SIAM Review
Vol. 59, No. 3, pp. 469–491

Cubature, Approximation, and Isotropy in the Hypercube*

Lloyd N. Trefethen[†]

Abstract. Algorithms that combat the curse of dimensionality take advantage of nonuniformity properties of the underlying functions, which may be rotational (e.g., grid alignment) or translational (e.g., near-singularities localized at certain points of the domain). The significance of such effects is explored for four different classes of algorithms: low-rank compression, quasi-Monte Carlo integration, sparse grids, and cubature. The exponentially pronounced computational consequences of the anisotropy of the hypercube are described, notably its mismatch with the isotropy of the set of multivariate polynomials of a fixed degree on which some cubature formulas are based, and it is observed that the tensor product Gauss quadrature rule in $[-1,1]^d$ requires up to $(\pi/2)^d$ times fewer points if it is transformed to a nonpolynomial basis.

Key words. cubature, approximation, isotropy, sparse grids, quasi-Monte Carlo, low-discrepancy sequence, Padua points, curse of dimensionality, Euclidean degree, Euclidean cubature

AMS subject classifications. 41A63, 65D32, 65Y20

DOI. 10.1137/16M1066312

1. Introduction. Many algorithms of cubature, approximation, solution of integral equations or PDEs or stochastic PDEs, uncertainty quantification, data science, and optimization aim to diminish the "curse of dimensionality." Such algorithms take advantage of special properties of the functions being treated, such as alignment with the axes, but their authors do not always emphasize this aspect of their methods. For example, fast convergence is sometimes proved for functions that are "smooth," with smoothness defined by a measure that is anisotropic (i.e., not invariant with respect to rotations), typically involving mixed derivatives. Our aim here is to highlight the importance of rotational and translational nonuniformity for the effectiveness of such algorithms and to urge that these properties should be considered more explicitly in their analysis. An analogy to the theory of convergence of Krylov matrix iterations will be proposed in the conclusions.

The comparator throughout this discussion is the algorithm in the hypercube that achieves everything one could want, but at exponential cost:

0. *Tensor products.*

We shall focus on four classes of algorithms, whose combined literature numbers thousands of books and papers:

*Received by the editors March 17, 2016; accepted for publication (in revised form) June 21, 2016; published electronically August 7, 2017.

http://www.siam.org/journals/sirev/59-3/M106631.html

Funding: This work was supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007–2013)/ERC grant agreement 291068. The views expressed in this article are not those of the ERC or the European Commission, and the European Union is not liable for any use that may be made of the information contained here.

[†]Mathematical Institute, University of Oxford, Oxford, OX2 6GG, UK (trefethen@maths.ox.ac.uk).

1. *Low-rank compression.*
2. *Quasi-Monte Carlo (QMC) integration.*
3. *Sparse grids.*
4. *Polynomial-based cubature and Padua points.*

The first three beat the curse of dimensionality at least in part by being anisotropic, taking advantage of alignment of functions with the axes, and as a result, all of these methods are far from invariant with respect to changes of variables. The first method additionally takes advantage of localization of near-singularities. The fourth is, in a sense, perfectly isotropic, since it is based on the set of multivariate polynomials of a fixed degree, which is invariant with respect to rotations, but we shall show that, paradoxically, this introduces a difficulty for problems posed in the hypercube, because this domain is far from isotropic. Specifically, we shall argue that algorithms based on multivariate polynomials of fixed degree have anisotropic resolution power in the hypercube, giving less resolution along diagonals than along diameters, and propose the consideration instead of polynomials of what we call fixed *Euclidean degree.*

There is no doubt that all of these algorithms are useful in applications. The explanation for this would appear to be that the functions that arise in applications are not mathematically arbitrary, but tend to have special properties. This hints at the complexity of the subject, for it can never be easy to make statements about the deviation from randomness of practical problems.

This paper is organized as follows. For each of the four classes of methods, we first describe a uniformity issue that arises and demonstrate its importance. The essential issues are described with the help of experiments and figures mainly for $d = 2$, where multidimensional effects appear in their simplest form. We then discuss theorems from the literature of the method under consideration to illustrate how anisotropy or translational nonuniformity may be reflected in a rigorous analysis. Theoretical analysis in the low-rank, QMC, and sparse grids literatures is typically based on (anisotropic) measures of smoothness, whereas the cubature literature emphasizes polynomial degree of exactness.

In the final section 7 we show that a change of variables in each dimension can improve the complexity of tensor product representations by an exponential factor:

$0'$. *Transformed tensor product grids.*

This observation is in line with the discussion of the earlier sections in that, once again, the key point concerns nonuniformity of standard algorithms.

Of course, some authors do discuss the special properties of multivariate functions that enable certain algorithms to be effective, and we conclude this introduction with three examples. On p. 136 of their survey of QMC integration Dick, Kuo, and Sloan write [23]:

> Note that we do *not* say that all high-dimensional problems can be successfully tackled by QMC methods. Rather, the interest is in recognizing and analysing mathematically the particular features that make *some* high-dimensional problems manageable.

Another example comes from the introduction to a recent paper of Dahmen et al. [21]:

> Solutions to real-world high-dimensional problems are thought to have a structure different from high-dimensional regularity that renders them more amenable to numerical approximation. The challenge is to explicitly define these new structures, for a given class of problems, and then build numerical methods that exploit them.

Finally, from p. 16 of Constantine's book on active subspaces [20]:

The only way to fight the curse of dimensionality is to identify and exploit special structure in the model.

2. Tensor Products. "Method zero" is tensor products. This is the most straightforward procedure by which, for generations, expansions and numerical methods on a one-dimensional interval have been lifted up to a higher-dimensional box.

For numerical convenience, our hypercube will be $[-1,1]^d$ rather than $[0,1]^d$. In one dimension, a function may be approximated by a polynomial

$$p(x) = \sum_{i=0}^{n-1} a_i T_i(x), \tag{2.1}$$

which we have written in terms of Chebyshev polynomials T_i rather than monomials x^i since this is good practice numerically. Such an approximation might be obtained, for example, by interpolation in a grid of n Chebyshev points. The two-dimensional analogue of (2.1) is the bivariate polynomial

$$p(x,y) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} a_{ij} T_i(x) T_j(y), \tag{2.2}$$

and in d dimensions it is a multivariate polynomial of *maximal degree* $n-1$,

$$p(\mathbf{x}) = \sum_{i_1=0}^{n-1}\cdots\sum_{i_d=0}^{n-1} a_{i_1,\dots,i_d} T_{i_1}(x_1)\cdots T_{i_d}(x_d), \tag{2.3}$$

with $\mathbf{x} = (x_1,\dots,x_d)$. A multivariate polynomial approximation of this form can be obtained by interpolation in the tensor product grid of $N = n^d$ points obtained by taking the same n values as before in each coordinate, as illustrated later in Figure 10 for $n = 16$ and $d = 2$; for any data on the grid, there is a unique interpolant. We shall return to polynomial interpolation in section 6.

3. Low-Rank Compression. A powerful tool in many areas of computational science is the approximation of matrices, tensors, and their multivariate function analogues by structures of low rank. (A function $f(x,y)$ is the continuous analogue of a matrix $A = (A_{ij})$, and a function such as $f(x,y,z)$ in three or more dimensions is the continuous analogue of a tensor.) Such ideas have been exploited by Bebendorf [2, 3], Carvajal, Chapman, and Geddes [15], Drineas, Kannan, and Mahoney [25], Goreinov et al. [29, 54], Gorodetsky, Karaman, and Marzouk [30], Grasedyck, Kressner, and Tobler [31], Hackbusch [36], Khoromskij [43], and many others, including my own Chebfun2 and Chebfun3 projects for computing with functions in two or three dimensions, joint work with Townsend and Hashemi, respectively [26, 39, 64]. Some further references include [6, 21, 38].

Details vary, but the basic idea of low-rank representation of a multivariate function $f(\mathbf{x})$ in d variables $\mathbf{x} = (x_1,\dots,x_d)$ is to represent it by a finite sum of separated products,

$$f_r(\mathbf{x}) = \sum_{i=1}^{r} g_1^{(i)}(x_1)\, g_2^{(i)}(x_2)\cdots g_d^{(i)}(x_d) \tag{3.1}$$

(not always written in this fully separated form). If $r = 1$, we have the familiar case of a *separable* function, such as $f(x, y) = g(x)h(y)$ in two dimensions. In many applications good approximations are obtained for surprisingly small values of r, which is called the *tensor rank* of f_r if the representation is minimal [44]. For example, $f(\mathbf{x}) = \exp(\alpha(x_1^2+\cdots+x_d^2))$ has rank 1 for any α and d, and $f(\mathbf{x}) = \sin(\alpha(x_1+\cdots+x_d))$ has rank d (over the real field; the rank is 2 over the complex field [36]). A function that depends on only one variable, like $f(\mathbf{x}) = \tanh(\alpha(x_1))$, has rank 1. For $d = 2$, optimal low-rank approximations are described by the singular value decomposition (SVD), and there are generalizations of the SVD to higher dimensions [44, 47]. The methods referenced above, which can be regarded as approximate generalizations of Gaussian elimination with complete pivoting [63, 65], are computationally faster than the SVD, though not optimal in the approximation sense.

Low-rank approximations have been advocated in dimensions ranging from 2 up to hundreds and higher. At the high-dimensional end, which has found application in quantum mechanics and stochastic/parametric PDEs, among other areas, a method that has received particular attention is tensor-train decomposition [30, 43, 53]. At the low-dimensional end, a major part of this literature is devoted to representing large matrices not by global low-rank approximations, but by low-rank approximations of submatrices composed in a hierarchical fashion, which we shall comment on in the last two paragraphs of this section. A survey of low-rank representations can be found in [31].

