Explicit Cost Bounds of Algorithms for Multivariate Tensor Product Problems

Columbia University Computer Science Department Report CUCS-020-96

Grzegorz W. Wasilkowski and Henryk Woźniakowski *

March, 1994

Abstract

We study multivariate tensor product problems in the worst case and average case settings. They are defined on functions of d variables. For arbitrary d, we provide explicit upper bounds on the costs of algorithms which compute an ε-approximation to the solution. The cost bounds are of the form

$$(c(d)+2)\,\beta_1\left(\beta_2\,+\,\beta_3\,\frac{\ln\,1/\varepsilon}{d-1}\right)^{\beta_4(d-1)}\,\left(\frac{1}{\varepsilon}\right)^{\beta_5}.$$

Here $c(d)$ is the cost of one function evaluation (or one linear functional evaluation), and β_i's do not depend on d; they are determined by the properties of the problem for $d=1$. For certain tensor product problems, these cost bounds do not exceed $c(d)\,K\,\varepsilon^{-p}$ for some numbers K and p, both independent of d.

We apply these general estimates to certain integration and approximation problems in the worst and average case settings.

We also obtain an upper bound, which is independent of d, for the number, $n(\varepsilon,d)$, of points for which discrepancy (with unequal weights) is at most ε,

$$n(\varepsilon,d)\,\leq\,7.26\,\varepsilon^{-2.454},\quad\forall\,d,\varepsilon\leq 1.$$

*The first author was partially supported by the the National Science Foundation, and the second by the National Science Foundation and the Air Force Office of Scientific Research.

1 Introduction

We study linear multivariate problems in the worst case and average case settings. They are defined as computing an ε-approximation to a linear operator acting on functions f of d variables. An ε-approximation is obtained by an algorithm which combines information about the function f. This information is provided by a number of continuous linear functionals $L(f)$. The functional L can be a function evaluation, $L(f) = f(x)$ for some x, or an arbitrary continuous linear functional. The information given by function values is of special interest since that is usually what we can only compute in practice. The optimal choice of sample points and algorithms is a challenging problem for many multivariate problems. We add that the optimal choice of arbitrary continuous linear functionals and algorithms is known, see Chapters 4 and 6 of [23].

We assume that the cost of computing $L(f)$ is $c(d) \geq 1$. We also assume that we can perform combinatory operations such as arithmetic operations and comparisons at unit cost. The cost of an algorithm that computes an ε-approximation is defined as the cost of computing all functionals and combinatory operations. The computational complexity is defined as the minimal cost of computing an ε-approximation in the given setting.

We stress that we are interested in both large and modest d. The reason is that in some applications d is large whereas in many other applications d is modest, say between 2 and 12. Examples of problems with large d include some finance and physics problems. For finance problems dealing with a pool of 30-year mortgages, we usually have $d = 360$ (30 years times 12 months). For physics problems dealing with path integrals, we typically approximate the path integral by finite dimensional integrals. Then d is the dimension of the domain of the approximation, and can be arbitrarily large. There are numerous examples of problems with modest d such as integration, approximation, and the solution of partial differential equations which involve a small number of variables.

Efficient solution of multivariate problems is especially important for large d. However, even a small improvement for modest d can be significant since in some applications, thousands of problems with modest d have to be computed. Therefore we would like to find the best possible solution for both large and modest d.

In this paper we study tensor product problems. They are fully determined by the one dimensional case, $d = 1$, specified by a linear operator $S_1 : F_1 \to G_1$. In the worst case setting, we consider elements from the unit ball in F_1 with both F_1 and G_1 Hilbert spaces, whereas in the average case setting we consider elements from the whole space F_1 with F_1 a Banach space equipped with a Gaussian measure μ_1 and G_1 a Hilbert space.

The algorithms studied in this paper were presented by Smolyak [19] who studied tensor product problems in the worst case setting. Antecedents of these algorithms may be found in [2]. Later, these algorithms have been used in many papers for specific problems; we only mention some recent papers [7, 13, 14, 15, 20, 21, 22, 26, 30]. In all these papers, only asymptotic cost (or error) bounds are provided, i.e., ε is assumed to be sufficiently small whereas d is arbitrary but fixed. In this paper we will obtain explicit cost bounds for all d and ε.

