BIT Numer Math (2009) 49: 55–73
DOI 10.1007/s10543-009-0210-7

New cubature formulae and hyperinterpolation in three variables

Stefano De Marchi · Marco Vianello · Yuan Xu

Received: 22 May 2008 / Accepted: 13 January 2009 / Published online: 30 January 2009

Abstract A new algebraic cubature formula of degree $2n+1$ for the product Chebyshev measure in the d-cube with $\approx n^d/2^{d-1}$ nodes is established. The new formula is then applied to polynomial hyperinterpolation of degree n in three variables, in which coefficients of the product Chebyshev orthonormal basis are computed by a fast algorithm based on the 3-dimensional FFT. Moreover, integration of the hyperinterpolant provides a new Clenshaw-Curtis type cubature formula in the 3-cube.

Keywords Cubature · Polynomial hyperinterpolation · Fast algorithms · Clenshaw-Curtis type cubature formula

Mathematics Subject Classification (2000) 65D32 · 65D05

1 Introduction

A cubature formula with high accuracy is an important tool for numerical computation and has various applications. One of the applications is to construct polynomial

Communicated by T. Lyche.

Work supported by the National Science Foundation under Grant DMS-0604056, by the "ex-60%" funds of the Universities of Padova and Verona, and by the INdAM-GNCS.

S. De Marchi (✉)
Department of Computer Science, University of Verona, S.da Le Grazie 15, Verona, Italy
e-mail: stefano.demarchi@univr.it

M. Vianello
Department of Pure and Applied Mathematics, University of Padua, Via Trieste 63, Padua, Italy
e-mail: marcov@math.unipd.it

Y. Xu
Department of Mathematics, University of Oregon, 11B Deady Hall, Eugene, OR 97403-1222, USA
e-mail: yuan@uoregon.edu

hyperinterpolation, introduced by Sloan [18], which is an approximation process constructed by applying the cubature formula on the expansion coefficients of the orthogonal projection operator.

A cubature formula of degree $2n+1$ with N nodes with respect to a measure $d\mu$ supported on a set Ω takes the form

$$\int_\Omega p(x)\,d\mu = \sum_{\xi\in X_n} w_\xi\, p(\xi) \quad \text{for all } p\in \Pi^d_{2n+1}(\Omega), \tag{1.1}$$

where $\{w_\xi\}$, called weights, are (positive) numbers, X_n is a set of points, called nodes,

$$\xi := (\xi_1, \xi_2, \ldots, \xi_d) \in X_n \subset \Omega \tag{1.2}$$

with $\mathrm{card}(X_n) = N$, and $\Pi^d_m(\Omega)$ denotes the subspace of d-variate polynomials of total degree $\le m$ restricted to Ω. For a cubature formula of degree $2n+1$ to exist, it is necessary that

$$N := \mathrm{card}(X_n) \ge \dim(\Pi^d_n(\Omega)) = \binom{n+d}{d} = \frac{n^d}{d!}(1+o(1)). \tag{1.3}$$

There are improved lower bounds of the same order in terms of n. A challenging problem is to construct cubature formulae with fewer nodes, that is, with the number of nodes N close to the lower bound.

In this paper we consider the case that the measure is given by the product Chebyshev weight function

$$d\mu = W_d(x)\,dx, \quad W_d(x) := \frac{1}{\pi^d}\prod_{i=1}^{d}\frac{1}{\sqrt{1-x_i^2}} \tag{1.4}$$

on the cube $\Omega := [-1,1]^d$. For $d=1$, the Gaussian quadrature formula of degree $2n+1$ needs merely $N=n+1$ points. Our main result is a new family of cubature formulae that uses $N \approx n^d/2^{d-1}$ nodes. For $d=2$ these formulae are known to have minimal number of nodes. For $d\ge 3$ they are still far from the lower bound, but they appear to be the best ones that are known at this moment. We refer to Sect. 2 for further discussions. We present numerical tests on these cubature formulae in three variables and also apply them to constructing the corresponding polynomial hyperinterpolation operator in three variables.

For every function $f\in C(\Omega)$ the μ-orthogonal projection of f on $\Pi^d_n(\Omega)$ is

$$\mathcal{S}_n f(x) = \sum_{|\alpha|\le n} a_\alpha p_\alpha(x), \quad a_\alpha := \int_\Omega f(x) p_\alpha(x)\,d\mu, \tag{1.5}$$

where $x=(x_1,x_2,\ldots,x_d)$ is a d-dimensional point, α is a d-index of length $|\alpha|$

$$\alpha = (\alpha_1,\ldots,\alpha_d)\in\mathbb{N}^d_0, \qquad |\alpha| := \alpha_1+\cdots+\alpha_d, \tag{1.6}$$

and the set of polynomials $\{p_\alpha, 0 \le |\alpha| \le n\}$ is any μ-orthonormal basis of $\Pi_n^d(\Omega)$ with p_α of total degree $|\alpha|$ (concerning the theory of multivariate orthogonal polynomials, we refer the reader to the monograph [9]). Clearly, $\mathcal{S}_n p = p$ for every $p \in \Pi_n^d(\Omega)$. Given a cubature formula (1.1) of degree $\le 2n$, we obtain from (1.5) the polynomial approximation of degree n by the *discretized expansion coefficients* $\{c_\alpha\}$

$$f(x) \approx \mathcal{L}_n f(x) := \sum_{|\alpha| \le n} c_\alpha p_\alpha(x), \quad c_\alpha := \sum_{\xi \in X_n} w_\xi f(\xi) p_\alpha(\xi), \tag{1.7}$$

where $c_\alpha = a_\alpha$ and thus $\mathcal{L}_n p = \mathcal{S}_n p = p$ for every $p \in \Pi_n^d(\Omega)$. This is the hyperinterpolation operator. It satisfies the basic estimate: for every $f \in C(\Omega)$,

$$\|f - \mathcal{L}_n f\|_{L^2_{d\mu}(\Omega)} \le 2\sqrt{\mu(\Omega)}\, E_n(f) \to 0, \quad n \to \infty, \tag{1.8}$$

where $E_n(f) := \inf\{\|f - p\|_\infty, p \in \Pi_n^d(\Omega)\}$, so that it converges in mean. The convergence rate can be estimated by a multivariate version of Jackson theorem (see, for example, [16]), which shows that $E_n(f) = \mathcal{O}(n^{-p})$ for $f \in C^p(\Omega)$, $p \in \mathbb{R}^+$. It becomes an effective approximation tool in the uniform norm when its operator norm (the so-called Lebesgue constant) grows slowly (cf. [5, 11, 17, 19]). The hyperinterpolation has been used effectively in several cases: originally for the sphere [17, 19], and more recently for the square [4, 5], the disk [11], and the cube [6]. We will use our new cubature formulae to construct a hyperinterpolation operator of three variables for the Chebyshev weight function on the cube. We show that the computation can be carried out efficiently using the 3-dimensional FFT and that the algorithm can be completely vectorized. We will also present numerical results on hyperinterpolation of several test functions.

