi_0(x_0) & \dots & \varphi_0(x_N) & \varphi_0(x_{N+1}) \\ \vdots & \ddots & \vdots & \vdots \\ \varphi_D(x_0) & \dots & \varphi_D(x_N) & \varphi_D(x_{N+1}) \end{pmatrix} \begin{pmatrix} \alpha c_0 \\ \vdots \\ \alpha c_N \\ \alpha c_{N+1} \end{pmatrix} = \mathbf{0},$$

and by combining this with (3.1), but now for $\mu_j^{(K+1)}$, we obtain the following:

$$\begin{pmatrix} \varphi_0(x_0) & \dots & \varphi_0(x_N) & \varphi_0(x_{N+1}) \\ \vdots & \ddots & \vdots & \vdots \\ \varphi_D(x_0) & \dots & \varphi_D(x_N) & \varphi_D(x_{N+1}) \end{pmatrix} \begin{pmatrix} v_0 - \alpha c_0 \\ \vdots \\ v_N - \alpha c_N \\ v_{N+1} - \alpha c_{N+1} \end{pmatrix} = \begin{pmatrix} \mu_0^{(K+1)} \\ \vdots \\ \mu_D^{(K+1)} \end{pmatrix}.$$

This equation can be interpreted as a quadrature rule depending on the free parameter α with nodes $X_{N+1}^{(K+1)}$ and weights $\{v_k - \alpha c_k \mid k = 0, \dots, N+1\}$. The parameter α can be used to remove one node from the quadrature rule, as nodes with weight equal to zero can be removed from the quadrature rule without deteriorating it. There are two options, $\alpha = \alpha_1$ or $\alpha = \alpha_2$:

$$\alpha_1 = \min_k \left(\frac{v_k}{c_k} \mid c_k > 0 \right) =: \frac{v_{k_1}}{c_{k_1}},$$
$$\alpha_2 = \max_k \left(\frac{v_k}{c_k} \mid c_k < 0 \right) =: \frac{v_{k_2}}{c_{k_2}}.$$

The sets $\{v_k - \alpha_1 c_k\}$ and $\{v_k - \alpha_2 c_k\}$ consist of nonnegative weights and (at least) one weight equal to zero. Both α_1 and α_2 are well defined, because $\mathbf{c}$ has both positive and negative elements. The latter follows from the fact that φ_0 is assumed to be a constant and that $\mathbf{c}$ is not equal to the zero vector, i.e.,

$$0 = \sum_{k=0}^{N+1} \varphi_0(x_k) c_k = \varphi_0 \sum_{k=0}^{N+1} c_k.$$

The desired quadrature rule that integrates all $\varphi \in \Phi_D$ exactly and consists of $N = D$ nodes can be constructed by choosing either $i = 1$ or $i = 2$ and determining the nodes and weights as follows:

$$X_N^{(K+1)} = X_{N+1}^{(K+1)} \setminus \{x_{k_i}\},$$
$$W_N^{(K+1)} = \{v_k - \alpha_i c_k \mid k = 0, \ldots, k_i - 1, k_i + 1, \ldots, N + 1\}.$$

This rule has $N + 1$ nodes and integrates the moments $\mu_j^{(K+1)}$ for $j = 0, \ldots, D$ exactly. Note that to include the case of two weights becoming zero simultaneously (e.g., the symmetric quadrature rules of [36]), these sets can be implicitly defined as follows:

$$X_Q^{(K+1)} = \left\{ x_k \mid x_k \in X_{N+1}^{(K+1)} \text{ and } v_k > \alpha_i c_k \right\},$$
$$W_Q^{(K+1)} = \left\{ v_k - \alpha_i c_k \mid v_k \in W_{N+1}^{(K+1)} \text{ and } v_k > \alpha_i c_k \right\}$$

with $Q \leq N \leq D$. Without loss of generality, we assume that $Q = N$ throughout this article.

The correctness of this method follows from the fact that the first $D+1$ sample moments of the first K samples are integrated exactly using the constructed quadrature rule after iteration K. Therefore by construction the following theorem is proved.

Theorem 1. *Let $\mathcal{A}_N^{(K)}$ be a positive quadrature rule operator such that*

$$\mathcal{A}_N^{(K)} \varphi_j = \mu_j^{(K)} \text{ for } j = 0, \ldots, D$$

with $N = D$. Then after applying the procedure above, a positive quadrature rule operator $\mathcal{A}_N^{(K+1)}$ is obtained such that

$$\mathcal{A}_N^{(K+1)} \varphi_j = \mu_j^{(K+1)} \text{ for } j = 0, \ldots, D.$$

For different sample sets, even when drawn from the same distribution, the procedure constructs different quadrature rules. If desired, this nondeterministic nature of the quadrature rule can be eradicated by using deterministic samplers, such as quasi Monte Carlo sequences [3]. These are not used in the quadrature rules constructed in this article, as these sequences are generally not straightforward to construct for distributions with a noninvertible cumulative distribution function. Another aspect of the algorithm that can create variation in the resulting quadrature rules is the choice of the parameter α. It is possible to incorporate knowledge about the integrand in the choice for α at each iteration, but in this article

---

Algorithm 1. The implicit quadrature rule.

Input: Samples $\{y_0, \ldots, y_{K_{\max}}\}$, $\Phi_D = \mathrm{span}\{\varphi_0, \ldots, \varphi_D\}$
**Output:** Positive quadrature rule $X_N = \{x_0, \ldots, x_N\}$, $W_N = \{w_0, \ldots, w_N\}$ with $N = D$

1: Initialize $X_N^{(D)} = \{y_0, \ldots, y_D\}$
2: Initialize $W_N^{(D)} = \{1/(D+1), \ldots, 1/(D+1)\}$
3: **for** $K = D, \ldots, K_{\max} - 1$ **do**
4:     **Add node:** $X_{N+1}^{(K+1)} \leftarrow X_N^{(K)} \cup \{y_{K+1}\}$
5:                             $W_{N+1}^{(K+1)} \leftarrow (K+1)/(K+2)W_N^{(K)} \cup \{1/(K+2)\}$
6:     **Update weights:** Construct $V_D(X_{N+1}^{(K+1)})$
7:                             Determine (nontrivial) $\mathbf{c}$ such that $V_D(X_{N+1}^{(K+1)})\mathbf{c} = \mathbf{0}$
8:                             $\alpha_1 \leftarrow \min_k(v_k/c_k \mid c_k > 0)$, with $v_k \in W_{N+1}^{(K+1)}$
9:                             $\alpha_2 \leftarrow \max_k(v_k/c_k \mid c_k < 0)$, with $v_k \in W_{N+1}^{(K+1)}$
10:    **Choose:**   Either $\alpha \leftarrow \alpha_1$ or $\alpha \leftarrow \alpha_2$
11:    **Remove node:** $X_N^{(K+1)} \leftarrow \left\{x_k \mid x_k \in X_{N+1}^{(K+1)} \text{ and } w_k > \alpha c_k\right\}$
12:                             $W_N^{(K+1)} \leftarrow \left\{w_k - \alpha c_k \mid w_k \in W_{N+1}^{(K+1)} \text{ and } w_k > \alpha c_k\right\}$
13: **end for**
14: **Return** $X_N^{(K_{\max})}$, $W_N^{(K_{\max})}$

---

the smallest value is used, because it is assumed that we do not have a priori knowledge about the integrand.

The steps of the method are outlined in Algorithm 1 and examples of implicit quadrature rules obtained using sample sets drawn from well-known distributions are depicted in Figures 3 and 4. In Figure 3, the nodes and weights are shown for various polynomial degrees D, based on $K_{\max} = 10^5$ samples drawn from several common univariate distributions. For the distributions with compact support, the nodes cluster at the boundaries of the domain. However, the nodes exhibit an irregular pattern upon increasing the degree, and determining
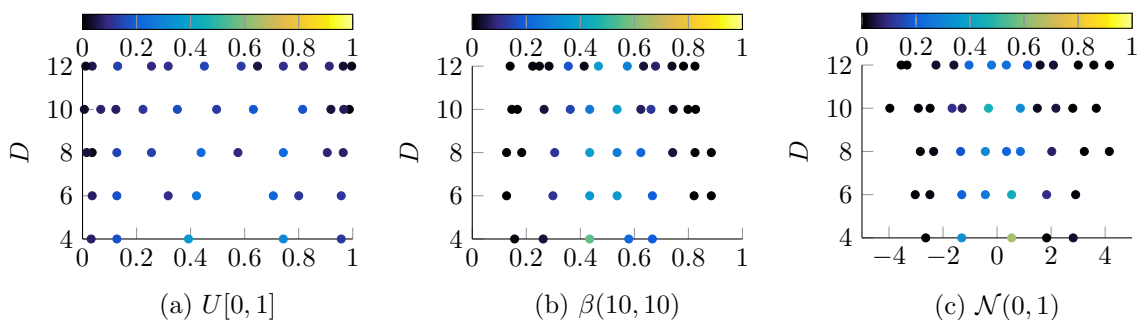
Figure 3. *Examples of implicit quadrature rules for various degrees, using the same 10^5 samples for each degree.*
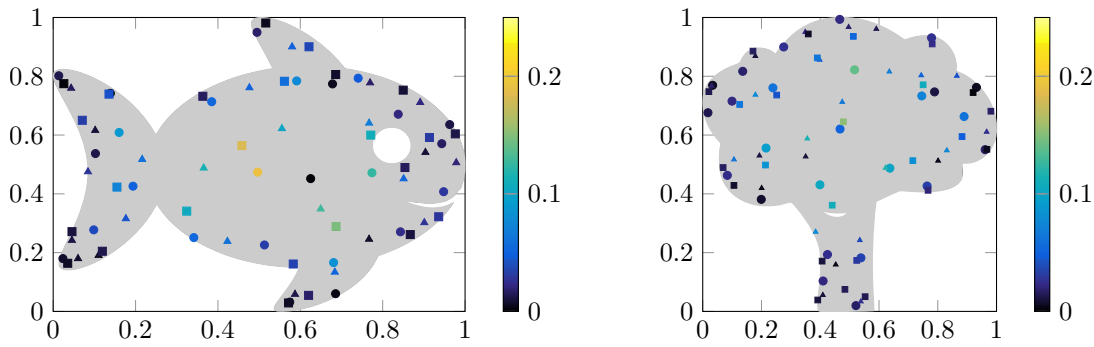
Figure 4. *Three implicit quadrature rules of 25 nodes (using different symbols) determined using the bivariate uniform distribution restricted to the gray area, using three different permutations of a set of 10^5 samples.*

a quadrature rule with a higher degree does not result in a nested rule (this will be addressed in subsection 3.2). In the second example, nodal sets are generated in two dimensions on two different irregular domains; see Figure 4. This shows a major strength of the proposed implicit quadrature rule: it can be applied to arbitrary sample sets, including domains that are not simply connected, and positive weights are still guaranteed. Depending on the ordering of the samples in the set, different nodes and weights are obtained, indicating that the quadrature rules for these sets are not unique. It is generally not possible to obtain exactly the same quadrature rule for two permutations of the sample set, since choosing either α_1 or α_2 can be exploited to preserve only a single node in the rule. Theoretically this can be resolved by removing multiple nodes from the rule, as will be done in the next section, though it is often unfeasible to do so.