The following example shows that compressing a function by low-rank approximation is not always possible. Suppose the hyperbolic tangent mentioned above is rotated in d-space to become

$$f(\mathbf{x}) = \tanh(\alpha(d^{-1/2}(x_1 + \cdots + x_d))). \tag{3.2}$$

Instead of rank 1, f now has infinite rank. Computationally we ask, what rank is needed to approximate it to a given precision? In the case $d = 2$, we can explore the matter using Chebfun, which by default computes approximations (2.1) and (2.2) to about 15 digits of accuracy:

```
>> f = chebfun2(@(x,y) tanh(10*x));
>> rank(f)
ans = 1

>> f = chebfun2(@(x,y) tanh(10*(x+y)/sqrt(2)));
>> rank(f)
ans = 103
```

This experiment reveals that with $\alpha = 10$, the aligned function has rank 1, as expected, whereas the rotated one has rank 103, a number that grows linearly with α. Further experiments show that to represent this function to 15-digit precision, a tensor product Chebyshev grid needs about $14\alpha \times 14\alpha$ points, whereas the low-rank representation requires about 9α terms each involving two vectors of length about 18α. Thus, the low-rank representation achieves no compression.

Figure 1, taken from the "square and round pegs" example at www.chebfun.org, gives a visual illustration of the dependence of low-rank compressibility on alignment with axes. The two functions shown are

$$f(x, y) = \frac{1}{(1 + (2x)^{20})(1 + (2y)^{20})} \tag{3.3}$$

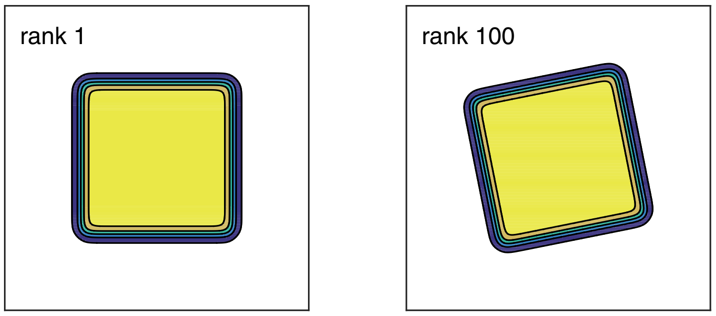


Fig. 1 *"Square peg" and "tilted peg" examples* (3.3)–(3.4)*, both resolved to about* 15*-digit accuracy in the unit square by Chebfun*2*. Chebfun*2 *constructs low-rank representations by an approximation to a continuous analogue of the algorithm of Gaussian elimination with complete pivoting of numerical linear algebra* [63, 64]*. First a rank-*1 *approximation is formed that captures the largest function value in the square; then a second rank-*1 *piece is constructed that captures the largest value of the error in the first approximation; and so on iteratively until machine precision is reached.*

and its rotated and slightly expanded cousin

$$g(x,y) = \frac{1}{(1+(2x+0.4y)^{20})(1+(2y-0.4x)^{20})}. \tag{3.4}$$

The first has rank 1, but the second has numerical rank 100.

Besides functions aligned with the axes, low-rank representations can also be effective in dealing with functions whose regions of complexity are localized. For a two-dimensional example, the Runge function

$$f(x,y) = (10 + x^2 + y^2)^{-1} \tag{3.5}$$

can be approximated to 15-digit precision by a 17×17 tensor product (2.2), or by a rank $r = 5$ representation (3.1) involving $2r = 10$ univariate functions of degree 20; this is not much compression. If the singularity is made narrower by changing 10 to 0.01, however, the tensor product dimension increases to 330×330, whereas the low-rank approximation increases only to rank 21 with univariate functions of degree 380. Thus, for this second and sharper function, the compression is substantial. It depends on the singularity being localized, however. If a second singularity is introduced,

$$f(x,y) = (0.01 + x^2 + y^2)^{-1} + (0.01 + (x-0.5)^2 + (y-0.5)^2))^{-1}, \tag{3.6}$$

the rank doubles to 43 and thus the compression is halved. The localization effect is illustrated in Figure 2, which shows skeletons of low-rank approximations to

$$f(x,y) = \left((x+1.05)^2 + (y+1.05)^2\right)^{-1/2} \tag{3.7}$$

and

$$g(x,y) = \frac{1}{0.001 + (x-0.2)^2 + (y-0.5)^2}. \tag{3.8}$$

In both cases the low-rank representation concentrates attention along lines where the function is rapidly varying.

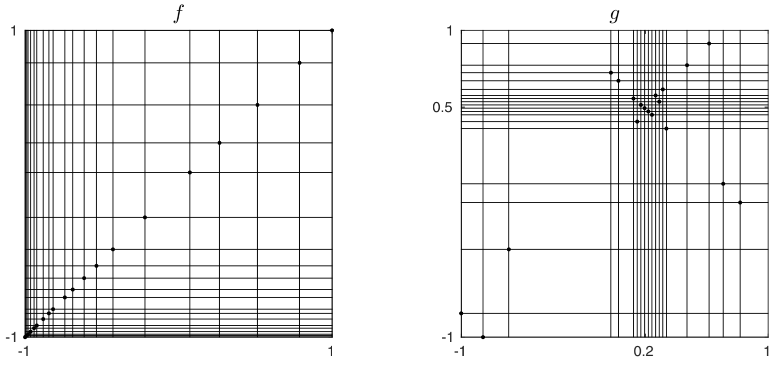


Fig. 2 *Successful low-rank compression of the functions* (3.7) *and* (3.8) *with localized near-singularities, both computed by Chebfun* [64] *with accuracy specification* 10^{-12} *and rank* 19. *The function* f *has a singularity off the bottom-left corner at* $x = y = -1.05$, *and* g *has complex singularities near* $x = 0.2$, $y = 0.5$. *Dots mark the pivot locations as the* 19 *rank*-1 *pieces are successively added up, each such step making the error zero on the corresponding horizontal and vertical lines. Tensor product representations to the same accuracy would require on the order of* 10^4 *and* 10^5 *data values, respectively. In higher dimensions the savings for functions with localized singularities can be huge.*

The foregoing examples make it clear that low-rank approximations can be much more efficient than tensor products in some cases, but not all. This raises the question of what theorems are stated in the literature to support their use. To discuss this we must distinguish the two cases of global and hierarchical representations.

First, concerning global low-rank approximations, there are theorems showing effective approximation of functions dominated by singularities near one edge or corner of the domain, as in the function f of Figure 2. See, for example, Theorem 1 of [3]. Such analysis might make use of a notion stemming from [10] of a function being *asymptotically smooth* [2, 36, 67], which is a precise formulation of the condition that the smoothness increases with distance from the singularity.

A quite different set of theorems for tensor algorithms for PDEs appears in [21], where it is shown under suitable hypotheses that if the right-hand side of an elliptic PDE can be approximated with low rank, then the solution can be approximated with low rank too. An analogous result for Lyapunov equations is given in [56].

On the other hand, there are hierarchical representations, applicable for lower dimension d. These have a long history, featuring celebrated ideas such as the fast multipole method [32], wavelets [5], and Calderon–Zygmund decomposition [14]. Closer to this paper, see the books by Bebendorf [2] and Hackbusch [36] and many other items in our bibliography. The prototypical application of these representations is to represent a function $f(x, y)$ that is the kernel of an integral operator singular along the diagonal line $x = y$. Viewed as a whole, these are not low-rank representations but geometrically graded ones, with fine structure near the diagonal and coarser structure further away. It is the individual patches, which Tyrtyshnikov calls the tiles of a mosaic, that have low rank, and again a common assumption for analysis is that of asymptotic smoothness. Under this hypothesis, theorems about efficient representations based on low-rank tiles have been proved. On an individual tile, the curse of dimensionality has not been beaten, and indeed, the low-rank approximation on an individual tile might be represented by a tensor product; it is only in the large that compression is achieved by the exploitation of the graded nature of the function.

In summary, both global and hierarchical low-rank representations can take advantage of the special structure of smoothness increasing with distance from singularities, but it would appear that there is no theory that claims that the representations can beat the curse of dimensionality for approximating general functions. This reflects the familiar fact from linear algebra that although some matrices have singular values decreasing rapidly to zero, many do not.

4. Quasi-Monte Carlo Integration.

The low-rank representations of the last section can be applied to all sorts of problems of function approximation, differential equations, integral equations, optimization, and other areas. In turning to our second class of algorithms, we narrow down to the conceptually simplest of all applications, cubature, which is a standard term for numerical integration in multiple dimensions [18, 35, 46]. Suppose an integrable function f is defined in the hypercube $[-1,1]^d$ and we are interested in its integral $I = I[f]$ over this domain. A *cubature formula* is a functional for approximating I of the form

$$I_N = I_N[f] = \sum_{i=1}^{N} w_i f(\mathbf{s}_i), \tag{4.1}$$

defined by *nodes* $\mathbf{s}_i \in [-1,1]^d$ and *weights* $w_i \in \mathbf{R}$. For $d = 1$, the special case of quadrature, Gauss and Clenshaw–Curtis and related formulas give high accuracy if f is smooth. Specifically, if f has a piecewise continuous νth derivative of bounded variation for $\nu \ge 2$, the convergence is algebraic [70],

$$I - I_N = O(N^{-\nu-1}), \tag{4.2}$$

and if f is analytic, the convergence is geometric [66],

$$I - I_N = O(C^{-N}), \quad C > 1. \tag{4.3}$$

(Note that these formulas pertain to N-point quadrature formulas applied on a global grid, corresponding to what is called *p-convergence* in the literature of finite element methods, as opposed to quadrature formulas like Simpson's rule constructed by composition of fixed-degree pieces, corresponding to *h-convergence.*) In d dimensions, one may take a tensor product of n-point Gauss rules to obtain a product Gauss rule with $N = n^d$ points. The mathematics is essentially the same, but because N grows exponentially with d, this corresponds to much slower convergence,

$$I - I_N = O(N^{(-\nu-1)/d}), \quad I - I_N = O(C^{-N^{1/d}}). \tag{4.4}$$

For $d = 2$ or 3 these rates may still be quite good, but in higher dimensions they become increasingly unsatisfactory.