The paper is organized as follows. In Sect. 2 we construct new cubature formulae and report results of numerical tests, where comparisons are made with tensor-product Gauss-Chebyshev formulae. Hyperinterpolation in three variables is considered in Sect. 3, where we show how to compute it effectively and report the results of numerical tests. Finally in Sect. 4, we obtain a new (nontensorial) Clenshaw-Curtis type formula in the cube by integrating the hyperinterpolant in Sect. 3 and show that it has a clear superiority over tensorial Clenshaw-Curtis and Gauss-Legendre cubature on nonentire test integrands, a phenomenon known for 1-dimensional and 2-dimensional Clenshaw-Curtis formulae (see [20, 21]).

2 Algebraic cubature for the d-dimensional Chebyshev measure

We consider cubature formula for the product Chebyshev weight function (1.4), which is normalized so that its integral over $\Omega = [-1, 1]^d$ is 1. For $d = 1$, we write $w(x) = W_1(x)$.

In what follows we use for conveninece the notation $\Pi_n = \Pi_n^1([-1, 1])$. The Gaussian quadrature formula for w takes the form

$$\int_{-1}^{1} f(x) w(x) dx = \frac{1}{n} \sum_{k=1}^{n} f\left(\cos \frac{(2k-1)\pi}{2n}\right), \quad \forall f \in \Pi_{2n-1}. \tag{2.1}$$

For $d = 2$, a cubature formula of degree $2n - 1$ needs at least (cf. [14])

$$N^* = \dim(\Pi_{n-1}^2(\Omega)) + \left\lfloor \frac{n}{2} \right\rfloor = \frac{n(n+1)}{2} + \left\lfloor \frac{n}{2} \right\rfloor \tag{2.2}$$

many nodes. Cubature formulae that attain this lower bound can be constructed for the product Chebyshev weight $W_2(x)$ (see [15, 24] and the references therein) by studying common zeros of associated orthogonal polynomials. In [1], these cubature rules were derived by an elementary method which depends on a factorization of the Gauss-Lobatto quadrature into two sums, over *even* indices and *odd* indices, respectively. This factorization method was also used for $d > 2$ in [1] and yields a cubature formula of degree $2n - 1$ for W_d with roughly $n^d/2^{d/2}$ many nodes.

A close inspection of the factorization method shows that it actually allows us to derive cubature formulae of degree $2n - 1$ for W_d with roughly $2(n/2)^d$ nodes. This number of nodes is substantially less than n^d of the product cubature formula or $n^d/2^{d/2}$ of the formulae in [1], although it is likely still far from optimal as seen from (1.3).

We start with the Gauss-Lobatto formula for w on $[-1, 1]$. It takes the form

$$\int_{-1}^{1} f(x)w(x)dx = \frac{1}{n}\left(\frac{1}{2}f(-1) + \sum_{j=1}^{n-1} f\left(\cos\frac{j\pi}{n}\right) + \frac{1}{2}f(1)\right) := I_n f, \tag{2.3}$$

which again holds for all $f \in \Pi_{2n-1}$. We proceed to factor this rule into two terms. The factorization depends on whether n is even or n is odd. Define

$$\begin{aligned} n = 2m: \quad & I_n^E f := \frac{1}{n}\left(\frac{1}{2}f(-1) + \sum_{j=1}^{m-1} f\left(\cos\frac{2j\pi}{n}\right) + \frac{1}{2}f(1)\right), \\ & I_n^O f := \frac{1}{n}\sum_{j=1}^{m} f\left(\cos\frac{(2j-1)\pi}{n}\right) \end{aligned} \tag{2.4}$$

and define

$$\begin{aligned} n = 2m-1: \quad & I_n^E f := \frac{1}{n}\left(\sum_{j=1}^{m-1} f\left(\cos\frac{2j\pi}{n}\right) + \frac{1}{2}f(1)\right), \\ & I_n^O f := \frac{1}{n}\left(\frac{1}{2}f(-1) + \sum_{j=1}^{m-1} f\left(\cos\frac{(2j-1)\pi}{n}\right)\right), \end{aligned} \tag{2.5}$$

where we use the superscripts E and O to signify that the sum is taken over even indices or odd indices, respectively. Evidently, the quadrature (2.3) becomes

$$\int_{-1}^{1} f(x)w(x)dx = I_n^E f + I_n^O f, \quad \forall f \in \Pi_{2n-1},$$

by definition.

The Chebyshev polynomials, T_n, are orthogonal with respect to w on $[-1, 1]$,

$$T_n(t) := \cos n\theta, \quad t = \cos\theta.$$

The following elementary lemma plays a key role in constructing cubature formulae on $[-1, 1]^d$.

Lemma 2.1 *For $n \geq 0$ and $k \in \mathbb{Z}$,*

$$I_n^E T_k = \begin{cases} 0, & k \neq 0 \bmod n, \\ \frac{1}{2}, & k = 0 \bmod n, \end{cases} \quad \text{and} \quad I_n^O T_k = \begin{cases} 0, & k \neq 0 \bmod n, \\ \frac{1}{2}, & k = 0, 2n, 4n, \ldots, \\ -\frac{1}{2}, & k = n, 3n, \ldots. \end{cases}$$

Proof The proof follows from elementary trigonometric identities. For example, for $n = 2m$, an elementary calculation shows that

$$I_n^O T_k = \frac{1}{n}\sum_{j=1}^{m} \cos k\frac{(2j-1)\pi}{2m} = \frac{\sin k\pi}{4m \sin\frac{k\pi}{2m}} = \frac{\sin k\pi}{2n \sin\frac{k\pi}{n}}$$

from which $I_n^O T_k = 0$ for $k \neq 0 \mod n$ follows immediately. The case when k is a multiple of n follows from the first equal sign of the above equation without summing it up. Similarly,

$$I_n^E T_k = \frac{1}{n}\left(\frac{1}{2}\cos k\pi + \sum_{j=1}^{m-1} \cos k\frac{j\pi}{m} + \frac{1}{2}\right) = \frac{\sin k\pi \cos\frac{k\pi}{n}}{2n \sin\frac{k\pi}{n}},$$

from which the stated result follows. The proof for $n = 2m - 1$ is similar and is omitted for brevity. □

Let $\sigma \in \{E, O\}^d$, that is,

$$\sigma = (\sigma_1, \ldots, \sigma_d) \quad \text{with } \sigma_i = E \text{ or } \sigma_i = O.$$

For a function $f : \mathbb{R}^d \mapsto \mathbb{R}$, we define the sum

$$I_n^{\sigma_1} \cdots I_n^{\sigma_d} f$$

as a d-fold multiple sum in which I^{σ_k} is applied to the k-th variable of f. Let us define

$$\bar{\sigma}_i = \begin{cases} E & \sigma_i = O, \\ O & \sigma_i = E. \end{cases} \tag{2.6}$$

For each $\sigma \in \{E, O\}^d$, we then define

$$I_{n,d}^{\sigma} f := I_n^{\sigma_1} \cdots I_n^{\sigma_d} f + I_n^{\bar{\sigma}_1} \cdots I_n^{\bar{\sigma}_d} f.$$

Since the sum introduces a symmetry among $\sigma \in \{E, O\}^d$, there are 2^{d-1} distinct $I_{n,d}^{\sigma} f$ sums.