3.2. Nested implicit rule. The approach of the previous section can be used to construct a quadrature rule given the number of basis vectors D and a fixed number of samples K. For varying D these quadrature rules are, however, not nested. In this section the algorithm is extended such that the constructed quadrature rules contain nodes that can be provided beforehand. By providing the nodes of an existing quadrature rule, a sequence of nested quadrature rules can be constructed.

The problem setting is as follows. Let X_N be an indexed set of quadrature rule nodes and assume a desired number of basis vectors D is specified, with $D \geq N$. The goal is to add M nodes to X_N in order to obtain a positive quadrature rule with nodes X_{N+M} (so $X_N \subset X_{N+M}$) that integrates all $\varphi \in \Phi_D$ exactly. Note that in general all weights will differ, i.e., $W_N \not\subset W_{N+M}$. We desire to add a small number of nodes, thus M to be small, but it is straightforward to observe that M is bounded as follows:

$$D \leq N + M \leq N + D + 1.$$

The first bound $D \leq N + M$ describes that a quadrature rule constructed with our algorithms does not integrate more basis functions exactly than its number of nodes. The second bound $N + M \leq N + D + 1$ describes that it is possible to simply add a quadrature rule with $D + 1$ nodes to the existing quadrature rule by setting all existing weights to 0. This is often not

desired in applications but provides a theoretical bound on the number of nodes obtained using our algorithms.

Algorithm 1 can be straightforwardly extended to incorporate nodes that are provided beforehand. The algorithm proceeds as usual, with the difference that nodes can be removed only if they were *added* during the algorithm but not if *provided* in advance. This approach yields a sequence of nested quadrature rules but is not optimal because it results in a quadrature rule with (possibly many) more nodes than necessary. Therefore the null space of the Vandermonde matrix $V_D(X_N)$ is multidimensional. In such a case there are multiple nodes that can be removed together, even though removing the nodes individually yields a quadrature rule with negative weights. For example, removing two nodes from the rule yields a positive rule, but removing only one of the two yields a negative rule.

In this section the focus is therefore on the removal step of Algorithm 1, which is extended to incorporate the removal of multiple nodes. By combining such an algorithm with Algorithm 1 the nested implicit quadrature rule is obtained.

Sequentially removing multiple nodes that result in a positive quadrature rule can result in intermediate quadrature rules with negative weights. Therefore the first step is to extend the removal procedure outlined in subsection 3.1 such that it supports negative weights. This is discussed in subsection 3.2.1. The main algorithm that generalizes the approach of the basic implicit rule is presented and discussed in subsection 3.2.2. With this algorithm, the nested implicit quadrature rule follows readily, which is discussed in subsection 3.2.3.

3.2.1. Negative weight removal. The procedure from the previous section determines α_1 and α_2 that can be used for the removal of a node. However, the equations for α_1 and α_2 were derived assuming positive weights. In this section, similar equations will be derived without assuming positive weights.

Let X_N, W_N be a quadrature rule with (possibly) negative weights. The goal is to remove one node to obtain X_{N-1} and W_{N-1} such that the resulting quadrature rule has positive weights and $\mathcal{A}_{N-1}\varphi_j = \mathcal{A}_N\varphi_j$ for $j = 0, \ldots, N-1$. As introduced before, let $V_{N-1}(X_N)$ be the respective $N \times (N+1)$ Vandermonde matrix and let $\mathbf{c} \in \mathbb{R}^{N+1}$ be a nontrivial null vector of that matrix. The goal is to have only positive weights; hence with the same reasoning as before, we obtain the following bound:

$$w_k - \alpha c_k \geq 0 \text{ for all } k \text{ and a certain } \alpha.$$

This translates into two cases:

$$\alpha \begin{cases} \geq w_k/c_k & \text{for all } k \text{ with } c_k < 0, \\ \leq w_k/c_k & \text{for all } k \text{ with } c_k > 0. \end{cases}$$

Hence the following bounds should hold for any such α:

$$\alpha_{\min} \leq \alpha \leq \alpha_{\max} \text{ with}$$

$$\alpha_{\min} = \max_k \left(\frac{w_k}{c_k} \mid c_k < 0 \right) =: \frac{w_{k_{\min}}}{c_{k_{\min}}},$$

(3.3)
$$\alpha_{\max} = \min_k \left(\frac{w_k}{c_k} \mid c_k > 0 \right) =: \frac{w_{k_{\max}}}{c_{k_{\max}}}.$$

Such α does not necessarily exist, but if it does, either $\alpha = \alpha_{\min}$ or $\alpha = \alpha_{\max}$ can be used to remove either the node $x_{k_{\min}}$ or $x_{k_{\max}}$, respectively, from the rule (as their weight becomes 0). The case with only positive weights (which was considered in subsection 3.1) fits naturally in this, with $\alpha_1 = \alpha_{\max}$ and $\alpha_2 = \alpha_{\min}$. If all weights are positive, it is evident that $\alpha_{\min} < 0 < \alpha_{\max}$.

Even a stronger, less trivial result holds: if $\alpha_{\max} < \alpha_{\min}$, then *no* node exists that results in a positive quadrature rule after removal, and if $\alpha_{\min} < \alpha_{\max}$, there exist *exactly two* nodes such that removing one of the two results in a quadrature rule with positive weights. In other words, determining $\alpha_{\min}$ and $\alpha_{\max}$ as above yields all possible nodes that can be removed resulting in a positive quadrature rule. The details are discussed in the proof of the following lemma.

Lemma 2. *Let X_N, W_N be a quadrature rule integrating all $\varphi \in \Phi_N$ exactly. The following statements are equivalent:*

1. *$\alpha_{min} \leq \alpha_{max}$.*
2. *There exists an $x_{k_0} \in X_N$ such that the quadrature rule with nodes $X_N \setminus \{x_{k_0}\}$ that integrates all $\varphi \in \Phi_{N-1}$ exactly has nonnegative weights.*
3. *Let any $x_{k_0} \in X_N$ be given such that the quadrature rule with nodes $X_N \setminus \{x_{k_0}\}$ that integrates all $\varphi \in \Phi_{N-1}$ exactly has nonnegative weights. Then the weights of this rule, say W_{N-1}, are formed by*

$$W_{N-1} = \{w_k - \alpha c_k \mid k \neq k_0\},$$

where c_k are the elements of a null vector of $V_{N-1}(X_N)$ and either $\alpha = \alpha_{min}$ or $\alpha = \alpha_{max}$.

Proof. The proof consists of three parts: $1 \rightarrow 2 \rightarrow 3 \rightarrow 1$.

$(1 \rightarrow 2)$. Proving $1 \rightarrow 2$ follows immediately from the removal step outlined above (see (3.3)).

$(2 \rightarrow 3)$. Suppose 2 holds and let x_{k_0} be given. Without loss of generality assume $k_0 = N$. Let W_{N-1} be the weights of the quadrature rule nodes $X_{N-1} = X_N \setminus \{x_N\}$ and let $w_k^{(N)} \in W_N$ and $w_k^{(N-1)} \in W_{N-1}$. It holds that the nodes X_N and weights W_N form a quadrature rule that integrates all $\varphi \in \Phi_N$ exactly and that the nodes X_{N-1} and weights W_{N-1} form a quadrature rule that integrates all $\varphi \in \Phi_{N-1}$ exactly. Therefore for $j = 0, \dots, N-1$ the following holds:

$$\sum_{k=0}^{N} \varphi_j(x_k) w_k^{(N)} = \sum_{k=0}^{N-1} \varphi_j(x_k) w_k^{(N-1)};$$

so for these j it follows that

$$\sum_{k=0}^{N-1} \varphi_j(x_k)(w_k^{(N)} - w_k^{(N-1)}) + \varphi_j(x_N) w_N^{(N)} = 0.$$

Hence the vector $\mathbf{c} \in \mathbb{R}^{N+1}$ with elements $c_k = w_k^{(N)} - w_k^{(N-1)}$ (and $c_N = w_N^{(N)}$) is a null vector of $V_{N-1}(X_N)$. Then it follows that $\alpha = 1$. Without loss of generality, assume that $c_{k_0} \neq 0$.

The remainder of this part consists of demonstrating that either $\alpha = \alpha_{\min}$ or $\alpha = \alpha_{\max}$. Assume $c_{k_0} > 0$ (with $k_0 = N$). It holds that $w_{k_0}^{(N)} = c_{k_0}$ and $w_k^{(N)} \geq c_k$ for all other k. For all k with $c_k > 0$ (including k_0), we therefore obtain

$$\frac{w_k^{(N)}}{c_k} \geq 1.$$

Equality is attained at $k = k_0$, hence $1 = \min(w_k^{(N)}/c_k \mid c_k > 0) = \alpha_{\max}$. In a similar way it can be demonstrated that if $c_{k_0} < 0$, we have $1 = \max(w_k^{(N)}/c_k \mid c_k < 0) = \alpha_{\min}$, concluding this part of the proof.

$(3 \to 1)$. Suppose 3 holds and let the weights be given as in the lemma. Let $\alpha_{\min}$, $\alpha_{\max}$, $k_{\min}$, and $k_{\max}$ be given. By definition of $\alpha_{\max}$, it holds that $c_{k_{\max}} > 0$ (see, (3.3)). So if $\alpha \geq \alpha_{\max}$, then $w_{k_{\max}} \leq \alpha c_{k_{\max}}$. So to have positive weights, we must have $\alpha \leq \alpha_{\max}$.

Similarly we have that $c_{k_{\min}} < 0$ and therefore if $\alpha \leq \alpha_{\min}$, it holds that $w_{k_{\min}} \leq \alpha c_{k_{\min}}$. So to have positive weights, we must have $\alpha \geq \alpha_{\min}$.