And so there has been great interest in devising alternative cubature formulas. An extreme case is *Monte Carlo integration,* in which the nodes $\mathbf{s}_i$ are simply random points in the hypercube and the weights w_i are all equal to $2^d/N$,

$$I_N[f] = \frac{2^d}{N} \sum_{i=1}^{N} f(\mathbf{s}_i). \tag{4.5}$$

Here the convergence rate is

$$I - I_N = O(N^{-1/2}), \tag{4.6}$$

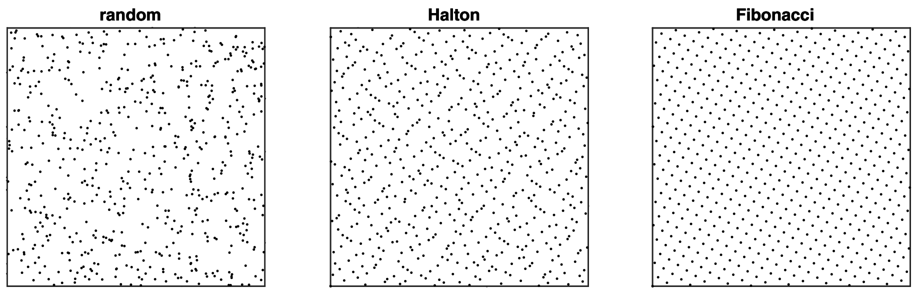


Fig. 3 610-*point random and QMC point sets in the unit square. Unlike random points, QMC points generally depend on the orientation of the axes, as does much of the mathematical analysis of their effectiveness.*

independent of d and of the smoothness of f (the constant in the "O" depends only on the variance of f) [13, 23]. *Quasi-Monte Carlo (QMC) formulas* aim to improve this rate while still being efficient in high dimensions [13, 23, 50]. Their characteristic feature is that all the weights are equal, so that (4.5) still applies, but the points $\mathbf{s}_i$ are chosen in a more uniform fashion by a procedure that is partly or fully deterministic. The traditional aim of such methods is convergence at a rate closer to $O(N^{-1})$, which can be a dramatic improvement over (4.6). Theoretical bounds often take the (disappointingly large) form

$$I - I_N = O((\log N)^{d-1}N^{-1}), \tag{4.7}$$

or sometimes the same form with $(\log N)^{d-1}$ replaced by $(\log N)^d$.

Among the well-known choices of points for QMC simulations are the *Halton* sequences and *Fibonacci* sets. Figure 3 illustrates these in two dimensions, taking $N = 610$ for the comparison because that is a convenient choice for the Fibonacci case.

Loosely speaking, one hopes for good convergence of a QMC method if the function is smooth and the points are well distributed. A foundational theorem for making this precise is the *Koksma–Hlawka inequality* [13, 23, 41, 50]. This result asserts that the error can be bounded by the product of the discrepancy D_N^* of the set of points and the variation $V[f]$ of the integrand:

$$|I - I_N| \le D_N^* V[f]. \tag{4.8}$$

The first of these quantities, the *(star-)discrepancy* D_N^*, is defined as

$$D_N^* = \max_{\mathbf{x}\in[-1,1]^d} |I[\chi_{\mathbf{x}}] - I_N[\chi_{\mathbf{x}}]|\,, \tag{4.9}$$

where $\chi_{\mathbf{x}}$ is the characteristic function corresponding to the d-dimensional rectangular box delimited by vertices $\mathbf{x}$ and $x_1 = \cdots = x_d = -1$ (one corner of the hypercube). This is a measure of how uniformly distributed the sample points are, and indeed, QMC sequences are also called *low-discrepancy sequences.* The best-distributed point sets known achieve $D_N^* = O((\log N)^{d-1}N^{-1})$ or $O((\log N)^d N^{-1})$ and that is where (4.7) comes from. The other quantity on the right-hand side of the Koksma–Hlawka inequality, the *Hardy–Krause variation* $V[f]$, can be defined in terms of integrals of mixed first partial derivatives of f if they are integrable:

$$V[f] = \sum_{\emptyset \neq u \subseteq \{1:d\}} \int_{[-1,1]^{|u|}} \left| \frac{\partial^{|u|} f}{\partial \mathbf{x}_u}(\mathbf{x}_u; 1) \right| d\mathbf{x}_u. \tag{4.10}$$

The notation indicates a sum over all nonempty subsets u of $\{1{:}d\} = \{1, 2, \ldots, d\}$, with first derivatives taken with respect to the selected variables and the unselected variables held at the boundary value 1.

Both D_N^* and $V[f]$ are anisotropic, for they are defined in terms of the coordinate axes. As an indication of how pronounced the anisotropy is, we note that both of these quantities may change exponentially when a function is rotated. For example, the following is a grid-aligned function with small variation:

$$f(\mathbf{x}) = \sin(k\pi x_1): \quad V[f] \sim 4k$$

(approximation for large k). Rotating the function gives a function with exponentially larger variance:

$$f(\mathbf{x}) = \sin(k\pi d^{-1/2}(x_1 + \cdots + x_d)): \quad V[f] \sim (k\pi d^{-1/2})^d.$$

Thus, rotation can turn a very good function into a very bad one. As a milder example, the square peg function (3.3) of Figure 1 has $V[f] \approx 4.0$, whereas for the tilted peg function (3.4) it is $V[f] \approx 16.5$.

Similarly, an equispaced tensor product grid of $N = n^d$ points has $D_N^* = O(n^{-1}) = O(N^{-1/d})$, whereas if the grid is tilted slightly, D_N^* may fall closer to $O(N^{-1})$. Thus, rotation can turn a very bad grid into a better one. Indeed, it is obvious from Figure 3 that this Fibonacci grid is just a square lattice tilted (this applies to some Fibonacci grids, but not all). It would have a much higher discrepancy if it were aligned with the axes—yet would we really expect that to be a much worse quadrature formula?

Yet QMC methods often work, and they hit the headlines in 1995 when Paskov and Traub evaluated an integral in dimension $d = 360$ arising in finance [55]. The explanation of such successes appears to be that for integrands of practical interest, in the words of Sloan and Woźniakowski, "some coordinate directions are more important than others, in the sense of being in some way more difficult" [23, p. 178]. One technique for analyzing this phenomenon has been to consider settings in which some dimensions are given more weight than others. If the grading of weights is pronounced enough, then high- or even infinite-dimensional integration problems are effectively compactified and rendered manageable [22]. For an extensive review of these issues with references, as well as an outline of the theory of *tractability* of high-dimensional integration problems, see [23, 51, 52].

The discussion above has emphasized the apparently modest (though actually not so modest) goal of improving the error of a Monte Carlo formula from $O(N^{-1/2})$ to $O(N^{-1})$. There is another part of the QMC literature, beginning with the so-called *lattice rules*, that aims to achieve still faster convergence for sufficiently smooth integrands [23, 45, 50]; the Fibonacci grid of Figure 3 is in this category. A prototypical lattice rule is applied to periodic rather than arbitrary functions in the hypercube, and much higher rates of convergence can be proved for functions that are smooth in a certain sense. Given a d-vector $\mathbf{h}$ of integers, first $r(\mathbf{h})$ is defined by

$$r(\mathbf{h}) = \prod_{i=1}^{d} \max\{1, |h_i|\}. \tag{4.11}$$

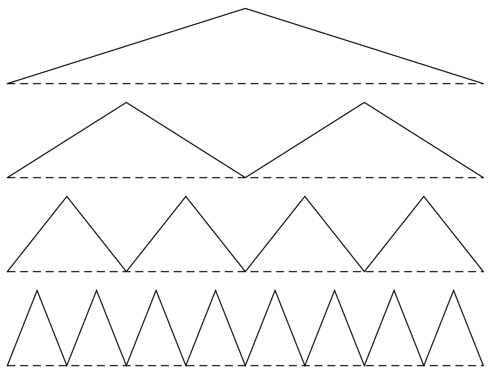


Fig. 4 *Hierarchical basis elements for a space of piecewise linear functions with zero boundary values on an interval in one dimension. The first row shows one basis element, the second two, the third four, and the fourth eight; together these span a space of dimension* 15*. The idea of sparse grids is to employ tensor products of such functions but omit many of the cross-terms that have small support in multiple dimensions.*

Then the smoothness class $\mathcal{E}_{\alpha}^{d}(C)$ of [50, p. 103], for example, is the set of periodic functions on $[-1,1]^d$ that satisfy

$$|\hat{f}(\mathbf{h})| \leq Cr(\mathbf{h})^{-\alpha}, \tag{4.12}$$

where $\hat{f}(\mathbf{h})$ is the coefficient of the Fourier series of f associated with wave number vector $\mathbf{h}$. Similarly, one says that a lattice rule has *Zaremba index* s if it is exact for all functions f [18, p. 9] whose Fourier coefficients are zero when $r(\mathbf{h}) \geq s$. Both of these definitions are strongly anisotropic: (4.11) implies that a small rotation of a function like $\sin(k\pi x_1)$ will change its smoothness exponentially. One might argue that if a periodic function in the hypercube is rotated, it will no longer be periodic, so the idea of rotation should not come into the discussion. However, that argument loses some force since in an actual application, one might have to render a nonperiodic function periodic by a suitable transformation of variables [50, sec. 5.1]. In practice, such transformations in high dimensions are very expensive, and we comment on the reason for this in the penultimate paragraph of the discussion.