Theorem 2.2 *For $d \geq 1$ and each $\sigma \in \{E, O\}^d$, the cubature formula*

$$\int_{[-1,1]^d} f(x)W_d(x)dx = 2^{d-1} I^{\sigma}_{n,d} f \tag{2.7}$$

is exact for $f \in \Pi^d_{2n-1}(\Omega)$ and its number of nodes, N, satisfies

$$N = 2\left(\left\lfloor \frac{n}{2} \right\rfloor\right)^d (1 + o(n^{-1})).$$

Proof For $k = (k_1, \ldots, k_d) \in \mathbb{N}_0^d$ let $T_k(x) := T_{k_1}(x_1) \cdots T_{k_d}(x_d)$, which is a polynomial of total degree $|k| := k_1 + \cdots + k_d$. It is well known that $\{T_k : |k| \leq n\}$ is an orthogonal basis of $\Pi^d_n(\Omega)$ (cf. [9]). Thus, it suffices to establish (2.7) for $f \in \{T_k : |k| \leq 2n-1\}$. In this case we have

$$\int_{[-1,1]^d} T_k(x)W_d(x)dx = 2^{d-1}[I^{\sigma_1}_n T_{k_1} \cdots I^{\sigma_d}_n T_{k_d} + I^{\bar{\sigma}_1}_n T_{k_1} \cdots I^{\bar{\sigma}_d}_n T_{k_d}].$$

Since $T_{(0,\ldots,0)}(x) \equiv 1$, we can write

$$\int_{[-1,1]^d} T_k(x)W_d(x)dx = \int_{[-1,1]^d} T_k(x)T_{(0,\ldots,0)}(x)W_d(x)dx = \delta_{k,(0,\ldots,0)},$$

where the last equality is due to the orthogonality of $\{T_k\}$. From the definition of I^E_n and I^O_n, it is evident that $I^E_n 1 = I^O_n 1 = 1/2$. Hence, for $k = (0, \ldots, 0)$, the right hand side is equal to $2^{d-1}(2^{-d} + 2^{-d}) = 1$, verifying the equation for $k = (0, \ldots, 0)$.

Assume now $0 < |k| \leq 2n-1$. If one of $k_i \neq 0 \mod n$, then $I^{\sigma}_{n,d} T_k = 0$ by Lemma 2.1. We are left with the case that $k_i = 0 \mod n$ for all i. Since $|k| \leq 2n-1$, there can be at most one $k_i = n$. Furthermore, $|k| > 0$ shows that there is exactly one $k_i = n$. Thus the right hand side becomes $I^{\sigma_i}_n T_n + I^{\bar{\sigma}_i}_n T_n = I^E_n T_n + I^O_n T_n$, which is zero as $I^E_n T_n = 1/2$ and $I^O_n T_n = -1/2$ according to the Lemma 2.1. The last claim of the statement follows by the observation that each term in formulas (2.4) and (2.5) uses $\lfloor \frac{n}{2} \rfloor + \mathcal{O}(1)$ points. □

For the case of $d = 2$, Theorem 2.2 contains two distinct cubature formulae for $\sigma = (E, E), (E, O)$, respectively, whose number of nodes are either equal to N^* in (2.2) or $N^* + 1$, those are the ones that have appeared in [15, 24], and later in [1], as mentioned earlier. For $d = 3$, there are 4 distinct formulae for σ, that is

$$(E, E, E), (E, E, O), (E, O, E), (O, E, E).$$

For $n = 2m$, the number of nodes is

$$N = \frac{(n+1)^3 + (n+1)}{4}$$

for $\sigma = (E, E, E)$ and

$$N = \frac{(n+1)^3 - (n+1)}{4}$$

for $\sigma = (E, E, O), (E, O, E), (O, E, E)$, respectively.

In order to demonstrate the effectiveness of the new cubature formula, we present in Figs. 1–2 numerical results of (2.7) with $\sigma = (E, E, E)$ on the integrals of six text functions with respect to the product Chebyshev measure on the 3-cube. The first three functions are analytic entire (a polynomial, an exponential and a Gaussian), whereas the other three are less smooth: one analytic but not entire (a 3-dimensional version of the Runge test function), one C^∞ nonanalytic, and one C^2. These functions are analogues of test functions for algebraic cubature in dimension 1 and 2, see [20, 21]. We compare them with two natural choices for cubature on a tensor product domain: the tensor-product Gauss-Chebyshev and Gauss-Chebyshev-Lobatto formulae. The results, obtained with Matlab (cf. [10]), demonstrate the superiority of the new formula in all cases, especially for the less smooth functions, in terms of number of function evaluations. It should be pointed out that, however, the superiority for the less smooth functions arises for *even n* (a sort of parity phenomenon). Other numerical tests (not reported for brevity) have shown that the cubature formula has the same behavior for $\sigma = (E, E, O), (E, O, E), (O, E, E)$.

3 Implementing hyperinterpolation in the 3-cube

A natural question associated with cubature formulae is polynomial interpolation. Let X_{n-1} denote the set of the nodes of the cubature formula (2.7). The interpolation problem looks for a polynomial subspace, $\mathcal{S}$, of the lowest degree such that

$$P(x) = f(x), \quad x \in X_{n-1}, \ \forall f \in C(\mathbb{R}^d)$$

has a unique solution in $\mathcal{S}$. In the case of $d = 2$, this problem is completely solved in [24], where $\mathcal{S}$ is a subspace of $\Pi_n^2(\Omega)$ which includes $\Pi_{n-1}^2(\Omega)$, and compact formulae of the fundamental interpolation polynomials are also given there. For $d > 2$, however, the problem is much harder, since the number of nodes of our cubature is far from $\dim(\Pi_n^d(\Omega))$. For example, if $d = 3$, then $\dim(\Pi_{n-1}^d(\Omega)) = n(n+1)(n+2)/6 \approx n^3/6$, whereas our cubature requires $\approx n^3/4$ nodes. The problem essentially comes down to study the polynomial ideal that has X_{n-1} as its variety (see [25]).