If there exists an α such that $\alpha_{\min} \leq \alpha$ and $\alpha \leq \alpha_{\max}$, it must hold that $\alpha_{\min} \leq \alpha_{\max}$. ∎

The lemma demonstrates that $\alpha_{\min}$ and $\alpha_{\max}$ from (3.3) can be used to determine whether there exists a node that yields a positive quadrature rule after removal (i.e., if $\alpha_{\min} \leq \alpha_{\max}$) and if such a node exists, either $\alpha_{\min}$ or $\alpha_{\max}$ can be used to determine it (by determining k_0 as in the proof). If $\alpha_{\max} > \alpha_{\min}$, no such node exists, but this is not an issue, since the algorithm to construct quadrature rules discussed in this article does by construction not end up in this case.

3.2.2. Removal of multiple nodes. Let X_N and W_N form a positive quadrature rule. In this section, the goal is to determine all subsets of M nodes that have one specific property in common: removing those M nodes results in a positive quadrature rule of $N + 1 - M$ nodes that exactly integrates all $\varphi \in \Phi_{N-M}$. We call a subset with this property an M-removal. Hence in the previous section a procedure has been presented to determine all 1-removals.

Lemma 2 is the main ingredient for deriving all M-removals. The idea boils down to the following. Let an M-removal be given, say, $(q_1, \ldots, q_M) \subset X_N$. If the first $M - 1$ nodes from this M-removal are removed, the Mth node q_M can be determined straightforwardly using $\alpha_{\min}$ or $\alpha_{\max}$ from (3.3). There are two possible values of α (namely either $\alpha_{\min}$ or $\alpha_{\max}$), hence there exists a second node, say, $\hat{q}_M$, such that $(q_1, \ldots, q_{M-1}, \hat{q}_M)$ is also an M-removal. The order in which the nodes are removed is irrelevant, so each node q_k can be replaced in this way by a different node $\hat{q}_k$ resulting in a valid M-removal, i.e., a set of M nodes that can be removed while preserving positive weights and obtaining a quadrature rule that exactly integrates all $\varphi \in \Phi_{N-M}$.

We denote the procedure of obtaining a different M-removal from an existing one by the operator $F \colon [X_N]^M \to [X_N]^M$, where $[X_N]^M$ denotes the set of all M-subsets of X_N. If $(q_1, \ldots, q_M)$ is an M-removal, applying F yields the M-removal $(q_1, \ldots, q_{M-1}, \hat{q}_M)$. Such an operator is well defined, since Lemma 2 prescribes that there exist exactly two M-removals whose first $M - 1$ elements equal $q_1, \ldots, q_{M-1}$. Notice that the operator F, which depends on the nodes and weights of the quadrature rule, can be computed by determining $\alpha_{\min}$ and $\alpha_{\max}$ from Lemma 2 after removal of $q_1, \ldots, q_{M-1}$.

By permuting the M-removal before applying F, one M-removal yields (up to a permutation at most) M other M-removals. These M-removals can be considered in a similar fashion and recursively more M-removals can be determined. This procedure yields all M-removals, which is demonstrated in the following lemma.

Lemma 3. *Let $(q_1, \ldots, q_M)$ and $(s_1, \ldots, s_M)$ be any two different M-removals of the positive quadrature rule X_N, W_N. Let the operator $F \colon [X_N]^M \to [X_N]^M$, as described in the text, be such that*

$$F(q_1, \ldots, q_{M-1}, q_M) = (q_1, \ldots, q_{M-1}, \hat{q}_M)$$

for a given M-removal $(q_1, \ldots, q_M)$, i.e., it replaces q_M by $\hat{q}_M$ such that $(q_1, \ldots, q_{M-1}, \hat{q}_M)$ is an M-removal. Then there exists a finite number of permutations $\sigma_1, \ldots, \sigma_n$ such that

$$(\sigma_1 \circ F \circ \sigma_2 \circ F \circ \cdots \circ F \circ \sigma_{n-1} \circ F \circ \sigma_n)(q_1, \ldots, q_M) = (s_1, \ldots, s_M).$$

Proof. Let $F_k \colon [X_N]^M \to [X_N]^M$ be the operator that first permutes q_k to the end of the M-removal and second applies F, i.e.,

$$F_k(q_1, \ldots, q_M) = F(q_1, \ldots, q_{k-1}, q_{k+1}, \ldots, q_M, q_k) = (q_1, \ldots, q_{k-1}, q_{k+1}, \ldots, q_M, \hat{q}_k).$$

Notice that $F_k = F \circ \pi_k$, where π_k denotes the permutation that appends q_k. Hence if there exist $k_1, \ldots, k_n$ such that $F_{k_1} \circ \cdots \circ F_{k_n}(q_1, \ldots, q_M)$ equals $(s_1, \ldots, s_M)$ up to a permutation, the proof is done.

Consider $W_N = \{w_0, \ldots, w_N\}$ and let $V_{N-M}(X_N)$ be the Vandermonde matrix with respect to this quadrature rule. This is an $(N - M + 1) \times (N + 1)$-matrix, so there exist M linearly independent null vectors $\mathbf{c}^1, \ldots, \mathbf{c}^M \in \mathbb{R}^{N+1}$. We use the following notation for the vector $\mathbf{c}^k$:

$$\mathbf{c}^k = (c_0^k, \ldots, c_N^k)^{\mathrm{T}}.$$

Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_M)$ be an M-tuple and consider the following set:

$$S = \left\{ \boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_M) \ \middle| \ \text{for all } k = 0, \ldots, N \text{ with } w_k - \sum_{j=1}^M \alpha_j c_k^j \geq 0 \right\}.$$

The set S is a closed simplex, as it is formed by a finite number of linear inequalities. Moreover it is nonempty, as $(0, \ldots, 0) \in S$ (this follows from the fact that $w_k \geq 0$ for all k).

The boundary of S, i.e., ∂S, is of special interest. If $(\alpha_1, \ldots, \alpha_M) \in \partial S$, it holds that $w_k - \sum_{j=1}^M \alpha_j c_k^j = 0$ for at least one k. At vertices of the simplex, the highest number of weights, namely M, is 0. The operator F_k can be used to traverse the vertices of the simplex. The simplex S for determining a 2-removal for a quadrature rule of three nodes is sketched in Figure 5.

Consider an M-removal $\mathbf{q} = (q_1, \ldots, q_M)$. There is exactly one M-tuple $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_M)$ resulting in the removal of these nodes. These α's coincide with a vertex of simplex S. Applying F_k to $\mathbf{q}$ yields a different M-tuple. These two M-tuples are connected through an edge of the simplex. Due to Lemma 2 all M-tuples that are connected to $\boldsymbol{\alpha}$ through an edge can be found. Therefore, for a given vertex, the operator F_k yields all connected vertices and can be used to traverse the boundary of the simplex. This concludes the proof. ∎
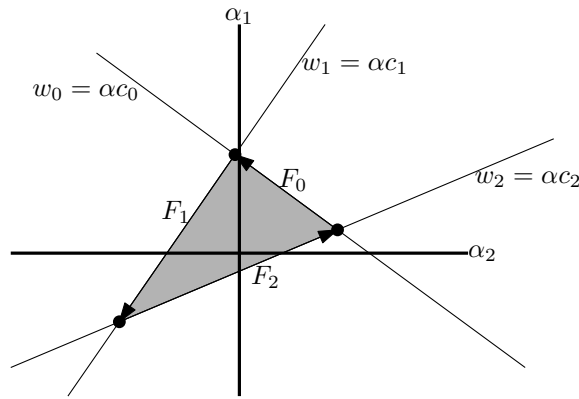
Figure 5. *Graphical sketch of the simplex describing the removal of two nodes from a quadrature rule of three nodes. The gray area describes the simplex where all values of (α_1, α_2) yield positive weights. The operator F_k (see proof of Lemma 3) can be used to traverse the boundary of the simplex.*

The statement of the lemma is constructive: given a single M-removal, all M-removals can be found. By repetitively constructing a 1-removal using the methods from subsection 3.2.1, an initial M-removal can be obtained (which is a vertex of the simplex S discussed in the proof). Lemma 3 ensures that any other M-removal can be reached from this removal. An outline of this procedure can be found in Algorithm 2. The computational cost of calculating the null vectors can be alleviated by decomposing $V_{N-M}(X_{N-M})$ (e.g., using an LU or QR decomposition) once and computing the null vectors of $V_{N-M}(X_{N-M+1})$ in the loop by reusing this decomposition.

The time complexity of this algorithm is $\mathcal{O}(Z \log Z + Z(N-M)^3)$, where Z is the number of M-removals. Here, the term $Z \log Z$ originates from storing all visited M-removals efficiently using a binary search tree (which results into Z lookups that scale with $\log Z$) and the term $Z(N-M)^3$ is obtained by factorizing $V_{N-M}(X_{N-M})$ and repeatedly computing the null vector of $V_{N-M}(X_{N-M+1})$ using this factorization. Algorithm 2 always terminates, as the number of subsets of M nodes is strictly bounded. Combining this with the proof of Lemma 3 proves the following theorem.

Theorem 4. *On termination, Algorithm 2 returns all M-removals of the positive quadrature rule X_N, W_N.*

Theoretically, Algorithm 2 can be used to determine *all* quadrature rules $\mathcal{A}_N^{(K)}$ with $\mathcal{A}_N^{(K)}\varphi = \mathcal{I}^{(K)}\varphi$ for all $\varphi \in \Phi_D$. All these rules are obtained by computing all M-removals with $M = K - N$ of the quadrature rule $X_K = Y_K$ with $w_k = 1/(K+1)$ for all $k = 0, \dots, K$. However, in practice this is intractable, as the number of M-removals grows rapidly in M and $K - N$ is typically a large quantity.

3.2.3. The nested implicit quadrature rule. In this section the key algorithm of this paper is presented, namely the nested implicit quadrature rule for arbitrary sample sets. It is constructed by combining the algorithms from the previous sections. Given a quadrature rule, two different refinements (or a combination of both) are considered. First, the number of samples K can be increased to obtain a more accurate estimate of $\mu_j^{(K)}$. Second, D can be increased to obtain a more accurate quadrature rule.

Algorithm 2. Removing multiple nodes.