More recently there have also been intensified efforts by Dick, Sloan, and others to improve Monte Carlo rates of convergence specifically for nonperiodic functions. See, e.g., [23, 24, 28].

5. Sparse Grids.

The subject of sparse grids begins with the idea of a hierarchical basis for a space of piecewise linear functions in an interval, as sketched in Figure 4 [71]. In one dimension, one could use a basis of 15 hat functions, identical apart from shifts, to span a space of piecewise linear functions with 16 pieces and zero at the endpoints, or one could use this basis of nonidentical functions, and it wouldn't make much difference which one was chosen. In higher dimensions, however, the difference is amplified. In two dimensions, for example, if we take products of 15 hat functions in x and y to form a basis with 225 elements for piecewise linear bivariate functions, then all the basis elements will be supported on little squares of small measure. However, suppose we instead take products in x and y of the 15 hierarchical functions shown in the figure. The space spanned will be the same, but now the difference in support between some of the products and others is large. The vision of sparse grids is that

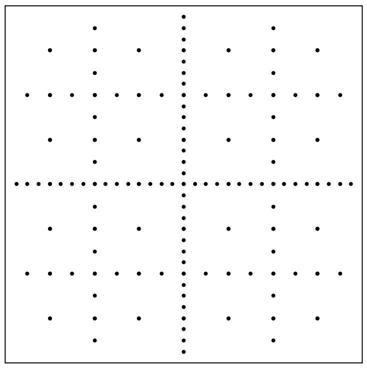


Fig. 5 *Schematic indicator of a sparse grid in $[-1,1]^2$, as described in the text. The full grid would have $(n-1)^2$ points (here, $n = 32$). In the sparse grid, this number reduces to $O(n\log_2(n))$, or $O(n(\log_2(n))^{d-1})$ in d dimensions. The result is good approximation of functions aligned with the axes, or, as this is typically quantified, of functions that are smooth as measured by mixed derivatives.*

many of the basis elements with small support may be discarded without much loss of approximation power. Such ideas were introduced by Smolyak in 1963 [58] and rediscovered independently by Zenger for PDE applications around 1990 [72]; a related idea stemming from Smolyak's era is that of the *hyperbolic cross* [1, 57]. Since 1990, Smolyak/sparse grids methods have been used extensively by many people, and a particularly influential group has been that associated with Michael Griebel [12].

Figure 5 shows a schematic to illustrate the choice of which basis functions are retained in the simplest version of a sparse grid. Each dot p in the figure lies at a point with dyadic coordinates in x and y and corresponds to a product of hat functions in the two directions, taking the value 1 at p and the value 0 at four other corner points. These corner points delimit the smallest rectangle with center p whose vertices belong to a coarser dyadic grid and are also dots in the figure or on the boundary. For example, the central dot of the whole figure corresponds to a function with global support $[-1,1]^2$, and the central dot of each subsquare corresponds to a function with support in that subsquare. The remaining dots sit at the centers of rectangles that are 2, 4, 8, or 16 times as long as they are wide. A basis constructed in this fashion in d dimensions contains only $O(n(\log_2(n))^{d-1})$ elements, exponentially fewer than the figure $O(n^d)$ for the full tensor product basis.

The resolution power of a sparse grid as in Figure 5 is just as we have seen in the past two sections: excellent for functions aligned with the axes, less good for unaligned functions. If f takes the form $f(x,y) = \phi(x)$ for some ϕ, for example, then the grid of the figure will resolve ϕ with 32 grid points. The rotated function $f(x,y) = \phi(2^{-1/2}(x+y))$, on the other hand, must effectively make do with closer to 8 grid points. Such disparities are amplified as the dimension d increases, yet sparse grids have been strikingly effective in some applications. As Conrad and Marzouk explain it, "Smolyak algorithms avoid the exponential cost of full tensor products when the input dimensions are not fully coupled" [17]. This is an established technology that has been extended in many directions, including to spatial adaptivity and elements of higher order than linear.

Looking at the theoretical literature of sparse grids, we find many results to the effect that if a function is smooth, it can be effectively approximated. To make such results possible, smoothness is measured in terms of *mixed derivatives.* If $(k_1, \dots, d_d)$ is a vector of nonnegative integers, then the mixed derivatives of f of order r are those of the form

$$\frac{\partial^{k_1+\cdots+k_d}}{\partial x_1^{k_1}\cdots \partial x_d^{k_d}} \tag{5.1}$$

with $\max_j k_j \le r$. Thus, $\partial^4 f/\partial x^2 \partial y^2$, for example, counts as a mixed derivative of order 2, whereas $\partial^4 f/\partial x^4$ is of order 4. From here we readily see, as shown earlier, that a function like $\sin(kx)$, $k \gg 1$, may appear smooth if aligned with an axis and much less smooth if rotated. For an example of how these measures are employed in estimating accuracy of sparse grid approximations, see Theorems 1 and 2 of [11], which state 2-, ∞-, and energy-norm upper bounds derived from (5.1) with $k_1 = \cdots = k_d = 2$. One of the early proponents of analysis of this kind was Temlyakov [62].

6. Polynomial-Based Cubature and Padua Points.

The subject of quadrature in one dimension is dominated by the idea of interpolating function samples by a polynomial, then integrating the interpolant. Equispaced points lead to Newton–Cotes quadrature, Chebyshev points to Clenshaw–Curtis, and roots of Legendre polynomials to Gauss. Polynomials are equally powerful for other one-dimensional computational problems too and, in particular, they are the basis of spectral methods for numerical solution of differential equations [9] and of Chebfun [66].

To find analogous non-Monte Carlo formulas in multiple dimensions, one turns to multivariate polynomials as in (2.3). The more usual way to describe a multivariate polynomial is as a linear combination of monomials of the form

$$\mathbf{x}^{\mathbf{k}} = x_1^{k_1} x_2^{k_2} \cdots x_d^{k_d}, \tag{6.1}$$

with the *degree* (also known as the *total degree*) defined as $|\mathbf{k}| = \sum k_i$. The degree of a polynomial is the maximum of the degrees of its monomials, which is very different from the maximal degree mentioned in section 2. The reason for this standard definition has to do with isotropy, or as an algebraist would put it, invariance with respect to linear transformations. For example, x^2 is a homogeneous degree-2 polynomial in x. Rotation into x-y-z space gives

$$\left(\frac{x+y+z}{\sqrt{3}}\right)^2 = \frac{x^2+y^2+z^2+2xy+2xz+2yz}{3}.$$

All the terms are still homogeneous of degree 2 according to the standard definition, and the polynomial is still of degree 2.

Interpolation by polynomials in the multivariate case is straightforward if one is prepared to work with a tensor product grid of $N = n^d$ points and polynomials (2.3) of the corresponding maximal degree $n-1$. Interpolation by polynomials of prescribed total degree, however, is not straightforward because of the challenge of selecting interpolation points that guarantee a unique interpolant, the condition known as *unisolvency.* (In two dimensions, the *Padua points* are suitable [7], and there are generalizations to higher dimensions [8, 19].) Nevertheless, much progress has been made in investigating cubature formulas that integrate all multivariate polynomials of degree m exactly, an idea going back to Maxwell [18, 35, 46, 49, 59]. We have defined a cubature formula as a functional (4.1) determined by nodes $\mathbf{s}_i \in [-1,1]^d$ and weights $w_i \in \mathbf{R}$. If I_N is equal to the integral of f over the hypercube whenever f

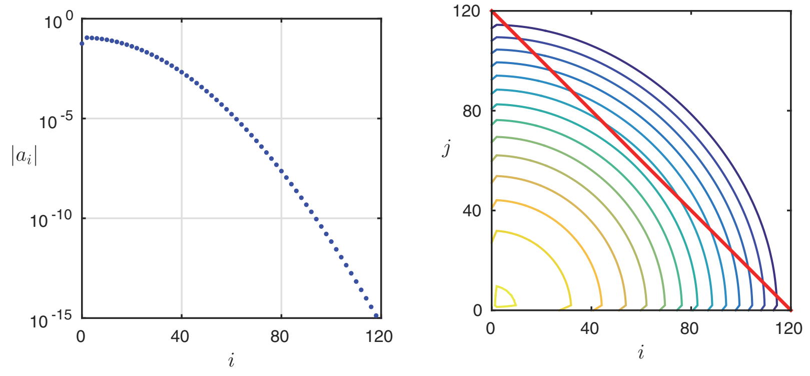


Fig. 6 *On the left, absolute values of Chebyshev coefficients a_i of (6.2) on $[-1,1]$, showing that 120 coefficients suffice to resolve f to 15-digit accuracy. On the right, a contour plot of the absolute values of the tensor product Chebyshev coefficients a_{ij} of the analogous two-dimensional function of (6.4). From inside out, the contours represent $10^{-2}, 10^{-3}, \ldots, 10^{-15}$. These circular arcs may seem to suggest isotropy, but that is not so: isotropy would correspond to the triangle of coefficients below the red line, those that appear in bivariate polynomials of degree 120. The coefficients enclosed by the outermost curve go up to degree about $120\sqrt{2}$.*

is a multivariate polynomial of degree at most m, the formula is said to have *degree m*. *Tchakaloff's theorem* asserts that there exists a degree-m formula with $N \le \binom{m+d}{d}$ points—for any domain, not just the hypercube [60]. Computing nodes and weights of cubature formulas of good polynomial degree, however, can be quite challenging [61].