A simpler approach to polynomial approximation via these new nodes is given by hyperinterpolation, as described in the Introduction. We now use cubature formula (2.7) to construct hyperinterpolation as in (1.7) for the 3-cube $\Omega = [-1, 1]^3$. In this case, $\{p_\alpha\}$ is the product Chebyshev orthonormal basis (cf. [9]), i.e.

$$p_\alpha(x) = \hat{T}_{\alpha_1}(x_1)\hat{T}_{\alpha_2}(x_2)\hat{T}_{\alpha_3}(x_3), \tag{3.1}$$

where $\hat{T}_k(\cdot) = \sqrt{2}\cos(k \arccos(\cdot))$, $k > 0$ and $\hat{T}_0(\cdot) = 1$. Moreover, let

$$C_n = \left\{\cos\frac{k\pi}{n}, k = 0, \dots, n\right\}$$

be the set of $n + 1$ Chebyshev-Lobatto points, and C_n^E, C_n^O its restriction to even and odd indices, respectively. Then,

$$X_n = (C_{n+1}^{\sigma_1} \times C_{n+1}^{\sigma_2} \times C_{n+1}^{\sigma_3}) \cup (C_{n+1}^{\bar{\sigma}_1} \times C_{n+1}^{\bar{\sigma}_2} \times C_{n+1}^{\bar{\sigma}_3}), \tag{3.2}$$

Fig. 1 Relative cubature errors (Chebyshev weight function) versus the number of function evaluations for three test functions

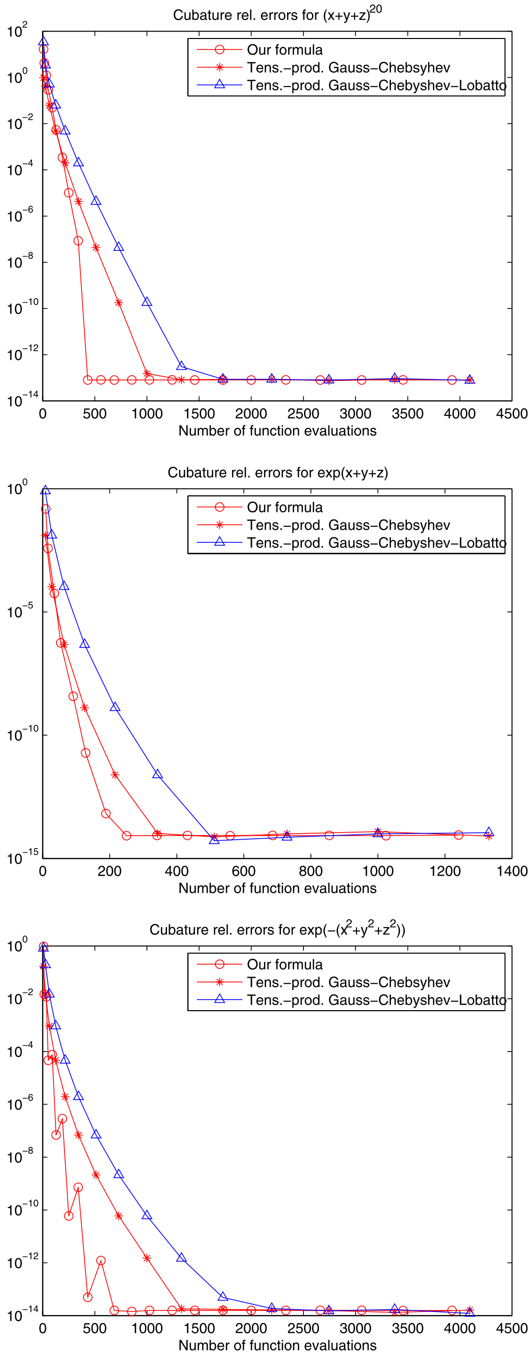
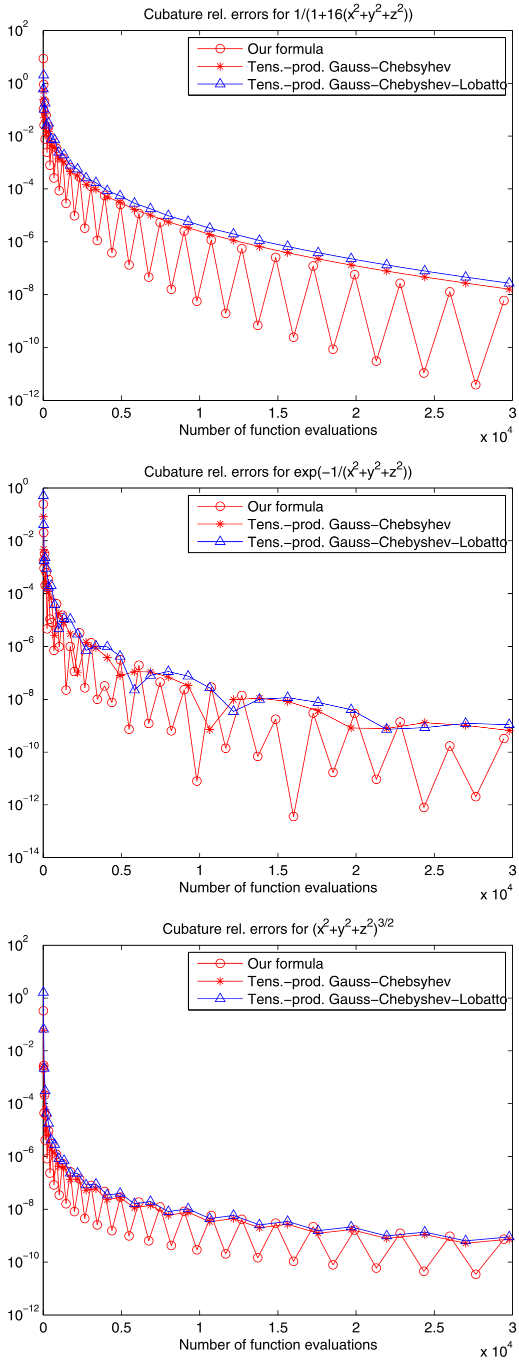


Fig. 2 Relative cubature errors (Chebyshev weight function) versus the number of function evaluations for three test functions



with $(\sigma_1, \sigma_2, \sigma_3) \in \{E, O\}^3$, see (2.6). The weights of the cubature formula (2.7) for $\xi \in X_n$, are

$$w_\xi = \frac{4}{(n+1)^3} \cdot \begin{cases} 1 & \text{if } \xi \text{ is an interior point}, \\ 1/2 & \text{if } \xi \text{ is a face point}, \\ 1/4 & \text{if } \xi \text{ is an edge point}, \\ 1/8 & \text{if } \xi \text{ is a vertex point}. \end{cases} \tag{3.3}$$

Note that, since

$$\dim(\Pi_n^3(\Omega)) = (n+1)(n+2)(n+3)/6 < N = \text{card}(X_n) \approx n^3/4,$$

the polynomial $\mathcal{L}_n f$ in (1.7) is not interpolant.