Input: Positive quadrature rule X_N, W_N, integer M with $1 \leq M < N + 1$
Output: All M-removals of X_N, W_N

1: Construct $V_{N-M}(X_N)$
2: Determine M independent null vectors $\mathbf{c}^k$ of $V_{N-M}(X_N)$
3: Construct an M-removal, say $\mathbf{q} \leftarrow (q_1, \ldots, q_M) \subset X_N$ (e.g., by repeatedly using Lemma 2)
4: $I \leftarrow \{\mathbf{q}\}$, the set containing all *queued* removals
5: $R \leftarrow \emptyset$, the set containing all *processed* removals
6: **while** $I \neq \emptyset$ **do**
7:      Get the first removal from I, say $\mathbf{q} \leftarrow (q_1, \ldots, q_M) \subset X_N$
8:      Remove $\mathbf{q}$ from I, i.e., $I \leftarrow I \setminus \{\mathbf{q}\}$.
9:      **for** $i = 1, \ldots, M$ **do**
10:         Construct X_{N-M+1} and W_{N-M+1} by removing $(q_1, \ldots, q_{i-1}, q_{i+1}, \ldots, q_M)$
11:         Determine $\mathbf{c}$ such that $V_{N-M}(X_{N-M+1})\mathbf{c} = \mathbf{0}$
12:         Compute $\alpha_{\min}, \alpha_{\max}, k_{\min}, k_{\max}$ from (3.3)
13:         **if** $q_i = x_{k_{\max}}$ **then**
14:             $\hat{q}_i \leftarrow x_{k_{\min}}$
15:         **else**
16:             $\hat{q}_i \leftarrow x_{k_{\max}}$
17:         **end if**
18:         $\hat{\mathbf{q}} \leftarrow (q_1, \ldots, q_{i-1}, \hat{q}_i, q_{i+1}, \ldots, q_M)$, which is an M-removal
19:         **if** $\hat{\mathbf{q}} \notin I$ and $\hat{\mathbf{q}} \notin R$ **then**   ▷ *NB: this means we have not visited vertex $\hat{\mathbf{q}}$ yet.*
20:             Add $\hat{\mathbf{q}}$ to I, i.e., $I \leftarrow I \cup \{\hat{\mathbf{q}}\}$
21:         **end if**
22:     **end for**
23:     $R \leftarrow R \cup \{\mathbf{q}\}$
24: **end while**
25: **Return** R

The set-up is similar to the one used so far, i.e., let $\{y_k\}$ be a sequence of samples, $X_N^{(K)}$ be a set of nodes, and $W_N^{(K)}$ be a set of nonnegative weights and assume the following holds for a certain D:

$$\sum_{k=0}^{N} \varphi_j(x_k)w_k = \mu_j^{(K)}, \text{ for } j = 0, \ldots, D, \, x_k \in X_N^{(K)}, \text{ and } w_k \in W_N^{(K)}.$$

Let D_+ and K_+ be the desired number of basis vectors and (possibly larger) number of samples, respectively, and assume $D_+ \geq D$. The goal is to determine $X_{N+M}^{(K_+)}$ and $W_{N+M}^{(K_+)}$ such that $W_{N+M}^{(K_+)}$ is nonnegative, $X_N^{(K)} \subset X_{N+M}^{(K_+)}$, and

$$\sum_{k=0}^{N+M} \varphi_j(x_k)w_k = \mu_j^{(K_+)}, \text{ for } j = 0, \ldots, D_+, \, x_k \in X_{N+M}^{(K_+)}, \text{ and } w_k \in W_{N+M}^{(K_+)}.$$

In other words, we consider K_+ samples Y_{K_+} and want to determine a positive quadrature rule that integrates all $\varphi \in \Phi_{D_+}$ exactly by adding M nodes to X_N (with M minimal).

The iterative procedure is similar to Algorithm 1 and consists of four steps: (i) determine or obtain the next sample y_{K+1}, (ii) update the nodes and weights according to (3.2), (iii) determine all possible removals (see Algorithm 2), and finally (iv) remove nodes such that the obtained quadrature rule is as small as possible. The last step consists of finding the M-removal such that $X_{N+M}^{(K_+)} \setminus X_N^{(K)}$ (i.e., the set of new nodes) becomes as small as possible. If a node $x_k \in X_N^{(K)}$ is part of the M-removal, its weight is simply set to 0 (this is not problematic: the weights change again in subsequent iterations).

The initialization of the algorithm depends on whether more basis vectors are considered, a larger number of samples is considered, or the set of samples is changed (e.g., the sequence of samples is redrawn):

1. If $D_+ = D$ and $K_+ > K$, the procedure is a continuation of the original Algorithm 1 and no initialization is necessary.
2. If $D_+ > D$, we need to reiterate over all samples to determine $\mu_j^{(K)}$ for $j > D$. The algorithm can be initialized using $X_N^{(K)}$ as nodes, using all weights equal to $1/(N+1)$, and using the samples $Y_{K_+} \setminus X_D^{(K)}$.
3. If the sequence of samples is regenerated from the underlying distribution, then in general $X_N^{(K)} \not\subset Y_{K_+}$. Therefore, the algorithm is initialized with $X_N^{(K+1)} \cup \{y_0\}$ and $W_N^{(K+1)} = \{0, \ldots, 0, 1\}$.

The outline of this algorithm is provided in Algorithm 3, which is a straightforward extension of Algorithm 1 with additional bookkeeping to incorporate the removal of multiple nodes. Some examples of nested sequences are gathered in Figures 6 and 7. In the first figure all three nodal sequences are initialized with the nodes 0, 1/2, and 1. If these nodes are used to construct conventional interpolatory quadrature rules, then the quadrature rule is positive if the uniform or Beta distribution is used but has negative weights if the normal distribution is considered (the weights are 3, −4, and 2, respectively). However, the proposed algorithm incorporates these nodes without difficulty in subsequent quadrature rules, resulting in positive weights. Note that the quadrature rules of polynomial degree 4 have six nodes in case of the Beta and normal distribution. The subsequent quadrature rules of the normal distribution have a higher number of nodes than the degree, which is due to the "bad" initial set of nodes.

A two-dimensional example is presented in Figure 7. Here, the initial quadrature rule is depicted with closed circles and its extension thereof with open circles.

Again it holds that different sample sets produce different quadrature rules. Similarly as in subsection 3.1, this can be eradicated by using deterministic samplers. An additional degree of freedom arises when choosing the M-removal, as there might be several M-removals that remove the largest number of nodes from $X_{N+M}^{(K_+)} \setminus X_N^{(K)}$. In the quadrature rules constructed in this work, we select one randomly.

The large advantage of the nested implicit quadrature rule is that it is dimension agnostic, basis agnostic, space agnostic, and distribution agnostic, which are properties it carries over from the basic implicit quadrature rule. Virtually any space and any distribution can be used, as long as the distribution has finite moments and a set of samples can be generated, can be determined, or is available.

Algorithm 3. The nested implicit quadrature rule.

Input: Samples $\{y_0, \ldots, y_{K_{\max}}\}$, quadrature nodes X_N, $\Phi_D = \text{span}\{\varphi_0, \ldots, \varphi_D\}$
**Output:** Positive quadrature rule X_{N+M}, W_{N+M}

1: Initialize $X_N^{(N)}$ and $W_N^{(N)}$, e.g., $X_N^{(N)} \leftarrow X_N$, $W_N^{(N)} \leftarrow \{1/(N+1), \ldots, 1/(N+1)\}$
2: $M \leftarrow 0$
3: **for** $K = N, \ldots, K_{\max}$ **do**
4:     **Add node:** $X_{N+M+1}^{(K+1)} \leftarrow X_{N+M}^{(K)} \cup \{y_{K-N}\}$
5:                        $W_{N+M+1}^{(K+1)} \leftarrow (K+1)/(K+2)W_{N+M}^{(K)} \cup \{1/(K+2)\}$
6:                        $M \leftarrow M+1$
7:     **Update weights:** Construct $V_D(X_{N+M}^{(K+1)})$
8:                        Determine null vectors $\mathbf{c}^1, \ldots, \mathbf{c}^M$ and determine all M-removals
9:     **Choose:**   Let $\mathbf{q} = (q_1, \ldots, q_M)$ be an M-removal
                              ▷ *NB: we choose the one that makes $X_{N+M}^{(K+1)} \setminus X_N$ the smallest.*
10:    **Remove node:**   Let $(\alpha_1, \ldots, \alpha_M)$ such that $w_k = \sum_{j=1}^{M} \alpha_j c_k^j$ for all k with $x_k \in \mathbf{q}$
11:                      $\hat{M} \leftarrow \# \left\{ x_k \in X_{N+M}^{(K+1)} \mid x_k \notin X_N \text{ and } w_k - \sum_{j=1}^{M} \alpha_j c_k^j = 0 \right\}$
12:                      $X_{N+\hat{M}}^{(K+1)} \leftarrow \left\{ x_k \in X_{N+M}^{(K+1)} \mid w_k - \sum_{j=1}^{M} \alpha_j c_k^j > 0 \text{ or } x_k \in X_N \right\}$
13:                      $W_{N+\hat{M}}^{(K+1)} \leftarrow \left\{ w_k - \sum_{j=1}^{M} \alpha_j c_k^j \mid x_k \in X_{N+\hat{M}}^{(K+1)} \right\}$
14:                      $M \leftarrow \hat{M}$
15: **end for**
16: **Return** $X_{N+M}^{(K_{\max})}$, $W_{N+M}^{(K_{\max})}$

**(a)** $U[0, 1]$ **(b)** $\beta(10, 10)$ **(c)** $\mathcal{N}(0, 1)$
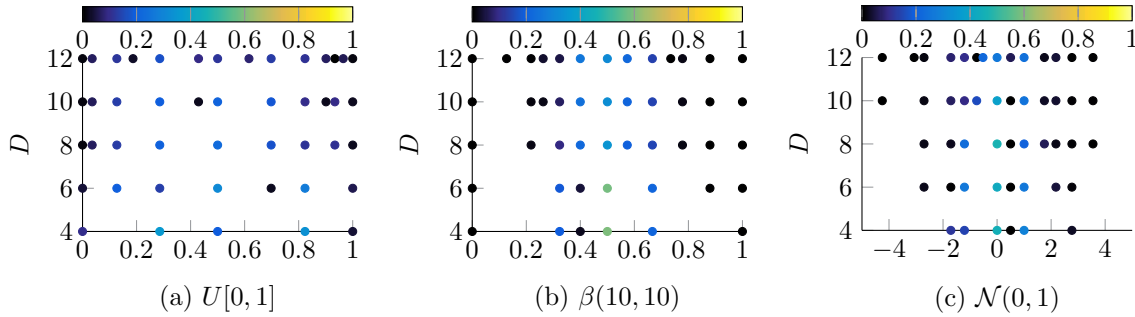
Figure 6. *Examples of nested nodal sets constructed with 10^5 samples. The initial nodes are in all three cases $[0, 1/2, 1]$.*

4. Numerical examples. Two different types of test cases are employed to demonstrate the discussed properties of our proposed quadrature rule, in particular the independence from the underlying distribution.