We come now to the paradox mentioned in the introduction. So far in this article we have highlighted the anisotropy of various high-dimensional approximation strategies. By contrast, these methods tuned to exact integration of multivariate polynomials of a given degree would appear to be perfectly isotropic. The trouble is, the hypercube itself is exponentially far from isotropic! The great apparent gain of a cubature formula based on multivariate polynomials is achieved, in part, by discriminating against most of the volume of the hypercube.

To explain this point, we focus on a concrete example in one, then two, then three dimensions. Consider the univariate function

$$f(x) = e^{-100x^2}. \tag{6.2}$$

On the interval $[-1,1]$, one needs a polynomial of degree about 120 to resolve f to 15-digit precision. This is shown in the left half of Figure 6, which plots the absolute values of the Chebyshev expansion coefficients a_i corresponding to the Chebyshev series

$$f(x) = \sum_{i=0}^{\infty} a_i T_i(x). \tag{6.3}$$

(In Chebfun, the command `chebcoeffs(chebfun(@(x) exp(-100*x.^2)))` computes the coefficients. The odd degree coefficients are zero and do not appear in the plot.) Since f is an entire function (analytic throughout the complex x-plane), the coefficients decrease faster than geometrically [66, Chap. 8], and for $i > 120$ we have $|a_i| < 10^{-15}$.

Now we make the function radially symmetric in two dimensions and consider

$$f(x,y) = e^{-100(x^2+y^2)}. \tag{6.4}$$

The right half of Figure 6 shows a two-dimensonal analogue of the last image. Now, there are Chebyshev coefficients in two variables corresponding to the double series

$$f(x,y) = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} a_{ij} T_i(x) T_j(y). \tag{6.5}$$

The plot shows contours of $|a_{ij}|$ at levels $10^{-1}, \dots, 10^{-15}$. (In Chebfun one can write `chebcoeffs2(chebfun2(@(x,y) exp(-100*(x.^2+y.^2))))`. Again, odd-index coefficients are zero and are omitted.) The function is isotropic, and multivariate polynomials of a fixed degree are isotropic, so one would expect again to need degree 120 for a 15-digit approximation. One's first impression might be that the figure confirms this expectation, but in fact, it reveals that one needs degree about $120\sqrt{2}$. The degree required for resolution has increased by a factor of $\sqrt{2}$.

Why has the degree gone up by $\sqrt{2}$? The surprisingly simple explanation is that the diagonal of the square is $\sqrt{2}$ times longer than its cross-section. In one dimension, for example, if we approximate $\exp(-100x^2)$ to 15 digits on $[-\sqrt{2}, \sqrt{2}]$ instead of $[-1,1]$, the necessary degree increases from 120 to 170, a factor of 1.42. For an isotropic function like (6.4), essentially the same effect will determine the degree needed for resolution along the diagonal direction. (My collaborators and I are developing a theory to make this rigorous, to appear in a future publication.) In d dimensions the factor will be $\sqrt{d}$, and the total number of coefficients needed will exceed what one might have expected by a factor on the order of $(\sqrt{d})^d = d^{\,d/2}$.

Why are the contours in Figure 6 circular? One can explain this (nonrigorously) by applying the argument above to functions oriented at angles other than $\pi/4$. If a function $f(x,y) = \phi(x)$ requires polynomial degree n to be resolved, we consider its counterclockwise rotation by some angle θ with $0 < \theta \le \pi/4$, namely, $f(x,y) = \phi(cx + sy)$ with $c = \cos\theta$ and $s = \sin\theta$. The half-diameter of the unit square as measured at this angle now increases from 1 to $\cos\theta + \sin\theta$. Thus we expect polynomial degree $n(\cos\theta + \sin\theta)$ to be needed for resolution, and this is just the ratio that corresponds to inflating the red line segment of the figure to the circular arc. A similar argument extends this conclusion to higher dimensions: for isotropic resolution in the hypercube, we need multivariate polynomials coefficients lying in a spherical ball, not a simplex. Figure 7 is an analogue of the previous figure in three dimensions.

These considerations suggest that the following definition may be useful: the *Euclidean degree* of a monomial (6.1) is the 2-norm $\|\mathbf{k}\|_2$ of its vector of exponents, and the Euclidean degree of a polynomial is the maximum of the Euclidean degrees of its monomials. For example:

$$x^5y^6z^7: \quad \text{(total) degree } 18, \ \ \text{Euclidean degree } \sqrt{110} \approx 10.5, \ \ \text{max degree } 7.$$

The motivation is that the set of all polynomials of a given Euclidean degree may be well adapted to approximating functions in the hypercube with uniform resolution in all directions.

Figure 8 presents a schematic representation of this surprising situation. The set of polynomials of a given total degree corresponds to the positive orthant of the innermost region of the figure, the unit ball in the 1-norm, whose volume is

$$\mathrm{Vol}_1 = \frac{2^d}{d!} \sim \frac{1}{\sqrt{2\pi d}} \left(\frac{2e}{d}\right)^d. \tag{6.6}$$

(This and the next two formulas describe volumes of the whole ball, not just the fractions in the positive orthant. This is a continuum approximation to a discrete set,

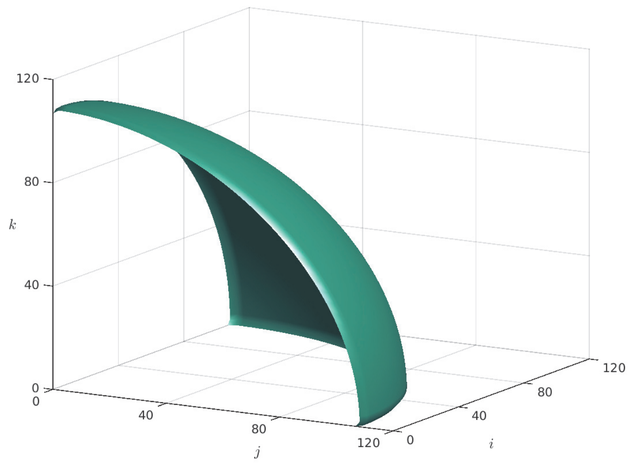
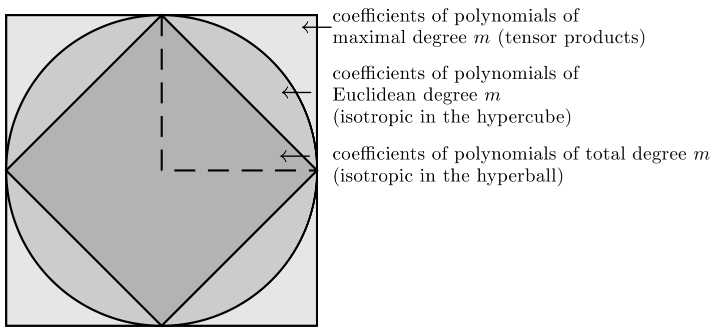


Fig. 7 *Like Figure* 6*, but for the trivariate function* $f(x,y,z) = \exp(-100(x^2+y^2+z^2))$*. The axes represent indices of Chebyshev coefficients in the* x*,* y*, and* z *directions, and the isosurface corresponds to* $|a_{ijk}| = 10^{-15}$*. As before, the conventional expectation would be for this surface to take the form of a flat face of a simplex rather than an orthant of a sphere.*



coefficients of polynomials of maximal degree m (tensor products)
coefficients of polynomials of Euclidean degree m (isotropic in the hypercube)
coefficients of polynomials of total degree m (isotropic in the hyperball)

Fig. 8 *From inside out, the* 1-, 2-, *and* ∞*-norm unit balls, shown in two dimensions as a schematic for* d *dimensions. The sections in the positive orthant correspond to sets of coefficients of polynomials whose volumes differ exponentially for large* m *as* $d \to \infty$*. The innermost set, a simplex, corresponds to the set of polynomials of fixed total degree. The outermost set, a hypercube, corresponds to the set of polynomials of fixed maximal degree, which is needed for tensor product interpolation as in* (2.3)*. The intermediate set, the intersection of a spherical hyperball with the orthant, is what is needed for isotropic resolution in the hypercube.*

and thus our formulas are aimed at elucidation of behavior for large degrees.) The set of polynomials of a given maximal degree corresponds to the positive orthant of the unit ball in the ∞-norm, with volume

$$\mathrm{Vol}_\infty = 2^d. \tag{6.7}$$

For isotropic resolution throughout the hypercube, however, our new observation is that what matters is the in-between region, the positive orthant of the unit ball in

Table 1 *Exponential separation as $d \to \infty$ between the volumes in coefficient space corresponding to the sets of multivariate polynomial coefficients associated with isotropic resolution in the hyperball* (Vol_1), *isotropic resolution in the hypercube* (Vol_2), *and tensor products in a hypercube* (Vol_∞).

Dimension d	Tensor product over-sampling ratio (6.9)	Cubature undersampling ratio (6.10)	Cubature aim (6.11)
1	1	1	1
2	1.27	1.57	2
3	1.91	3.14	6
4	3.24	7.40	24
5	6.08	19.7	120
6	12.4	58.1	720
7	27.1	186.	5040
8	62.1	639.	40320
9	155.	2338.	362880
10	402.	9037.	3628800

the 2-norm,

$$\mathrm{Vol}_2 = \frac{\pi^{d/2}}{(d/2)!} \sim \frac{1}{\sqrt{\pi d}} \left(\frac{2\pi e}{d}\right)^{d/2} \tag{6.8}$$

(for odd dimensions we define $(d/2)! = \Gamma(d/2+1)$). We can thus compute

$$\frac{\mathrm{Vol}_\infty}{\mathrm{Vol}_2} = \frac{(d/2)!}{(\pi/4)^{d/2}} \sim \sqrt{\pi d} \left(\frac{2d}{\pi e}\right)^{d/2}, \tag{6.9}$$

$$\frac{\mathrm{Vol}_2}{\mathrm{Vol}_1} = \frac{(\pi/4)^{d/2} d!}{(d/2)!} \sim \sqrt{2} \left(\frac{\pi d}{2e}\right)^{d/2}, \tag{6.10}$$

$$\frac{\mathrm{Vol}_\infty}{\mathrm{Vol}_1} = d! \sim \sqrt{2\pi d} \left(\frac{d}{e}\right)^{d}, \tag{6.11}$$

and Table 1 lists these ratios for dimensions $d = 1, \ldots, 10$.