Now, defining

$$F(\xi) = F(\xi_1, \xi_2, \xi_3) = \begin{cases} w_\xi f(\xi), & \xi \in X_n, \\ \\ 0, & \xi \in (C_{n+1} \times C_{n+1} \times C_{n+1}) \backslash X_n \end{cases} \tag{3.4}$$

we can write

$$\begin{aligned} c_\alpha &= \sum_{\xi \in X_n} w_\xi f(\xi) p_\alpha(\xi) \\ &= \sum_{\xi_1 \in C_{n+1}} \left(\sum_{\xi_2 \in C_{n+1}} \left(\sum_{\xi_3 \in C_{n+1}} F(\xi_1, \xi_2, \xi_3) \hat{T}_{\alpha_1}(\xi_1) \right) \hat{T}_{\alpha_2}(\xi_2) \right) \hat{T}_{\alpha_3}(\xi_3) \\ &= \beta_\alpha \sum_{i=0}^{n+1} \left(\sum_{j=0}^{n+1} \left(\sum_{k=0}^{n+1} F_{ijk} \cos \frac{k \alpha_1 \pi}{n+1} \right) \cos \frac{j \alpha_2 \pi}{n+1} \right) \cos \frac{i \alpha_3 \pi}{n+1}, \end{aligned}$$

where $\alpha = (\alpha_1, \alpha_2, \alpha_3) \in \{0, 1, \ldots, n\}^3$ and

$$\beta_\alpha = \prod_{s=1}^{3} \beta_{\alpha_s}, \quad \beta_{\alpha_s} = \begin{cases} \sqrt{2}, & \alpha_s > 0, \\ 1, & \alpha_s = 0, \end{cases} \quad s = 1, 2, 3. \tag{3.5}$$

This shows that the 3-dimensional coefficients array $\{c_\alpha\}$ is a scaled Discrete Cosine Transform of the 3-dimensional array

$$F_{ijk} = F\left(\cos \frac{i\pi}{n+1}, \cos \frac{j\pi}{n+1}, \cos \frac{k\pi}{n+1}\right), \quad 0 \le i, j, k \le n+1, \tag{3.6}$$

where we eventually pick up only the $(n+1)(n+2)(n+3)/6 \approx n^3/6$ hyperinterpolation coefficients corresponding to $|\alpha| = \alpha_1 + \alpha_2 + \alpha_3 \le n$.

A fast implementation of hyperinterpolation is now feasible (for example in Matlab), via the FFT. Indeed, we have written a Matlab code (see [8]), completely vectorized by several implementation tricks, whose kernel can be summarized as follows:

Algorithm: Fast total degree hyperinterpolation in the 3-cube

(i) construct the hyperinterpolation point set X_n as union of the two subgrids in (3.2);
(ii) compute the cubature weights in (3.3);
(iii) compute the 3-dimensional array $\{F_{ijk}\}$ at the complete grid $C_{n+1} \times C_{n+1} \times C_{n+1}$ by (3.4) (notice that f is evaluated *only* at X_n);
(iv) compute the 3-dimensional array of coefficients $\{c_\alpha\}$ by three nested applications of the 1-dimensional Real(FFT($\cdot$)) operator;
(v) select the coefficients $\{c_\alpha\}$ corresponding to the triples $\alpha = (\alpha_1, \alpha_2, \alpha_3)$ such that $|\alpha| = \alpha_1 + \alpha_2 + \alpha_3 \leq n$.

We recall that there is a simple way to approximate a function in the 3-cube by tensor-product of polynomials of degree n, that is, by a tensor-product discrete Chebyshev series (ultimately a tensor-product hyperinterpolant). Such an approximation uses $(n+1)^3$ function evaluations, and $(n+1)^3$ coefficients. In contrast, let us stress again the following facts on our total-degree hyperinterpolation of degree n in the 3-cube:

Remark

- the number of hyperinterpolation nodes, or function evaluations, is equal to $\text{card}(X_n) \approx n^3/4$;
- the number of hyperinterpolation coefficients is $\dim(\Pi_n^3(\Omega)) \approx n^3/6$.

In order to compare the performances of total-degree and tensor-product hyperinterpolation in the 3-cube, we show, in Figs. 3–4, the hyperinterpolation errors versus the number of nodes (i.e., of function evaluations) on the six test functions already used in Sect. 2, and we choose again $(\sigma_1, \sigma_2, \sigma_3) = (E, E, E)$, see (3.2). The errors are relative to the maximum deviation of the function from its mean and are computed on a uniform control grid. Since the computation of the coefficients via the FFT has roughly the same cost for both kinds of hyperinterpolation, in terms of both CPU time and storage, we have chosen the number of function evaluations as a measure of computational cost for the construction, and the number of coefficients as a measure of the compression capability of the algorithms.

The situation here is in some sense opposite to that of Figs. 1–2. Indeed, total-degree appears superior to tensor-product hyperinterpolation on the smoothest functions, but not on the less smooth ones. As it is natural from the observation above, the behavior of total-degree hyperinterpolation in terms of number of coefficients is better than that in terms of number of nodes (function evaluations).

4 A Clenshaw-Curtis-like formula for the cube

In the recent paper [20], pursuing an idea already present in [18], it has been shown how hyperinterpolation allows us to construct new cubature formulae. Given $h \in$

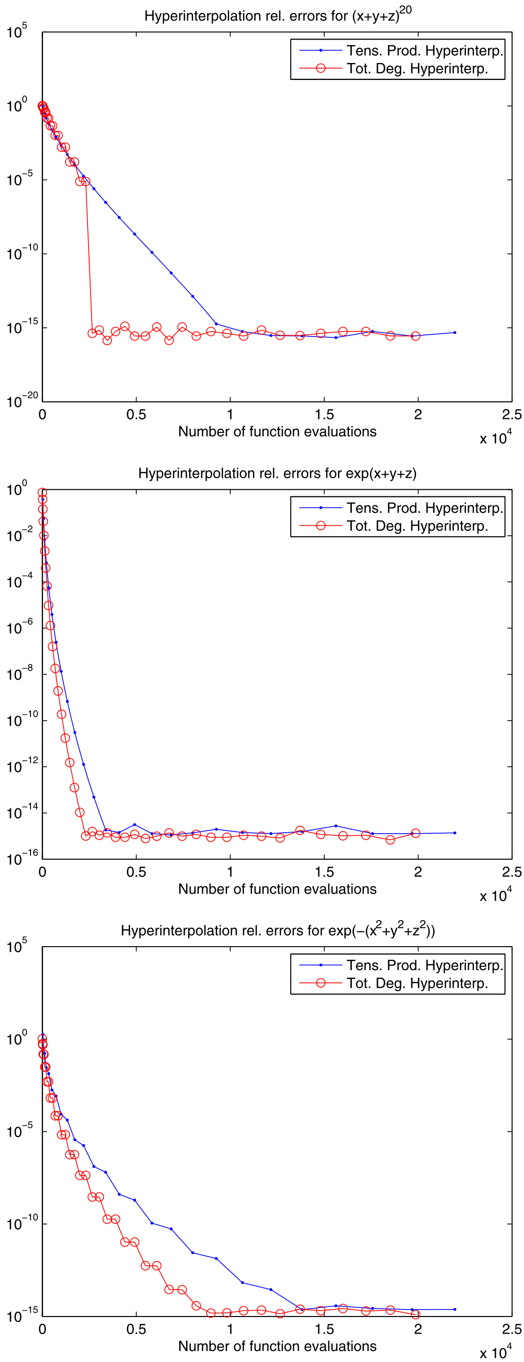


Fig. 3 Hyperinterpolation relative errors versus the number of function evaluations for three entire test functions