The first class of cases consists of explicitly known test functions and distributions to assess the accuracy of the quadrature rule for integration purposes. To this end, the Genz
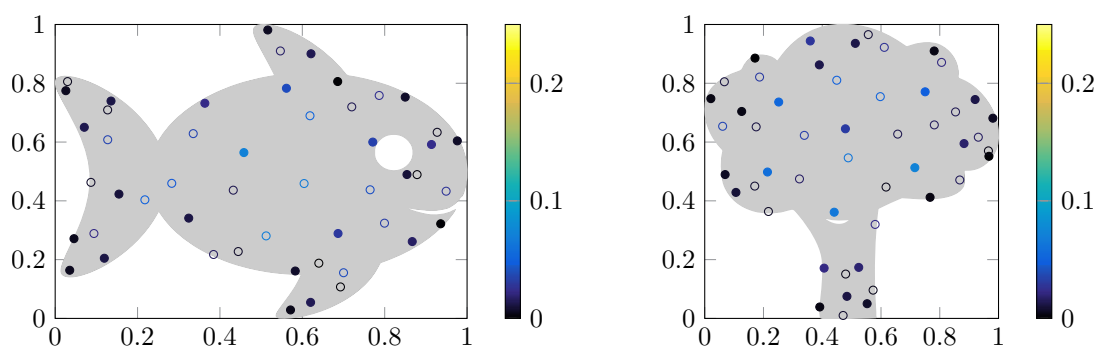
Figure 7. *The implicit quadrature rule of* 25 *nodes (closed circles) and* 50 *nodes (open and closed circles) respectively determined using the uniform distribution restricted to the gray area, using* 10^5 *samples. The colors refer to the weights of the largest quadrature rule.*

integration test functions [11] are employed and a comparison is made with a Monte Carlo approach. Moreover, it is instructive to see how the convergence compares with that of a sparse grid, although the comparison is strictly incorrect, as a Smolyak sparse grid converges to the true integral.

Second, a PDE with random coefficients is considered, where the goal is to infer statistical moments about the solution of a PDE with random boundary conditions. The equations under consideration are the inviscid Euler equations modeling the flow around an airfoil, where the inflow parameters and the shape of the airfoil are assumed to be uncertain.

The Genz integration test functions are studied in subsection 4.1. The Euler equations are considered in subsection 4.2.

4.1. Genz test functions. The Genz integration test functions [11] are a set of test function to assess the accuracy of numerical integration routines; see Table 1. Each test function has a certain attribute that is challenging for most numerical integration routines. The exact value of the integral of any of these test functions on the unit hypercube can be determined

Table 1

The test functions from Genz [11]. *All d-variate functions depend on the d-element vectors* **a** *and* **b**. *The vector* **b** *is an offset parameter to shift the function. The vector* **a** *describes the degree to which the family attribute is present.*

Integrand family	Attribute
$u_1(x) = \cos\left(2\pi b_1 + \sum_{i=1}^{d} a_i x_i\right)$	Oscillatory
$u_2(x) = \prod_{i=1}^{d} \left(a_i^{-2} + (x_i - b_i)^2\right)^{-1}$	Product peak
$u_3(x) = \left(1 + \sum_{i=1}^{d} a_i x_i\right)^{-(d+1)}$	Corner peak
$u_4(x) = \exp\left(-\sum_{i=1}^{d} a_i^2 (x_i - b_i)^2\right)$	Gaussian
$u_5(x) = \exp\left(-\sum_{i=1}^{d} a_i |x_i - b_i|\right)$	C_0 function
$u_6(x) = \begin{cases} 0 & \text{if } x_1 > b_1 \text{ or } x_2 > b_2 \\ \exp\left(\sum_{i=1}^{d} a_i x_i\right) & \text{otherwise} \end{cases}$	Discontinuous

exactly. In this section, these functions are used to test the implicit quadrature rule. The goal is to assess the absolute integration error for increasing number of nodes in a five-dimensional setting, i.e., to assess

$$e_N = \left| \mathcal{A}_N^{(K_{\max})} u - \frac{1}{K_{\max} + 1} \sum_{k=0}^{K_{\max}} u(y_k) \right|,$$

for samples $y_0, \ldots, y_{K_{\max}}$ and various increasing N. The number of samples is chosen such that the quadrature error dominates and the sampling error $|\mathcal{I}^{(K_{\max})} u - \mathcal{I} u|$ is small. We compare the approximation with that of a Monte Carlo approach, where we assess the following error:

$$e_N = \left| \frac{1}{N + 1} \sum_{k=0}^{N} u(y_k) - \frac{1}{K_{\max} + 1} \sum_{k=0}^{K_{\max}} u(y_k) \right|,$$

i.e., it is considered as a quadrature rule with nodes $\{y_k\}$ and weights $1/(N+1)$. If the underlying distribution is tensorized, we also study the Smolyak sparse grid, which is constructed using exponentially growing Clenshaw–Curtis quadrature rules in conjunction with the combination rule [26]. The sparse grid converges to the true value of the integral and therefore we use the true value of the integral to assess its convergence, even though the comparison is not completely fair in this case.

The numerical experiment is repeated twice for two different input distributions. First, the uniform distribution is used to be able to compare the methodology with conventional quadrature rule methods. Second, a highly correlated multivariate distribution (inspired by Rosenbrock function) is used to demonstrate the independence of the convergence rate from the input distribution.

To obtain meaningful results, the offset and shape parameters $\mathbf{a}$ and $\mathbf{b}$ of the Genz functions are chosen randomly and the numerical experiment is repeated 50 times. The obtained 50 absolute integration errors are averaged. The vector $\mathbf{a}$ is obtained by first sampling uniformly from $[0,1]^5$ and second scaling $\mathbf{a}$ such that $\|\mathbf{a}\|_2 = 5/2$. The vector $\mathbf{b}$ is uniformly distributed in $[0,1]^5$ without further scaling, as it is an offset parameter.

The implicit quadrature rule is generated with $K_{\max} = 10^4$ samples drawn randomly from the two input distributions respectively and the Monte Carlo approximation is determined using a subset of these samples, such that both the Monte Carlo approximation and the implicit quadrature rule converge to the same result. The initial quadrature rule of one single node is determined randomly and the rule is extended by applying Algorithm 3. Each extension is such that D doubles, up to $D = 2^{10} = 1024$, but we emphasize that any granularity can be used here. Recall that $\Phi_D = \mathrm{span}\{\varphi_0, \ldots, \varphi_D\}$, where φ_j are d-variate polynomials sorted graded reverse lexicographically. Hence each extension integrates a larger number of polynomials exactly. For sake of completeness, a comparison is made with a nonnested implicit quadrature rule, which is regenerated for each D by means of Algorithm 1.

4.1.1. Uniform distribution. The multivariate uniform distribution in $[0,1]^d$ (with $d = 5$ in this case) can be constructed by means of a tensor product of multiple univariate uniform distributions. It is therefore possible to approximate the integral using the well-known Smolyak sparse grid. The results of the four integration routines under consideration

(Monte Carlo, nested and nonnested implicit quadrature rules and Smolyak sparse grid) are depicted in Figure 8. Here, N denotes the number of nodes of the quadrature rules and the Smolyak sparse grid is refined by increasing the sparse grid level equally in all dimensions.

The accuracy of a quadrature rule is highly dependent on the analyticity and smoothness of the integrand. Globally analytic functions can be approximated well using polynomials, i.e., $\inf_{q \in \Phi_D} \|q - u\|_\infty$ decays fast. This property is reflected in the results.

The first four Genz functions (i.e., u_1, u_2, u_3, and u_4) are smooth and therefore the most suitable for integration by means of a quadrature rule. The best convergence is observed for the oscillatory, product peak, and Gaussian function, which are analytic. The corner peak is analytic, but has very slowly decaying derivatives, such that the quadrature rule approximation only converges exponentially fast for very large numbers of nodes (which are not considered here).

The continuous (but not differentiable) C_0 function follows a similar reasoning. It is not globally analytic, hence no exponential convergence is obtained. The Smolyak quadrature rule has a slightly larger error in this case compared to the implicit quadrature rule (arguably due to its negative weights), even though it seems that the rate of convergence is similar.

Integrating the discontinuous function by means of a positive quadrature rule does not yield any improvement over Monte Carlo sampling. The Smolyak sparse grid performs worse in this case due to its negative weights and usage of the Clenshaw–Curtis quadrature rule (which is not suitable for integration of discontinuous functions).

4.1.2. Rosenbrock distribution. The large advantage of the implicit quadrature rule is that it can be constructed using any arbitrary set of samples. In order to assess this applicability to general distributions, the following distribution (which we will call the Rosenbrock distribution) is considered:

$$\rho \colon \mathbb{R}^d \to \mathbb{R}, \text{ defined by } \rho(x) \propto \exp\left[-f(x)\right] \pi(x),$$

with π the PDF of the multivariate standard Gaussian distribution and f (a variant of) the multivariate Rosenbrock function:

$$f(x) = f(x_1, \ldots, x_d) = \sum_{i=1}^{d-1} \left[b\left(x_{i+1} - x_i^2\right)^2 + (a - x_i)^2\right] \text{ with } a = 1 \text{ and } b = 10.$$

The distribution ρ for $d = 2$ is depicted in Figure 9. This distribution is not optimal for integration by means of a sparse grid as it is not tensorized. Integration by means of a sparse grid converges prohibitively slow, even if the quadrature rules used for the construction are based on the marginals of the distribution. Therefore these results are omitted.

The *exact* integral over the corner peak function u_3 diverges in this case, so approximating such an integral will result in a diverging quadrature rule. The results of the other functions are gathered in Figure 10.

Similarly to the uniform case, the properties of the functions are reflected in the convergence rates of the approximations. The integrals of the smooth functions converge fast with a high rate, the convergence of the C_0 function is smaller, and the convergence of the discontinuous function is comparable to that of Monte Carlo. This result is significant, as it demonstrates that the convergence rate of $\mathcal{A}_N^{(K)}$ to the sampling-based integral $\mathcal{I}^{(K)}$ for $N \to K$ shows no significant dependence on the sample set used to construct the rules.