Let us summarize our observations and comment on the headings of the columns in Table 1. The final column, the "cubature aim," reflects the ambition of polynomial-based cubature: to get similar accuracy as a tensor product formula using $d!$ times fewer points. The claim of this section is that this aim is about half achievable, for one can view this number as the product of two ratios. A tensor product formula is anisotropic in the sense that it can resolve $\sqrt{d}$ times higher wave numbers along a diagonal than along an axis; the price paid for this in grid points is the "tensor product oversampling ratio." A cubature formula of a given degree, on the other hand, can only resolve $\sqrt{d}$ times *lower* wave numbers along a diagonal than along an axis, because the formula is isotropic while the hypercube is not, and this is quantified by the "cubature undersampling ratio." These considerations suggest the interesting challenge of constructing what one might call *Euclidean cubature formulas*, whose aim would be to neither oversample nor undersample. A Euclidean cubature formula of degree s would be a formula (4.1) that is exact for all polynomials of Euclidean degree s. (Ideas with some of this flavor can be found in [17] and in the theory of *optimized sparse grids* [33].) Perhaps such formulas might be useful in numerical practice.

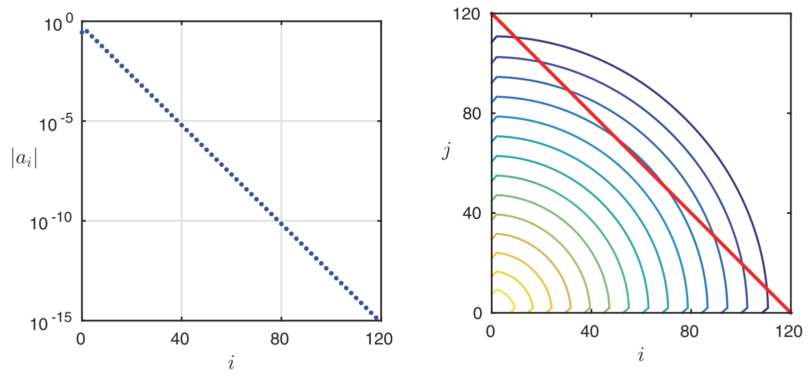


Fig. 9 *The same as Figure* 6, *but for the Runge functions* (6.12). *Again, the coefficients decrease geometrically at a rate determined by the Euclidean degree, not the total degree.*

In this discussion we have considered just the single example trio $\exp(-100x^2)$, $\exp(-100(x^2+y^2))$, $\exp(-100(x^2+y^2+z^2))$, functions that are analytic for all values of x, y, and z. One might wonder if this special property makes the example somehow atypical, but in fact, the same effects appear generally. Another familiar example of a function in x with an isotropic extension to x and y is the Runge function pair

$$f(x) = \frac{1}{1+10x^2}, \qquad f(x,y) = \frac{1}{1+10(x^2+y^2)}, \tag{6.12}$$

which is analytic for $(x,y) \in [-1,1]^2$ but singular for complex values of x and y with $x^2+y^2=-1$. The coefficient 10 has been chosen so that, as in the earlier example, a polynomial of degree about 120 is needed to resolve f on $[-1,1]$, as shown in the left-hand side of Figure 9. The right-hand side of the figure shows that just as before, the expansion coefficients in the bivariate case decrease with Euclidean degree, not total degree. However, note the difference between Figures 6 and 9. For the former, an entire function, the dicurve on the left bends downward and the curves on the right lie closer together further from the origin. For the latter, analytic but not entire, the curve on the left is a straight line and the curves on the right are evenly spaced.

Our illustrations have involved perfectly isotropic functions, exactly invariant with respect to rotation. Of course, the same issues will control resolution in the hypercube of nonisotropic functions too. As mentioned earlier, we expect to provide rigorous formulations of this claim in a future publication.

It was mentioned above that Padua points provide a unisolvent system in $[-1,1]^2$, with generalizations to higher dimensions. There has been some interest in using Padua points to define novel discretizations of PDEs. However, the considerations above suggest that the results of such experiments may be disappointing, since such discretizations may also underresolve along diagonals.

7. Transformed Tensor Product Grids. Throughout this article we have considered alternatives to tensor products. In this final section we return to the tensor product setting and highlight a method proposed in [37] that requires fewer samples than tensor product Gauss quadrature by a factor of up to $(\pi/2)^d$. This method is derived from a principle analogous to isotropy: assuming that a function is translationally as opposed to rotationally uniform. See Chapter 22 of [66] for a general presentation.

Instead of Gauss quadrature in each dimension, the idea is to use a *transformed Gauss* formula obtained by transplanting Gauss quadrature in the variable $t \in [-1,1]$ to the problem variable $s \in [-1,1]$ by a conformal map g. Polynomials have far greater resolution near the ends of an interval than the middle, and this is reflected in the clustering of the nodes in Gauss quadrature and other methods derived from polynomials. The purpose of g is to diminish this effect by replacing polynomials by nonpolynomial functions with more uniform resolution. To this end, g is chosen to map Bernstein ellipses with foci ± 1 in the x-plane onto regions with straighter sides, with the points ± 1 mapping to themselves. As described in [66], one choice for g is the "sausage map" obtained by truncating the Taylor series of $(2/\pi)\sin^{-1}(t)$ at an odd degree and rescaling the result so that $g(\pm 1) = \pm 1$; for degree 9, this gives

$$g(t) = \frac{1}{53089}(40320t + 6720t^3 + 3024t^5 + 1800t^7 + 1225t^9). \tag{7.1}$$

The result is a quadrature formula of the standard form

$$I_n = \sum_{i=1}^{n} w_i f(s_i), \tag{7.2}$$

but with nodes s_i and w_i that differ from the usual. Shown next are MATLAB codes for computing Gauss and transformed Gauss nodes and weights based on (7.1):

```
function [s,w] = gauss(n)
beta = .5./sqrt(1-(2*(1:n-1)).^(-2));
T = diag(beta,1) + diag(beta,-1); [V,D] = eig(T);
s = diag(D); [s,i] = sort(s); w = 2*V(1,i).^2;

function [s,w] = sausage(n);
[sg,wg] = gauss(n); p = 9; c = zeros(1,p+1);
c(p:-2:1) = [1 cumprod(1:2:p-2)./cumprod(2:2:p-1)]./(1:2:p);
c = c/sum(c); s = polyval(c,sg);
cp = c(1:p).*(p:-1:1); w = wg.*polyval(cp,sg)';
```

Figure 10 shows the corresponding 16×16 Gauss and transformed Gauss grids, where the transformation g has been applied in each direction.

Transforming the Gauss formula does not always improve convergence; it may have the opposite effect for functions that are analytic in a large neighborhood of $[-1,1]^d$ or which are dominated by singularities near the boundary. For many cases it accelerates convergence, however, and, in particular, this is provably the case in the class of functions analytic in a small ε-neighborhood of $[-1,1]$. A detailed analysis is given in [37]. As an example, Figure 11 plots the errors in approximation of the Runge function

$$f(\mathbf{x}) = \frac{1}{1 + 10(x_1^2 + \cdots + x_d^2)} \tag{7.3}$$

with $d = 8$ with $m = 3, \ldots, 20$ points in each direction (the value of the integral is $I \approx 10.359057801$). The transformed rule achieves 6- or 8-digit accuracy with ten times fewer sample points than the Gauss rule.

8. Discussion.

Many researchers in high-dimensional problems know that their methods take advantage of special properties of the functions being approximated,

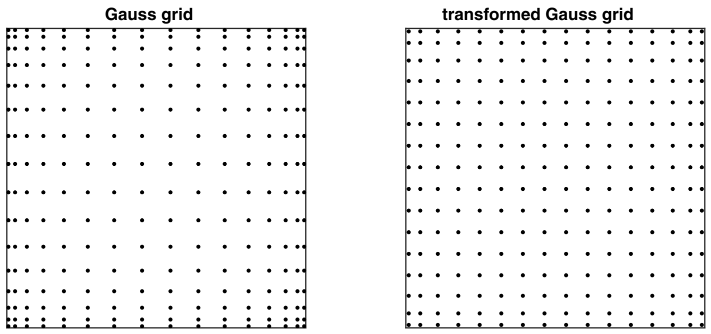


Fig. 10 16×16 *Gauss and transformed Gauss grids in* $[-1,1]^2$. *The latter, derived from the conformal map* (7.1) *of a Bernstein ellipse to a region with straighter sides, has approximately* 30% *better resolution in each dimension* [37].