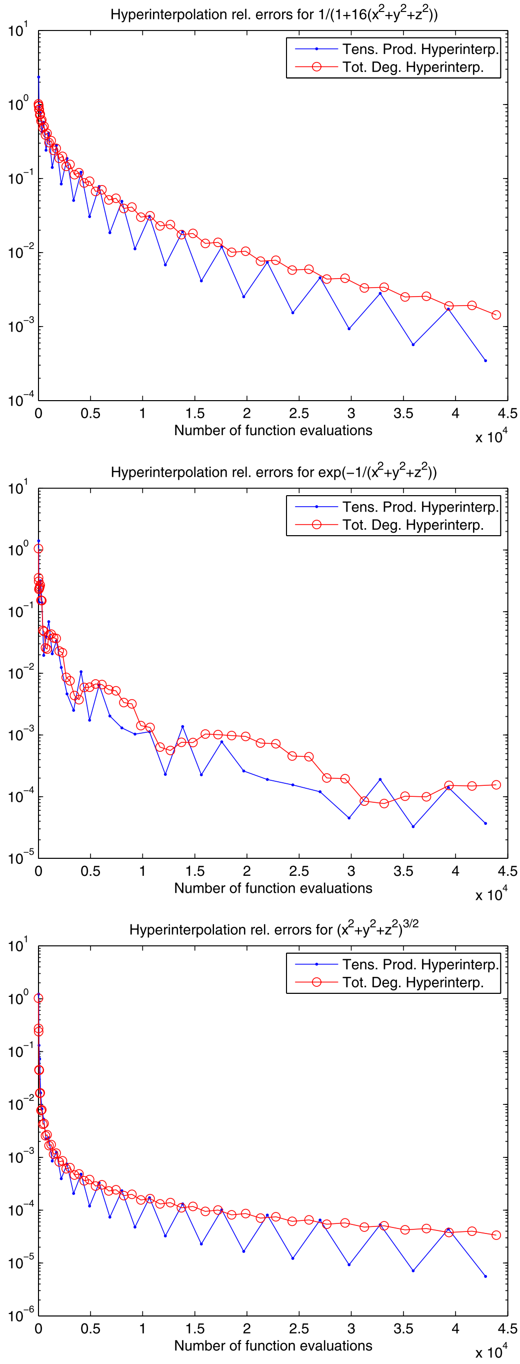


Fig. 4 Hyperinterpolation relative errors versus the number of function evaluations for three nonentire test functions

$L^2_{d\mu}(\Omega)$ and $f \in C(\Omega)$, we can approximate the integral of hf in $d\mu$ as

$$\int_\Omega h(x) f(x)\,d\mu \approx \int_\Omega h(x)\mathcal{L}_n f(x)\,d\mu$$
$$= \sum_{|\alpha|\le n} c_\alpha m_\alpha = \sum_{\xi\in X_n} \lambda_\xi f(\xi), \qquad (4.1)$$

where the generalized "orthogonal moments" $\{m_\alpha\}$ and the cubature weights $\{\lambda_\xi\}$ are defined by

$$m_\alpha := \int_\Omega h(x) p_\alpha(x)\,d\mu, \qquad \lambda_\xi := w_\xi \sum_{|\alpha|\le n} p_\alpha(\xi)\, m_\alpha. \qquad (4.2)$$

Observe that the cubature formula (4.1) is *exact* for every $f \in \Pi^d_n(\Omega)$, and that $\{m_\alpha\}$ are just Fourier coefficients of h with respect to the μ-orthonormal basis $\{p_\alpha\}$.

Concerning stability and convergence of such cubature formulae, the following result has been proved in [20]:

Theorem 4.1 *Let all the assumptions for the construction of the cubature formula* (4.1) *be satisfied, and in particular let $h \in L^2_{d\mu}(\Omega)$. Then the sum of the absolute values of the cubature weights has a finite limit*

$$\lim_{n\to\infty} \sum_{\xi\in X_n} |\lambda_\xi| = \int_\Omega |h(x)|\,d\mu. \qquad (4.3)$$

Notice that (4.3) ensures that the sum of absolute values of the weights is bounded, and thus by recalling that $\mathcal{L}_n$ is a projection operator on $\Pi^d_n(\Omega)$ we obtain the Polya-Steklov type (cf. [12]) convergence estimate

$$\left| \int_\Omega h(x) f(x)\,d\mu - \sum_{\xi\in X_n} \lambda_\xi\ f(\xi) \right| \le \left(\int_\Omega |h(x)|\,d\mu + \sup_n \sum_{\xi\in X_n} |\lambda_\xi| \right) E_n(f), \qquad (4.4)$$

where $E_n(f)$ denotes the error of the best polynomial approximation of degree n to f in the uniform norm.

Now, applying (4.1)–(4.2) in the case

$$d\mu = w(x)\,dx, \quad w \in L^1_{dx}(\Omega), \text{ with } h = \frac{1}{w} \in L^1_{dx}(\Omega), \qquad (4.5)$$

(since then $h^2 = 1/w^2 \in L^1_{d\mu}(\Omega)$) we obtain, via hyperinterpolation, a cubature formula for the standard Lebesgue measure from an algebraic cubature formula for another measure (absolutely continuous with respect to the former). The specialization of this approach to the 1-dimensional Chebyshev measure gives ultimately the popular Clenshaw-Curtis quadrature formula [7]. An extension to dimension 2 has been studied in [20], using two kinds of (hyper)interpolation nodes, those of (2.7) for $d = 2$ (cf. [24]) and the Padua points (cf. [2, 3]).

For the computation of the weights $\{\lambda_\xi\}$, we can use now a similar approach to the one used for the computation of the coefficients $\{c_\alpha\}$, resorting again to a Discrete Cosine Transform, where the roles of the points ξ and of the 3-indexes α are interchanged. Our algorithm below is in line with the one for computing weights for 1-dimensional Clenshaw-Curtis rules given by J. Waldvögel in [22].