(a) u_1 (b) u_2

(c) u_3 (d) u_4

(e) u_5 (f) u_6
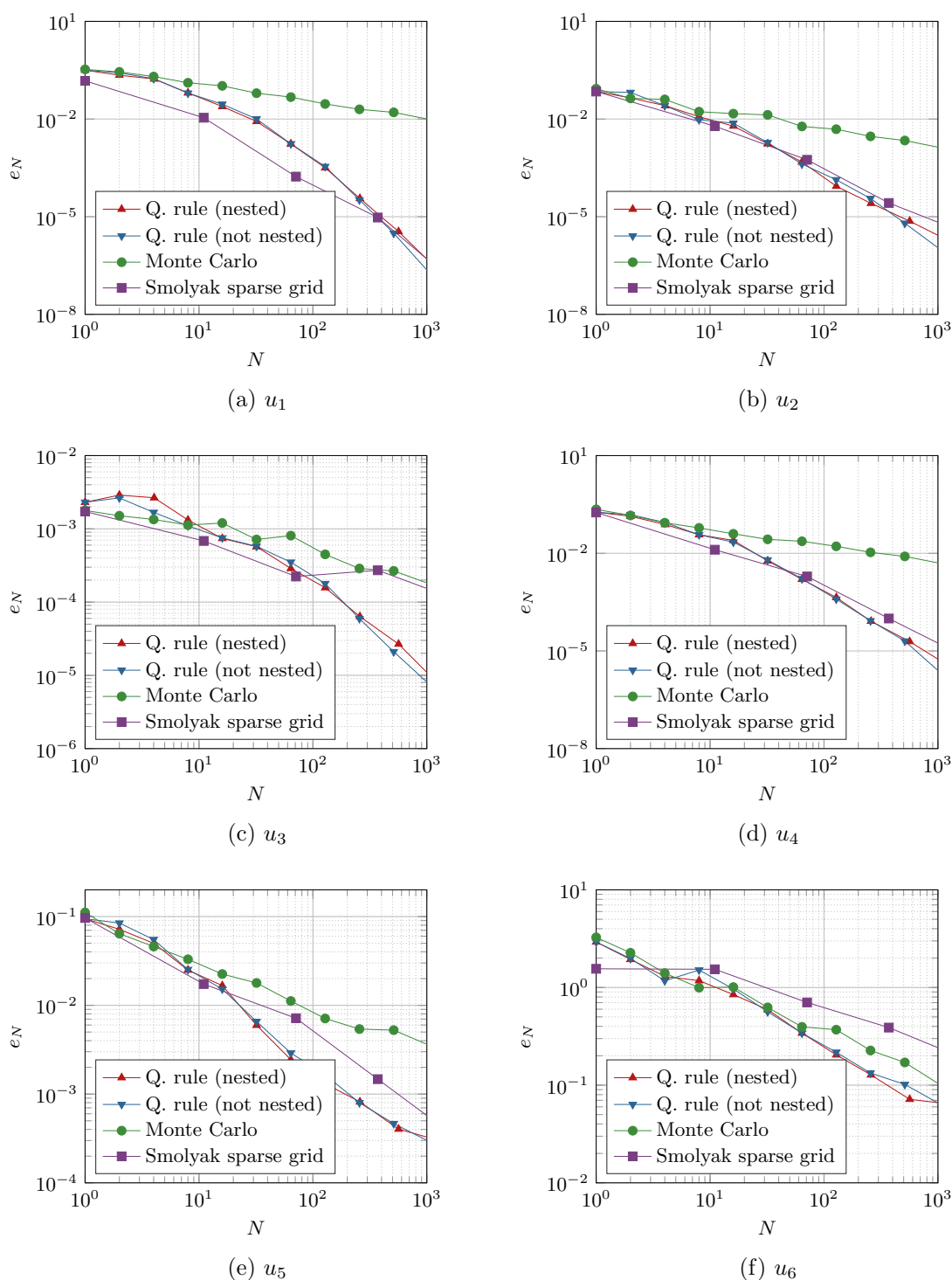
Figure 8. *Convergence of the absolute integration error for Genz test functions using the nested and nonnested implicit quadrature rules, Monte Carlo sampling, and the Smolyak sparse grid using the uniform distribution.*
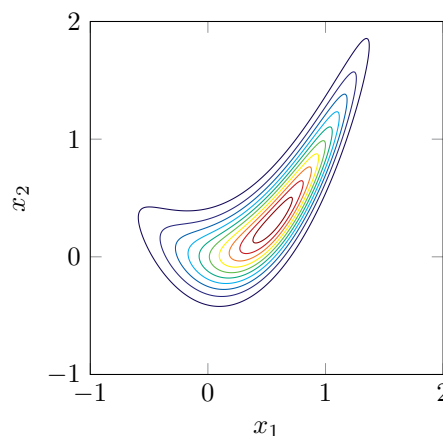
Figure 9. *The bivariate Rosenbrock distribution.*

4.2. Airfoil flow using Euler equations. In this section the flow over an airfoil is considered with uncertain geometry and inflow conditions. The quantity of interest is the pressure coefficient of the airfoil. The problem is five-dimensional: two parameters model uncertain environmental conditions and three parameters model the uncertain geometry of the airfoil. The geometry is described by the 4-digit NACA profile and the equations governing the flow are the inviscid Euler equations. Problems of this type are well known in the framework of uncertainty propagation [20, 40, 21] and allow us to demonstrate the applicability of the proposed quadrature rule to a complex uncertainty propagation test case in conjunction with a complex underlying distribution.

The five uncertain parameters are summarized in Table 2. The angle of attack and Mach number are distributed independently from the other parameters and describe uncertain inflow conditions. The remaining three parameters define the 4-digit NACA airfoil [15]. A NACA airfoil can be generated directly from these parameters and the mean of these parameters is approximately a NACA2312 airfoil. In this work the NACA airfoil with closed tip is considered by correction of the last parameter.

The compressible Euler equations are numerically solved using the finite volume solver SU2 [7, 28]. The mesh is generated using gmsh [12]. The implicit quadrature rules are determined by means of $K_{\max} = 10^6$ randomly drawn samples from the distributions described in Table 2, which are also being reused for consecutive refinements.

The quantity of interest in this test case is the pressure coefficient on the surface of the airfoil $C_p(x)$, i.e., the scaled pressure such that zero pressure equals a nonobstructed flow:

$$C_p(x) = \frac{p(x) - p_\infty}{\frac{1}{2}\rho_\infty V_\infty^2}.$$

Here, $p(x)$ is the pressure at location x, p_∞ is the freestream pressure (i.e., the pressure on the boundary in this case), ρ_∞ is the freestream density of air, and V_∞ is the freestream velocity of the fluid. The quadrature rule is applied piecewise to this quantity, where all pressure realizations are piecewise linearly interpolated onto the same mesh. Accuracy is measured by

(a) u_1

(b) u_2

(integral diverges)

(c) u_3

(d) u_4

(e) u_5

(f) u_6
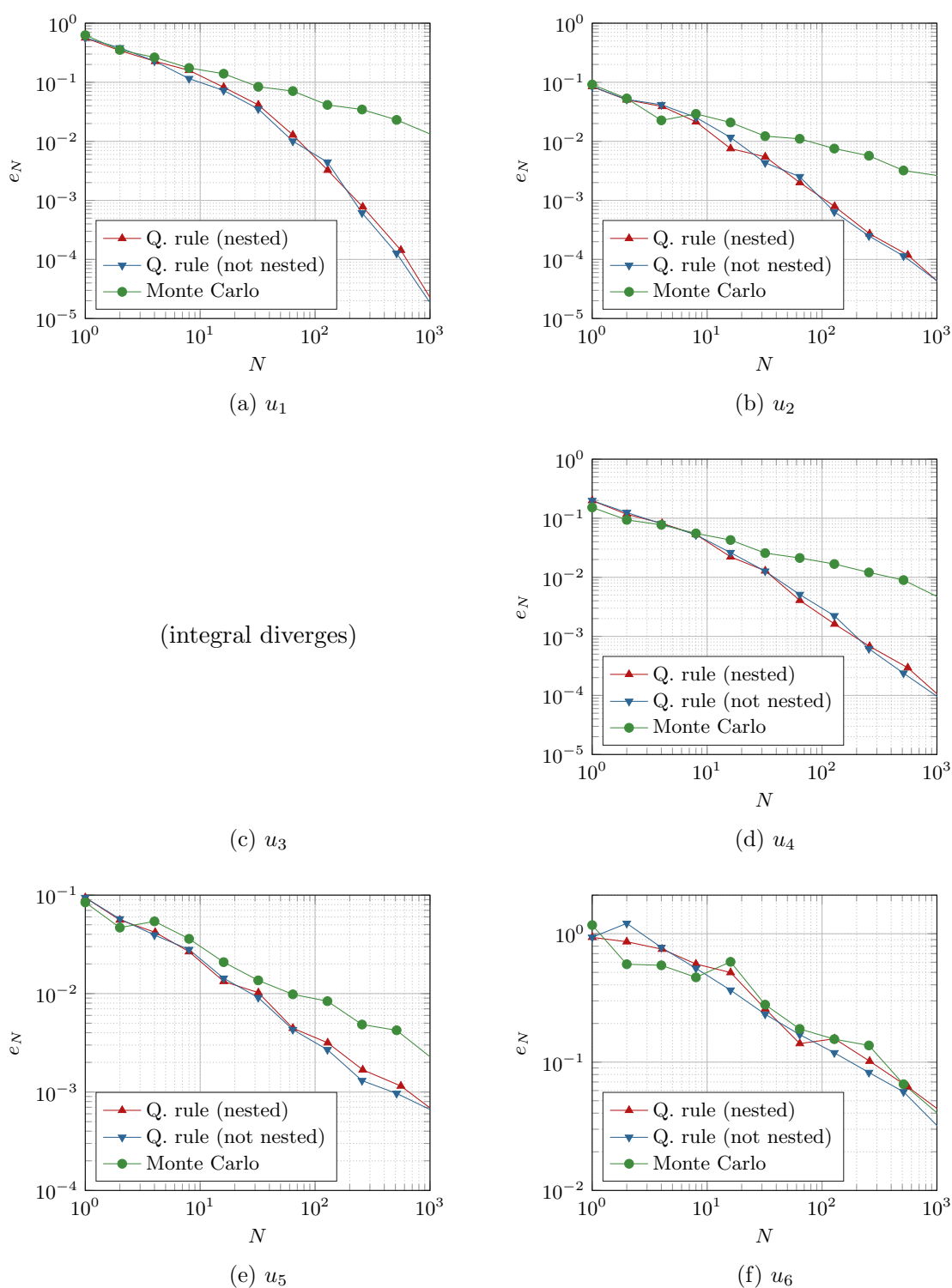
Figure 10. *Convergence of the absolute integration error for Genz test function using the nested and nonnested implicit quadrature rules and Monte Carlo sampling using the Rosenbrock distribution.*

Table 2
Uncertain parameters of the airfoil test case.

|	Parameter	Distribution
α	Angle of attack	Uniform in $[0°, 5°]$
M	Mach number	Beta$(4, 4)$ distributed in $[0.4, 0.6]$
t	Maximum thickness of the airfoil	Beta$(4, 4)$ distributed in $[0.11, 0.13]$
m	Maximum camber	Beta$(2, \cdot)$ distributed in $[0.02, 0.03]$ with mean: $\bar{m} = 1/4t^2 + 0.02$
p	Location of maximum camber	Uniform distributed in $[0.3, p_{max}]$ with $p_{max} = \frac{1}{2}(2t + 16m)^5 + 0.3$

using the lift coefficient, which is the dimensionless coefficient relating the lift generated by the airfoil with the farfield fluid density and velocity. It follows naturally by integrating the pressure coefficient over the airfoil surface. In Figure 11(a) we used the quadrature rule $\mathcal{A}_N u$ with $u = C_p$ and in Figure 11(b) we used the quadrature rule with $u = C_l$.