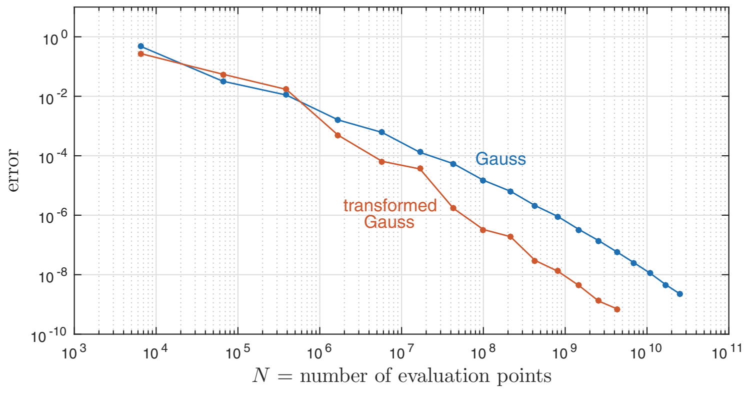


Fig. 11 *Errors in Gauss and transformed Gauss quadrature for the d-dimensional Runge function* (7.3) *with* $d = 8$. *The transformed formula requires an order of magnitude fewer sample points. In both cases the error is* $O(\exp(-Cn^{1/d}))$ *with* $C > 0$, *but the constant* C *for the transformed formula is larger.*

such as alignment with axes, but this matter is rarely emphasized in print. Indeed, it is startling how many articles contain remarks to the effect that because of the curse of dimensionality, we will now present a method to compress the data—as if the need for compression were synonymous with its possibility! For arbitrary functions, compressing away the curse of dimensionality is probably impossible. The surprise is that in practice it is so often possible after all. By paying more attention to the properties of multivariate functions that lead to this situation, perhaps we can better understand these schemes and improve them.

An interesting contrast is with the subject of Krylov subspace iterations for solving linear systems of equations, which begins with the conjugate gradient method [40]. Here, it is universally recognized that iteration achieves nothing for arbitrary matrices, and the whole subject is explicitly focused on how to exploit special matrix properties and enhance them with preconditioners [69].

In talking with colleagues about this work, I have been fascinated to hear of the many ways in which matters of high-dimensional geometry, and specifically the shape and volume of the hypercube, enter into mathematical and algorithmic research. For example, another field where high-dimensional problems arise is nonlinear optimization. Nick Gould has pointed out to me that the differences between the unit ball in the ∞-norm (the hypercube) and the 2-norm (the hyperball) have pervasive consequences in optimization, both theoretical and practical. Trust-region algorithms face the basic choice of whether to use balls or cubes [16], with advantages on both sides, and the problem of nonconvex quadratic programming is known to be NP-hard in a hypercube but of only polynomial complexity in a hyperball [68]. High-dimensionality is particularly challenging for global optimization [27, 42], where the starting idea, dividing a cube into many small subcubes, has the exponential complexity of tensor product grids, but alternative algorithms, and the theorems that support them, may not be rotationally invariant.

Virtually all the volume of a high-dimensional region is near the boundary, and virtually all the volume of a high-dimensional hypercube is outside the inscribed hypersphere. In this article we have seen the influences of these facts on numerical algorithms. Perhaps a general observation to be made is that the choice to set one's problem in a square or cube, which seems just a matter of convenience in two or three dimensions, may become a matter of exponential significance as $d \to \infty$. If the essence of the problem has nothing to do with the hypercube, then one may pay a heavy price for this convenience. Conversely, many problems live naturally in a hypercube—one may think, for example, of constraints of the form $a \le x_j \le b$ in optimization or cubature—and this may entail an anisotropy that is the very reason why fast algorithms are effective.

A related subject that has attracted much attention among purer mathematicians is the phenomenon of concentration of measure [48]. The theme here is that in a high-dimensional domain such as a hypercube or a hyperball, any set that includes a sizable fraction of the total volume (measure) must come close to almost every point in the region. If you transform a function smoothly to make it zero on the boundary of the hypercube, for example, so that a periodic lattice rule can be applied as discussed in section 4, then you have made it close to zero nearly everywhere inside the hypercube—so the cost in the number of grid points needed to resolve the part of the function you care about will probably be exponential.

We close with a passage from the 1961 book by Richard Bellman [4, p. 94], who coined the famous phrase:

> In view of all that we have said in the foregoing sections, the many obstacles we appear to have surmounted, what casts the pall over our victory celebration? It is the curse of dimensionality, a malediction that has plagued the scientist from earliest days.

Note Added in Proof. Since this article was written, a theorem has been establishd confirming the observation of section 6. See L. N. Trefethen, "Multivariate Polynomial Approximation in the Hypercube," Proc. Amer. Math. Soc., to appear.

Acknowledgments. My interest in multidimensional problems was initiated by stimulating discussions of lattice rules with Ian Sloan and Harald Niederreiter during a sabbatical visit to the University of New South Wales in 1990, and I became personally involved as I worked with Alex Townsend and then Behnam Hashemi here at Oxford to extend Chebfun to two and three dimensions. Most recently I learned of the related cubature problems through discussions with Terry Lyons and Maria Tchernychova, also at Oxford. I thank all of these people for helpful and interesting conversations. Many others have helped me by commenting on drafts of the present paper or in other discussions, including Jared Aurentz (who suggested the term Euclidean degree), Lehel Banjai, Harold Boas, Len Bos, John Burkhardt, Ronald Cools, Carl De Boor, Stefano De Marchi, Sergey Dolgov, Alan Genz, Saman Ghili, Mike Giles, Nick Gould, Nick Hale, Boris Khoromskij, Daniel Kressner, Randy LeVeque, Scott Moe, Hadrien Montanelli, Dirk Nuyens, Asbjørn Riseth, Ian Sloan, Endre Süli, Steve Vavasis, and Zuoxin (Klaus) Wang.

REFERENCES

[1] K. I. Babenko, *Approximation of periodic functions of many variables by trigonometric polynomials*, Dokl. Akad. Nauk SSSR, 132 (1960), pp. 247–250. (Cited on p. 479)

[2] M. Bebendorf, *Hierarchical Matrices*, Springer, Berlin, 2008. (Cited on pp. 471, 474)

[3] M. Bebendorf, *Adaptive cross approximation of multivariate functions*, Constr. Approx., 34 (2011), pp. 149–179. (Cited on pp. 471, 474)

[4] R. Bellman, *Adaptive Control Processes: A Guided Tour*, Princeton University Press, Princeton, NJ, 1961. (Cited on p. 488)

[5] G. Beylkin, R. Coifman, and V. Rokhlin, *Fast wavelet transforms and numerical algorithms* I, Comm. Pure Appl. Math., 44 (1991), pp. 141–183. (Cited on p. 474)

[6] G. Beylkin and M. J. Mohlenkamp, *Algorithms for numerical analysis in high dimensions*, SIAM J. Sci. Comput., 26 (2005), pp. 2133–2159, https://doi.org/10.1137/040604959. (Cited on p. 471)

[7] L. Bos, M. Caliari, S. De Marchi, M. Vianello, and Y. Xu, *Bivariate Lagrange interpolation at the Padua points: The generating curve approach*, J. Approx. Theory, 143 (2006), pp. 15–25. (Cited on p. 480)

[8] L. Bos, S. De Marchi, and M. Vianello, *Trivariate polynomial approximation on Lissajous curves*, IMA J. Numer. Anal., 2016, drw013. (Cited on p. 480)

[9] J. Boyd, *Chebyshev and Fourier Spectral Methods*, 2nd ed. (revised), Dover, Mineola, NY, 2001. (Cited on p. 480)

[10] A. Brandt, *Multilevel computations of integral transforms and particle interactions with oscillatory kernels*, Comput. Phys. Commun., 65 (1991), pp. 24–38. (Cited on p. 474)

[11] H.-J. Bungartz and M. Griebel, *A note on the complexity of solving Poisson's equation for spaces of bounded mixed derivatives*, J. Complexity, 15 (1999), pp. 167–199. (Cited on p. 480)

[12] H.-J. Bungartz and M. Griebel, *Sparse grids*, Acta Numer., 13 (2004), pp. 147–269. (Cited on p. 479)

[13] R. E. Caflisch, *Monte Carlo and quasi-Monte Carlo methods*, Acta Numer., 7 (1998), pp. 1–49. (Cited on p. 476)

[14] A. P. Calderon and A. Zygmund, *On the existence of certain singular integrals*, Acta Math., 88 (1952), pp. 85–139. (Cited on p. 474)

[15] O. A. Carvajal, F. W. Chapman, and K. O. Geddes, *Hybrid symbolic-numeric integration in multiple dimensions via tensor-product series*, in Proceedings of ISSAC'05, M. Kauers, ed., ACM, New York, 2005, pp. 84–91. (Cited on p. 471)

[16] A. R Conn, N. I. M. Gould, and P. L. Toint, *Trust-Region Methods*, SIAM, Philadelphia, 2000. (Cited on p. 488)

[17] P. R. Conrad and Y. M. Marzouk, *Adaptive Smolyak pseudospectral approximations*, SIAM J. Sci. Comput., 35 (2013), pp. A2643–A2670, https://doi.org/10.1137/120890715. (Cited on pp. 479, 484)

[18] R. Cools, *Constructing cubature formulae: The science behind the art*, Acta Numer., 6 (1997), pp. 1–54. (Cited on pp. 475, 478, 480)

[19] R. Cools and K. Poppe, *Chebyshev lattices, a unifying framework for cubature with Chebyshev weight functions*, BIT, 51 (2011), pp. 275–288. (Cited on p. 480)

[20] P. G. Constantine, *Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies*, SIAM, Philadelphia, 2015. (Cited on p. 471)

[21] W. Dahmen, R. DeVore, L. Grasedyck, and E. Süli, *Tensor-sparsity of solutions to high-dimensional elliptic partial differential equations*, Found. Comput. Math., 2015, pp. 1–62. (Cited on pp. 470, 471, 474)

[22] J. Dick, F. Y. Kuo, Q. T. Le Gia, D. Nuyens, and C. Schwab, *Higher order QMC Petrov–Galerkin discretization for affine parametric operator equations with random field inputs*, SIAM J. Numer. Anal., 52 (2014), pp. 2676–2702, https://doi.org/10.1137/130943984. (Cited on p. 477)