First, observe that the orthogonal moments $\{m_\alpha\}$ are simply

$$m_\alpha = m_{\alpha_1} m_{\alpha_2} m_{\alpha_3}, \quad m_{\alpha_j} = \int_{-1}^{1} \hat{T}_{\alpha_j}(t)\, dt. \tag{4.6}$$

Now, defining

$$\mu(\alpha) = \mu(\alpha_1, \alpha_2, \alpha_3) = \begin{cases} m_\alpha, & |\alpha| \le n, \\ 0, & \alpha \in \{0, 1, \dots, n\}^3 \backslash \{\alpha : |\alpha| \le n\} \end{cases} \tag{4.7}$$

we can write

$$\begin{aligned} \lambda_\xi &= w_\xi \sum_{|\alpha| \le n} \hat{T}_{\alpha_1}(\xi_1) \hat{T}_{\alpha_2}(\xi_2) \hat{T}_{\alpha_3}(\xi_3) m_\alpha \\ &= w_\xi \sum_{\alpha_3=0}^{n} \left(\sum_{\alpha_2=0}^{n} \left(\sum_{\alpha_1=0}^{n} \mu_\alpha \hat{T}_{\alpha_1}(\xi_1) \right) \hat{T}_{\alpha_2}(\xi_2) \right) \hat{T}_{\alpha_3}(\xi_3) \\ &= w_\xi G_{ijk}, \end{aligned}$$

where

$$G_{ijk} = \sum_{\alpha_3=0}^{n} \left(\sum_{\alpha_2=0}^{n} \left(\sum_{\alpha_1=0}^{n} \mu_\alpha \beta_{\alpha_1} \cos \frac{k\alpha_1\pi}{n+1} \right) \beta_{\alpha_2} \cos \frac{j\alpha_2\pi}{n+1} \right) \beta_{\alpha_3} \cos \frac{i\alpha_3\pi}{n+1},$$

cf. (3.5) for the definition of β_α, and the triples $(i, j, k) \in \{0, 1, \dots, n+1\}^3$ are such that

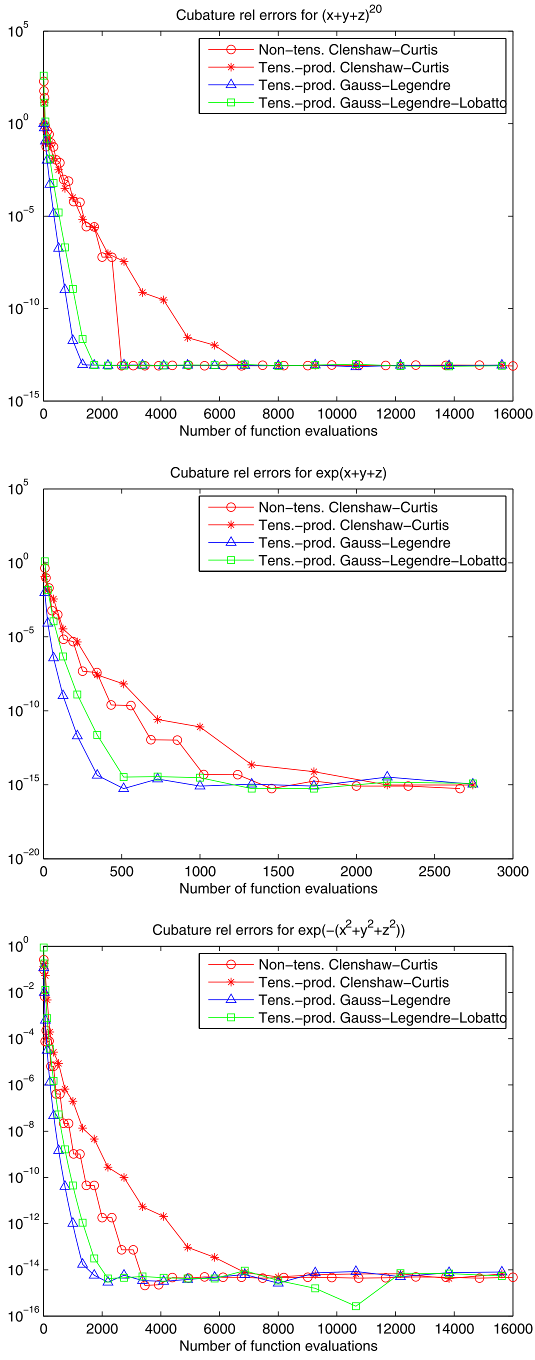
$$\xi = \left(\cos \frac{i\pi}{n+1}, \cos \frac{j\pi}{n+1}, \cos \frac{k\pi}{n+1} \right) \in X_n,$$

cf. (3.2).

This shows, that the 3-dimensional weights array $\{\lambda_\xi\}$ is a scaled Discrete Cosine Transform of the 3-dimensional array $\{\beta_\alpha \mu_\alpha\}$, where we eventually pick only up the $\mathrm{card}(X_n) \approx n^3/4$ cubature weights corresponding to the points in X_n. A fast implementation of the corresponding Clenshaw-Curtis-like formula in the cube is now feasible (for example in Matlab), again via the FFT, just mimicking the Algorithm for total degree hyperinterpolation in Sect. 3.

In Figs. 5–6 we display the relative errors of such a formula for $(\sigma_1, \sigma_2, \sigma_3) = (E, E, E)$ (cf. (3.2)) on the six test functions already used above, compared with those of the tensor-product Clenshaw-Curtis, Gauss-Legendre, and Gauss-Legendre-Lobatto formulae. The numerical results have been obtained with Matlab, using [10] for the Gaussian formulae and [23] for the tensor-product Clenshaw-Curtis formula.

Fig. 5 Relative cubature errors versus the number of cubature points for three test functions (nontensorial Clenshaw-Curtis-like formula)