The number of nodes of the quadrature rule is doubled until the difference between two consecutive quadrature rule estimations of the mean of the pressure coefficient is smaller than 10^{-2}, which is the case at 512 nodes. The results are summarized in Figure 11 and high order convergence is clearly visible. Both the mean and the standard deviation show convergence with a larger rate than that of Monte Carlo (i.e., larger than $1/2$). We want to emphasize the importance of positive weights for this engineering test case, as it ensures the estimation of the variance is nonnegative even in the presence of high nonlinearities.

The moments of the pressure coefficient around the airfoil are depicted in Figure 12, where the airfoil geometry that is plotted is the overlap of all airfoils (such that all depicted flow locations are in the flow for all quadrature rule nodes).

(a)

(b)
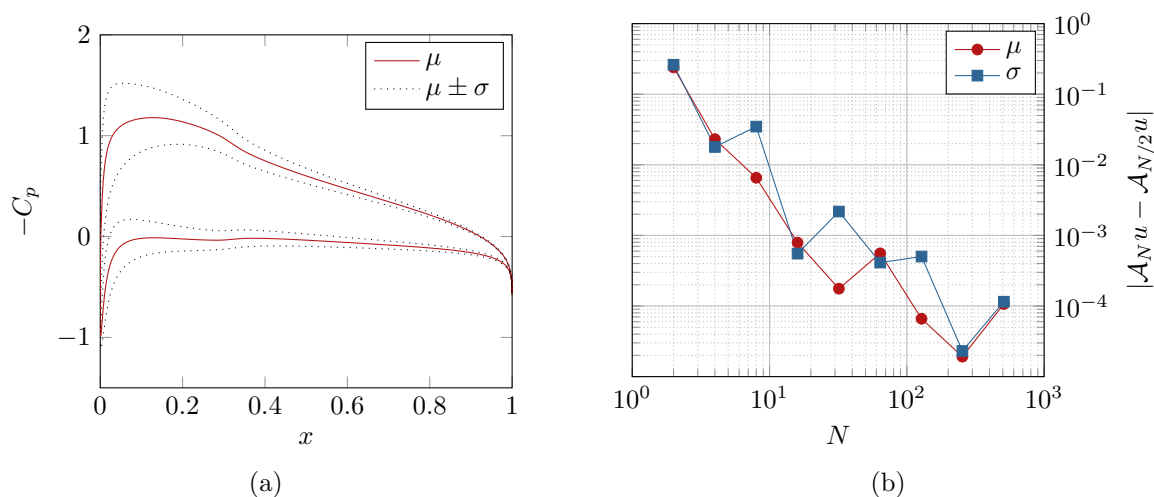
Figure 11. *Left: Mean μ and standard deviation σ of the pressure coefficient (C_p) of the airfoil determined using the finest quadrature rule. Right: Convergence of the mean μ and standard deviation σ of the lift coefficient (C_l) by calculating consecutive differences.*

(a) Mean

(b) Standard deviation

(c) Skewness

(d) Kurtosis
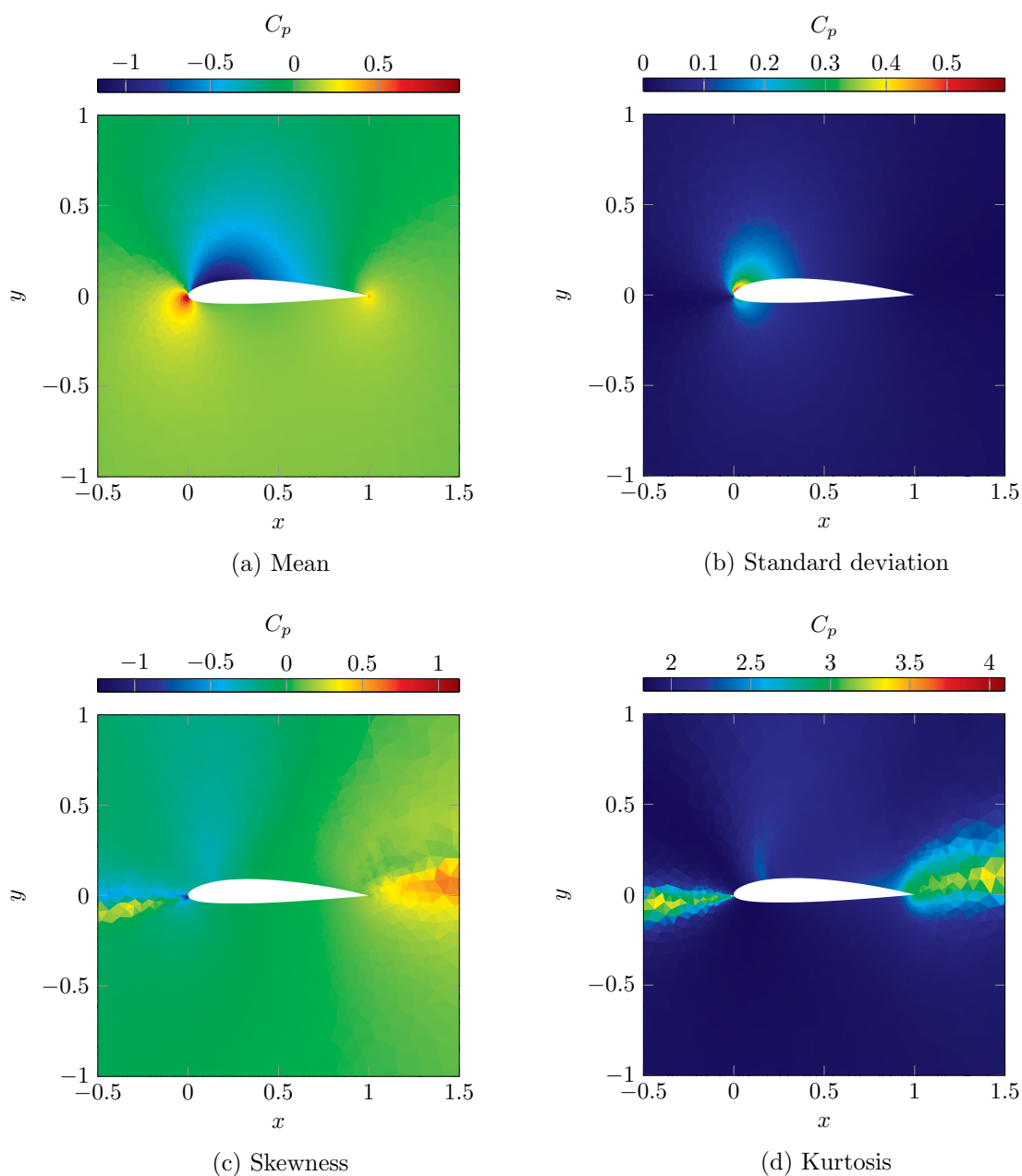
Figure 12. *The first four centralized moments of the pressure coefficient around the airfoil.*

The largest uncertainty of the flow is near the leading edge of the airfoil. This is in contrast to higher Mach number flows, for which it has been observed that the region of largest uncertainty occurs near the shock wave [37, 40]. The skewness of the pressure coefficient shows that its distribution is slightly skewed near the stagnation point on the leading edge of the airfoil. The skewness changes sign at the mean angle of attack. Moreover in the wake of the

airfoil the distribution has positive skewness, which means that outliers of the distribution will more likely be larger than smaller compared to the average pressure coefficient. The kurtosis demonstrates that in many regions of the flow the distribution has much less mass in the tails than a Gaussian (and therefore more unlikely produces outliers). However, again near the leading edge and trailing edge the distribution differs and the tails of the distributions are significantly more influential. We cannot conclude that these locations are the regions of highest uncertainty, as the standard deviation (which is the scaling factor of both the skewness and the kurtosis) is very small in these regions. It is merely a sign that the uncertain behavior of the flow cannot be fully captured by a Gaussian distribution.

5. Conclusion. In this article, a novel nested quadrature rule is proposed which is constructed by solely using samples from a distribution. As the weights of the rule are positive, high order convergence is obtained for sufficiently smooth functions. The algorithm to construct the quadrature rule ensures positive weights, high degree, and nesting regardless of the sample set. The quadrature rules are very suitable for the purpose of nonintrusive uncertainty propagation, because positive weights ensure numerical stability and nesting allows for refinements that reuse computationally expensive model evaluations.

The results from integrating Genz test functions demonstrate that the convergence rate of the quadrature rule is similar to that of the Smolyak sparse grid approach, if the underlying sample distribution is uncorrelated and defined on a hypercube. The real advantage of the proposed quadrature rule appears when this is not the case: for correlated distribution on nonhypercube domains our method still converges at a rate similar to the uncorrelated case, while a sparse grid quadrature rule hardly converges at all. Similar to the existing quadrature rules, the convergence depends on the specific properties of the integrand, in particular on its smoothness.

To demonstrate the applicability to practical test cases, the quadrature rule is used to determine the statistical moments of an airfoil flow with both independent and dependent input distributions. The results demonstrate the advantages of the quadrature rule: nesting can be used for easy refinements, positive weights ensure stability and positive approximations of positive quantities (such as the variance), dependency is naturally taken into account, and the accuracy of the rule yields high convergence rates.