[23] J. Dick, F. Y. Kuo, and I. H. Sloan, *High-dimensional integration: The quasi-Monte Carlo way*, Acta Numer., 22 (2013), pp. 133–288. (Cited on pp. 470, 476, 477, 478)

[24] J. Dick and F. Pillichshammer, *Digital Nets and Sequences: Discrepancy Theory and Quasi-Monte Carlo Integration*, Cambridge University Press, Cambridge, UK, 2010. (Cited on p. 478)

[25] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices* III: *Computing a compressed approximate matrix decomposition*, SIAM J. Comput., 36 (2006), pp. 184–206, https://doi.org/10.1137/S0097539704442702. (Cited on p. 471)

[26] T. A. Driscoll, N. Hale, and L. N. Trefethen, *Chebfun User's Guide*, Pafnuty Publications, Oxford, UK, 2014. See also www.chebfun.org. (Cited on p. 471)

[27] J. Fowkes, *Bayesian Numerical Analysis: Global Optimization and Other Applications*, D. Phil. thesis, Mathematical Institute, University of Oxford, UK, 2011. (Cited on p. 488)

[28] T. Goda and J. Dick, *Construction of interlaced scrambled polynomial lattice rules of arbitrary high order*, Found. Comput. Math., 15 (2015), pp. 1245–1278. (Cited on p. 478)

[29] S. A. Goreinov, I. V. Oseledets, D. V. Savostyanov, E. E. Tyrtyshnikov, and N. L. Zamarashkin, *How to find a good submatrix*, in Matrix Methods: Theory, Algorithms and Applications, World Scientific, Hackensack, NJ, 2010, pp. 247–256. (Cited on p. 471)

[30] A. A. Gorodetsky, S. Karaman, and Y. M. Marzouk, *Function-Train: A Continuous Analogue of the Tensor-Train Decomposition*, preprint, https://arxiv.org/abs/1510.09088, 2015. (Cited on pp. 471, 472)

[31] L. Grasedyck, D. Kressner, and C. Tobler, *A literature survey of low-rank tensor approximation techniques*, GAMM-Mitt., 36 (2013), pp. 53–78. (Cited on pp. 471, 472)

[32] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348. (Cited on p. 474)

[33] M. Griebel and J. Hamaekers, *Sparse grids for the Schrödinger equation*, ESAIM Math. Model. Numer. Anal., 41 (2007), pp. 215–247. (Cited on p. 484)

[34] M. Griebel and H. Harbrecht, *Approximation of bi-variate functions: Singular value decomposition versus sparse grids*, IMA J. Numer. Anal., 34 (2014), pp. 28–54. (Not cited)

[35] S. Haber, *Numerical evaluation of multiple integrals*, SIAM Rev., 12 (1970), pp. 481–526, https://doi.org/10.1137/1012102. (Cited on pp. 475, 480)

[36] W. Hackbusch, *Tensor Spaces and Numerical Tensor Calculus*, Springer, New York, 2012. (Cited on pp. 471, 472, 474)

[37] N. Hale and L. N. Trefethen, *New quadrature formulas from conformal maps*, SIAM J. Numer. Anal., 46 (2008), pp. 930–948. (Cited on pp. 485, 486, 487)

[38] N. Halko, P.-G. Martinnson, and J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Rev., 53 (2011), pp. 217–288, https://doi.org/10.1137/090771806. (Cited on p. 471)

[39] B. Hashemi and L. N. Trefethen, *Chebfun in three dimensions*, SIAM J. Sci. Comput., to appear, https://doi.org/10.1137/16M1083803. (Cited on p. 471)

[40] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Research Nat. Bur. Standards, 49 (1952), pp. 409–436. (Cited on p. 487)

[41] E. Hlawka, *Funktionen von beschränkter Variation in der Theorie der Gleichverteilung*, Ann. Mat. Pura Appl., 54 (1961), pp. 325–333. (Cited on p. 476)

[42] R. Horst and P. M. Pardalos, *Handbook of Global Optimization*, Springer, New York, 2013. (Cited on p. 488)

[43] B. N. Khoromskij, *Tensors-structured numerical methods in scientific computing: Survey on recent advances*, Chemometrics and Intelligent Laboratory Systems, 110 (2012), pp. 1–19. (Cited on pp. 471, 472)

[44] T. G. Kolda and B. W. Bader, *Tensor decompositions and applications*, SIAM Rev., 51 (2009), pp. 455–500, https://doi.org/10.1137/07070111X. (Cited on p. 472)

[45] N. M. Korobov, *The approximate computation of multiple integrals*, Dokl. Akad. Nauk SSSR, 124 (1959), pp. 1207–1210 (Russian). (Cited on p. 477)

[46] A. R. Krommer and C. W. Ueberhuber, *Computational Integration*, SIAM, Philadelphia, 1998. (Cited on pp. 475, 480)

[47] L. de Lathauwer, B. de Moor, and J. Vandewalle, *A multilinear singular value decomposition*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1253–1278, https://doi.org/10.1137/S0895479896305696. (Cited on p. 472)

[48] M. Ledoux, *The Concentration of Measure Phenomenon*, AMS, Providence, RI, 2001. (Cited on p. 488)

[49] J. C. Maxwell, *On approximate multiple integration between limits of summation*, Proc. Camb. Phil. Soc., 3 (1877), pp. 39–47. (Cited on p. 480)

[50] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*, SIAM, Philadelphia, 1992. (Cited on pp. 476, 477, 478)

[51] E. Novak and H. Woźniakowski, *Tractability of Multivariate Problems*, I: *Linear Information*, EMS Tracts Math. 6, Zurich, 2008. (Cited on p. 477)

[52] E. Novak and H. Woźniakowski, *Approximation of infinitely differentiable multivariate functions is intractable*, J. Complexity, 25 (2009), pp. 398–404. (Cited on p. 477)

[53] I. V. Oseledets, *Tensor-train decomposition*, SIAM J. Sci. Comput., 33 (2011), pp. 2295–2317, https://doi.org/10.1137/090752286. (Cited on p. 472)

[54] I. V. Oseledets and E. E. Tyrtyshnikov, *Breaking the curse of dimensionality, or how to use the SVD in many dimensions*, SIAM J. Sci. Comput., 31 (2009), pp. 3744–3759, https://doi.org/10.1137/090748330. (Cited on p. 471)

[55] S. H. Paskov and J. Traub, *Faster evaluation of financial derivatives*, J. Portfolio Management, 22 (1995), pp. 113–120. (Cited on p. 477)

[56] T. Penzl, *Eigenvalue decay bounds for solutions of Lyapunov equations: The symmetric case*, Systems Control Lett., 40 (2000), pp. 139–144. (Cited on p. 474)

[57] H.-J. Schmeisser and W. Sickel, *Spaces of functions of mixed smoothness and approximation from hyperbolic crosses*, J. Approx. Theory, 128 (2004), pp. 115–150. (Cited on p. 479)

[58] S. A. Smolyak, *Quadrature and interpolation formulas for tensor products of certain classes of functions*, Dokl. Akad. Nauk SSSR, 4 (1963), pp. 240–243. (Cited on p. 479)

[59] A. H. Stroud, *Approximate Calculation of Multiple Integrals*, Prentice-Hall, Englewood Cliffs, NJ, 1971. (Cited on p. 480)

[60] V. Tchakaloff, *Formules de cubature mécanique à coefficients non négatifs*, Bull. Sci. Math., 81 (1957), pp. 123–134. (Cited on p. 481)

[61] M. Tchernychova, *"Carathéodory" Cubature Measures*, D. Phil. thesis, Mathematical Institute, University of Oxford, UK, 2015. (Cited on p. 481)

[62] V. N. Temlyakov, *Approximation of Functions with Bounded Mixed Derivative*, Proc. Steklov Inst. Math. 178, AMS, Providence, RI, 1989. (Cited on p. 480)

[63] A. Townsend and L. N. Trefethen, *Gaussian elimination as an iterative algorithm*, SIAM News, March 2013. (Cited on pp. 472, 473)

[64] A. Townsend and L. N. Trefethen, *An extension of Chebfun to two dimensions*, SIAM J. Sci. Comput., 35 (2013), pp. C495–C518, https://doi.org/10.1137/130908002. (Cited on pp. 471, 473, 474)

[65] A. Townsend and L. N. Trefethen, *Continuous analogues of matrix factorizations*, Proc. R. Soc. A, 471 (2015), 20140585. (Cited on p. 472)

[66] L. N. Trefethen, *Approximation Theory and Approximation Practice*, SIAM, Philadelphia, 2013. (Cited on pp. 475, 480, 481, 485, 486)

[67] E. E. Tyrtyshnikov, *Tensor approximations of matrices generated by asymptotically smooth functions*, Sb. Math., 194 (2003), pp. 941–954. (Cited on p. 474)

[68] S. A Vavasis, *Nonlinear Optimization: Complexity Issues*, Oxford University Press, New York, 1991. (Cited on p. 488)

[69] A. J. Wathen, *Preconditioning*, Acta Numer., 24 (2015), pp. 329–376. (Cited on p. 487)

[70] S. Xiang and F. Bornemann, *On the convergence rates of Gauss and Clenshaw–Curtis quadrature for functions of limited regularity*, SIAM J. Numer. Anal., 50 (2012), pp. 2581–2587, https://doi.org/10.1137/120869845. (Cited on p. 475)

[71] H. Yserentant, *On the multi-level splitting of finite element spaces*, Numer. Math., 49 (1986), pp. 379–412. (Cited on p. 478)

[72] C. Zenger, *Sparse grids*, in Parallel Algorithms for Partial Differential Equations, W. Hackbusch, ed., Vieweg, Braunschweig, 1991, pp. 241–251. (Cited on p. 479)