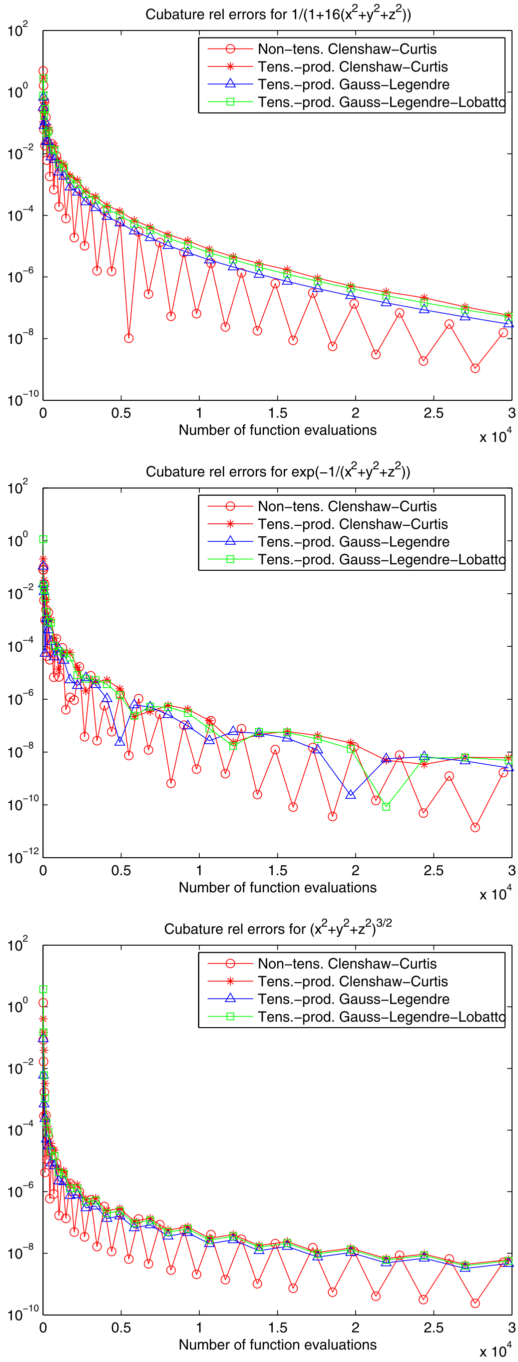


Fig. 6 Relative cubature errors versus the number of cubature points for three test functions (nontensorial Clenshaw-Curtis-like formula)

In particular, we see that with the entire test functions nontensorial Clenshaw-Curtis cubature is more accurate than the tensor-product version, but less accurate than the other two tensor-product formulae. On the other hand, in the less smooth cases the nontensorial Clenshaw-Curtis formula is better than all the other three, especially for *odd* hyperinterpolation degrees n, which correspond to use $n+1$ *even* in (2.7) (again a sort of parity phenomenon, cf. Fig. 2). This behavior echos that of 1-dimensional and 2-dimensional Clenshaw-Curtis formulae (see [20, 21]). Other numerical tests (not reported for brevity) have shown that the other versions of the nontensorial Clenshaw-Curtis formula, corresponding to

$$(\sigma_1, \sigma_2, \sigma_3) = (E, E, O),\ (E, O, E),\ (O, E, E)$$

in (3.2), produce essentially the same results.

Last, but not least, Trefethen [21] explained beautifully when and why Clenshaw-Curtis rules are as good as Gauss-Legendre rules in the one dimensional case. A natural question is if one can extend the results in [21] to our Clenshaw-Curtis rules on the cube. This question, however, seems to be difficult. First, there is no known non-tensorial Gaussian cubature, so that we can only make comparison with tensor product Gaussian cubature. Second, it turns out that our cubature preserves a polynomial subspace larger than Π^3_{2n-1} (see [13]), although not Π^3_{2n}. Moreover, we do not yet know the approximation properties of this polynomial subspace. We wish to return to this problem in the future.

Acknowledgements We thank an anonymous referee for his careful reading and constructive comments.

References

1. Bojanov, B., Petrova, G.: On minimal cubature formulae for product weight functions. J. Comput. Appl. Math. **85**, 113–121 (1997)
2. Bos, L., Caliari, M., De Marchi, S., Vianello, M., Xu, Y.: Bivariate Lagrange interpolation at the Padua points: the generating curve approach. J. Approx. Theory **143**, 15–25 (2006)
3. Bos, L., De Marchi, S., Vianello, M., Xu, Y.: Bivariate Lagrange interpolation at the Padua points: the ideal theory approach. Numer. Math. **108**, 43–57 (2007)
4. Caliari, M., De Marchi, S., Montagna, R., Vianello, M.: HYPER2D: a numerical code for hyperinterpolation at Xu points on rectangles. Appl. Math. Comput. **183**, 1138–1147 (2006)
5. Caliari, M., De Marchi, S., Vianello, M.: Hyperinterpolation on the square. J. Comput. Appl. Math. **210**, 78–83 (2007)
6. Caliari, M., De Marchi, S., Vianello, M.: Hyperinterpolation in the cube. Comput. Math. Appl. **55,**, 2490–2497 (2008)
7. Clenshaw, C.W., Curtis, A.R.: A method for numerical integration on an automatic computer. Numer. Math. **2**, 197–205 (1960)
8. De Marchi, S., Vianello, M.: `Hyper3`: a Matlab code for fast polynomial hyperinterpolation in the 3-cube. Preliminary version available at: http://profs.sci.univr.it/~demarchi/software.html
9. Dunkl, C.F., Xu, Y.: Orthogonal Polynomials of Several Variables. Encyclopedia of Mathematics and its Applications, vol. 81. Cambridge University Press, Cambridge (2001)
10. Gautschi, W.: Orthogonal Polynomials: Computation and Approximation. Numerical Mathematics and Scientific Computation. Oxford Science Publications. Oxford University Press, New York (2004)
11. Hansen, O., Atkinson, K., Chien, D.: On the norm of the hyperinterpolation operator on the unit disk and its use for the solution of the nonlinear Poisson equation. IMA J. Numer. Anal., published online 20 March 2008
12. Krylov, V.I.: Approximate Calculation of Integrals. Macmillan, New York (1962)

13. Li, H., Sun, J., Xu, Y.: Cubature formula and interpolation on the cubic domain. Numer. Math. Theory Method Appl. (accepted for publication)
14. Möller, H.M.: Kubaturformeln mit minimaler Knotenzahl. Numer. Math. **25**, 185–200 (1976)
15. Morrow, C.R., Patterson, T.N.L.: Construction of algebraic cubature rules using polynomial ideal theory. SIAM J. Numer. Anal. **15**, 953–976 (1978)
16. Plésniak, W.: Remarks on Jackson's theorem in $\mathbb{R}^N$. East J. Approx. **2**, 301–308 (1996)
17. Reimer, M.: Multivariate Polynomial Approximation. International Series of Numerical Mathematics, vol. 144. Birkhäuser, Basel (2003)
18. Sloan, I.H.: Interpolation and Hyperinterpolation over General Regions. J. Approx. Theory **83**, 238–254 (1995)
19. Sloan, I.H., Womersley, R.: Constructive polynomial approximation on the sphere. J. Approx. Theory **103**, 91–118 (2000)
20. Sommariva, A., Vianello, M., Zanovello, R.: Nontensorial Clenshaw-Curtis cubature. Numer. Algorithms **49**, 409–427 (2008)
21. Trefethen, L.N.: Is Gauss quadrature better than Clenshaw-Curtis? SIAM Rev. **50**, 67–87 (2008)
22. Waldvögel, J.: Fast construction of the Fejér and Clenshaw-Curtis quadrature rules. BIT **46**, 195–202 (2006)
23. von Winckel, G.: `flencurt.m`—fast Clenshaw-Curtis quadrature. Available online at Matlab central file exchange: http://www.mathworks.com/matlabcentral/fileexchange
24. Xu, Y.: Lagrange interpolation on Chebyshev points of two variables. J. Approx. Theory **87**, 220–238 (1996)
25. Xu, Y.: Polynomial interpolation in several variables, cubature formulae, and ideals. Adv. Comput. Math. **12**, 363–376 (2000)