The proposed algorithms provide a framework for the construction of quadrature rules that shows much potential for further extensions. For example, tailoring the basis to the integrand can yield an adaptive quadrature rule without deteriorating the accuracy of the rule as a whole. As the rule is solely based on sample sets, no stringent assumptions are necessary to apply the quadrature rule in such a different setting.

REFERENCES

[1] C. Bayer and J. Teichmann, *The proof of Tchakaloff's theorem*, Proc. Amer. Math. Soc., 134 (2006), pp. 3035–3041, https://doi.org/10.1090/s0002-9939-06-08249-9.
[2] H. Brass and K. Petras, *Quadrature Theory*, Math. Surveys Monographs 178, AMS, Providence, RI, 2011, https://doi.org/10.1090/surv/178.
[3] R. E. Caflisch, *Monte Carlo and quasi-Monte Carlo methods*, Auta Numer., 7 (1998), pp. 1–49, https://doi.org/10.1017/s0962492900002804.

[4] S. L. Cotter, M. Dashti, and A. M. Stuart, *Approximation of Bayesian inverse problems for PDEs*, SIAM J. Numer. Anal., 48 (2010), pp. 322–345, https://doi.org/10.1137/090770734.

[5] P. J. Davis, *A construction of nonnegative approximate quadratures*, Math. Comp., 21 (1967), pp. 578–582, https://doi.org/10.1090/s0025-5718-1967-0222534-4.

[6] L. Devroye, *Non-Uniform Random Variate Generation*, Springer-Verlag, New York, 1986, https://doi.org/10.1007/978-1-4613-8643-8.

[7] T. D. Economon, F. Palacios, S. R. Copeland, T. W. Lukaczyk, and J. J. Alonso, *SU2: An open-source suite for multiphysics simulation and design*, AIAA J., 54 (2016), pp. 828–846, https://doi.org/10.2514/1.j053813.

[8] A. Eggels, D. Crommelin, and J. Witteveen, *Clustering-based collocation for uncertainty propagation with multivariate dependent inputs*, Int. J. Uncertain. Quantif., 8 (2018), pp. 43–59, https://doi.org/10.1615/int.j.uncertaintyquantification.2018020215.

[9] J. Feinberg, V. G. Eck, and H. P. Langtangen, *Multivariate polynomial chaos expansions with dependent variables*, SIAM J. Sci. Comput., 40 (2018), pp. A199–A223, https://doi.org/10.1137/15m1020447.

[10] I. M. Franck and P. S. Koutsourelakis, *Sparse variational Bayesian approximations for nonlinear inverse problems: Applications in nonlinear elastography*, Comput. Methods Appl. Mech. Engrg., 299 (2016), pp. 215–244, https://doi.org/10.1016/j.cma.2015.10.015.

[11] A. Genz, *Testing multidimensional integration routines*, in Proceedings of the International Conference on Tools, Methods and Languages for Scientific and Engineering Computation, Elsevier North-Holland, Amsterdam, 1984, pp. 81–94.

[12] C. Geuzaine and J.-F. Remacle, *Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities*, Internat. J. Numer. Methods Engrg., 79 (2009), pp. 1309–1331, https://doi.org/10.1002/nme.2579.

[13] W. K. Hastings, *Monte Carlo sampling methods using Markov chains and their applications*, Biometrika, 57 (1970), pp. 97–109, https://doi.org/10.1093/biomet/57.1.97.

[14] D. Jackson, *Theory of Approximation*, Amer. Math. Soc. Colloq. Publ., AMS, Providence, RI, 1982.

[15] E. N. Jacobs, K. E. Ward, and R. M. Pinkerton, *The Characteristics of 78 Related Airfoil Sections from Tests in the Variable-Density Wind Tunnel*, Tech. report 460, NASA, 1933.

[16] J. D. Jakeman and A. Narayan, *Generation and application of multivariate polynomial quadrature rules*, Comput. Methods Appl. Mech. Engrg., 338 (2018), pp. 134–161, https://doi.org/10.1016/j.cma.2018.04.009.

[17] V. Keshavarzzadeh, R. M. Kirby, and A. Narayan, *Numerical integration in multiple dimensions with designed quadrature*, SIAM J. Sci. Comput., 40 (2018), pp. A2033–A2061, https://doi.org/10.1137/17m1137875.

[18] O. P. Le Maître and O. M. Knio, *Spectral methods for uncertainty quantification*, Sci. Comput., Springer, New York, 2010, https://doi.org/10.1007/978-90-481-3520-2.

[19] F. Leja, *Sur certaines suites liées aux ensembles plans et leur application à la représentation conforme*, Ann. Polon. Math., 4 (1957), pp. 8–13, http://eudml.org/doc/208291.

[20] D. Liu, A. Litvinenko, C. Schillings, and V. Schulz, *Quantification of airfoil geometry-induced aerodynamic uncertainties: Comparison of approaches*, SIAM/ASA J. Uncertain. Quantif., 5 (2017), pp. 334–352, https://doi.org/10.1137/15m1050239.

[21] G. J. A. Loeven and H. Bijl, *Airfoil analysis with uncertain geometry using the probabilistic collocation method*, in Proceedings of the 49th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, American Institute of Aeronautics and Astronautics, 2008, https://doi.org/10.2514/6.2008-2070.

[22] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, *Equation of state calculations by fast computing machines*, J. Chem. Phys., 21 (1953), pp. 1087–1092, https://doi.org/10.1063/1.1699114.

[23] H. N. Najm, *Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics*, Ann. Rev. Fluid Mech., 41 (2009), pp. 35–52, https://doi.org/10.1146/annurev.fluid.010908.165248.

[24] A. Narayan and J. D. Jakeman, *Adaptive Leja sparse grid constructions for stochastic collocation and high-dimensional approximation*, SIAM J. Sci. Comput., 36 (2014), pp. A2952–A2983, https://doi.org/10.1137/140966368.

[25] M. Navarro, J. Witteveen, and J. Blom, *Stochastic collocation for correlated inputs*, in Proceedings of the 1st ECCOMAS Thematic Conference on Uncertainty Quantification in Computational Sciences and Engineering, M. Papadrakis, V. Papadopoulos, and G. Stefanou, eds., 2015, pp. 218–236.

[26] E. Novak and K. Ritter, *Simple cubature formulas with high polynomial exactness*, Const. Approx., 15 (1999), p. 499, https://doi.org/10.1007/s003659900119.

[27] S. Oladyshkin and W. Nowak, *Data-driven uncertainty quantification using the arbitrary polynomial chaos expansion*, Reliability Engineering & System Safety, 106 (2012), pp. 179–190, https://doi.org/10.1016/j.ress.2012.05.002.

[28] F. Palacios, T. D. Economon, A. Aranake, S. R. Copeland, A. K. Lonkar, T. W. Lukaczyk, D. E. Manosalvas, K. R. Naik, S. Padron, B. Tracey, A. Variyar, and J. J. Alonso, *Stanford University Unstructured (SU2): Analysis and design technology for turbulent flows*, in Proceedings of the 52nd Aerospace Sciences Meeting, American Institute of Aeronautics and Astronautics, 2014, https://doi.org/10.2514/6.2014-0243.

[29] F. Piazzon, A. Sommariva, and M. Vianello, *Caratheodory–Tchakaloff subsampling*, Dolomites Res. Notes Approx., 10 (2017), pp. 5–14, https://doi.org/10.14658/pupj-drna-2017-1-2.

[30] E. K. Ryu and S. P. Boyd, *Extensions of Gauss quadrature via linear programming*, Found. Comput. Math., 15 (2014), pp. 953–971, https://doi.org/10.1007/s10208-014-9197-9.

[31] C. Schillings and C. Schwab, *Sparse, adaptive Smolyak quadratures for Bayesian inverse problems*, Inverse Problems, 29 (2013), 065011, https://doi.org/10.1088/0266-5611/29/6/065011.

[32] P. Seshadri, A. Narayan, and S. Mahadevan, *Effectively subsampled quadratures for least squares polynomial approximations*, SIAM/ASA J. Uncertain. Quantif., 5 (2017), pp. 1003–1023, https://doi.org/10.1137/16m1057668.

[33] Y. Shin and D. Xiu, *A randomized algorithm for multivariate function approximation*, SIAM J. Sci. Comput., 39 (2017), pp. A983–A1002, https://doi.org/10.1137/16m1075193.

[34] M. Sinsbeck and W. Nowak, *An optimal sampling rule for nonintrusive polynomial chaos expansions of expensive models*, Internat. J. Uncertain. Quantif., 5 (2015), pp. 275–295, https://doi.org/10.1615/int.j.uncertaintyquantification.2015008446.

[35] S. Smolyak, *Quadrature and interpolation formulas for tensor products of certain classes of functions*, Soviet Math., Dokl., 4 (1963), pp. 240–243.

[36] L. M. M. van den Bos, B. Koren, and R. P. Dwight, *Non-intrusive uncertainty quantification using reduced cubature rules*, J. Comput. Phys., 332 (2017), pp. 418–445, https://doi.org/10.1016/j.jcp.2016.12.011.

[37] L. M. M. van den Bos, B. Sanderse, W. A. A. M. Bierbooms, and G. J. W. van Bussel, *Bayesian model calibration with interpolating polynomials based on adaptively weighted Leja nodes*, Commun. Comput. Phys., 27 (2020), pp. 33–69, https://doi.org/10.4208/cicp.oa-2018-0218.

[38] G. A. Watson, *Approximation Theory and Numerical Methods*, John Wiley & Sons, New York, 1980.

[39] M. W. Wilson, *A general algorithm for nonnegative quadrature formulas*, Math. Comp., 23 (1969), pp. 253–253, https://doi.org/10.1090/s0025-5718-1969-0242374-1.

[40] J. A. S. Witteveen, A. Doostan, R. Pecnik, and G. Iaccarino, *Uncertainty Quantification of the Transonic Flow Around the RAE 2822 Airfoil*, Center for Turbulence Research, Stanford University, 2009.

[41] K. Wu, Y. Shin, and D. Xiu, *A randomized tensor quadrature method for high dimensional polynomial approximation*, SIAM J. Sci. Comput., 39 (2017), pp. A1811–A1833, https://doi.org/10.1137/16m1081695.

[42] D. Xiu, *Numerical Methods for Stochastic Computations*, Princeton University Press, Princeton, NJ, 2010.

[43] G. Zhang, D. Lu, M. Ye, M. Gunzburger, and C. Webster, *An adaptive sparse-grid high-order stochastic collocation method for Bayesian inference in groundwater reactive transport modeling*, Water Resources Research, 49 (2013), pp. 6871–6892, https://doi.org/10.1002/wrcr.20467.