The essence of these algorithms is that it is enough to know how to solve the tensor product problem for $d = 1$ efficiently. Then the algorithms for arbitrary d are fully determined in terms of the algorithms for $d = 1$. The one dimensional algorithms may use function values or, more generally, arbitrary linear functionals. As already mentioned, the choice of function values is especially interesting since for arbitrary linear functionals we know how to solve multivariate problems. Furthermore, the algorithms for $d = 1$ do not have to be optimal, although a poor choice makes the cost of the algorithms larger for arbitrary d.

The algorithms are *linear*. That is, they depend linearly on the information. This property makes their implementation easier. In fact, the weights of the algorithm for $d \geq 2$ are given by linear combinations of the corresponding tensor products weights of the one dimensional algorithms.

The algorithms use specially chosen sample points (or linear functionals). Even if we use equally-spaced sample points for $d = 1$ then the position of sample points in d dimensions is very different from grid points. In fact, it is known that grid points are a very poor choice of sample points for tensor product problems, see e.g., [12, 33]. Information used by the algorithms is called *hyperbolic cross* information and have been successfully applied for a number of problems, see Section 3 for details and references.

We summarize the result of Smolyak [19] more precisely. He studied the worst case setting only. Let $A_\varepsilon(d)$ denote an algorithm which computes an ε-approximation for the d dimensional case. For simplicity we assume that $\varepsilon \leq \varepsilon_0 < 1$. Then, under natural assumptions, he showed that the cost of $A_\varepsilon(d)$ is bounded from above by

$$K_d \, (c(d) + 2) \, \varepsilon^{-\beta_5} \; (\ln 1/\varepsilon)^{\beta_4 (d-1)} \tag{1}$$

for certain nonnegative β_4 and β_5 which depend on the particular tensor product problem. Here, K_d is an unknown function of d. Note that for small ε, the dominant factor of the cost, $\varepsilon^{-\beta_5}$, is independent of d. The dependence on d is through $\ln 1/\varepsilon$ and through the unknown K_d. Hence, asymptotically with respect to ε (i.e., when d is fixed and ε tends to zero) the bound seems very satisfactory. However, from a practical point of view, the dependence on d is also crucial. Therefore, without knowing the behavior of K_d, it is impossible to assess the quality of this bound especially when both ε and d may vary.

In this paper, we analyze the error and cost of algorithms with strong emphasis on the dependence on d. As far as we know the dependence on d has not been studied before. Our analysis is for both the worst case and average case settings. The average case setting is of interest per se. Moreover, for some problems, the worst case setting can be analyzed through an average case setting. In the average case setting we are able to get tighter bounds than in the worst case setting. Therefore, the average case setting is also of interest from the purely worst case setting point of view.

Among specific results, we prove that if the one dimensional algorithm combines the information in an optimal way and the information is *nested* (see Section 4 for the definition) then optimality of the algorithm is preserved for arbitrary d. This holds in both settings. However, optimality of information is not preserved. That is, if optimal information is used for $d = 1$, then the corresponding information for $d \geq 2$ needs not be optimal; however we do not lose much.

The main result of this paper is the explicit cost bound of the algorithm $A_\varepsilon(d)$ as a function of both ε and d. This bound holds under a natural assumption that the cost of the one dimensional algorithm $A_\varepsilon(1)$ is a polynomial in $1/\varepsilon$. To explain the cost bound, let $B = \|S_1\|_{\mathrm{set}}$ denote the norm of the one dimensional operator S_1. The norm $\|\cdot\|_{\mathrm{set}}$ depends on the setting. In the worst case setting it is the usual operator norm, whereas in the average case setting it is defined as the variance of $\|S(f)\|_{G_1}$, i.e., $\left(\int_{F_1} \|S(f)\|^2_{G_1} \mu_1(df)\right)^{1/2}$. For $d \geq 2$, the norm of the corresponding operator is B^d. Thus, the problem is trivial for $\varepsilon \geq B^d$ since $A_\varepsilon(d) = 0$ solves the problem with error not exceeding ε and zero cost. Hence, it is enough to consider only $\varepsilon < B^d$. Then the cost bound for the algorithm $A_\varepsilon(d)$ is of the form

$$\mathrm{cost}(A_\varepsilon(d)) \,\leq\, (c(d)+2)\;\beta_1 \left(\beta_2 \,+\, \beta_3\,\frac{\ln\, 1/\varepsilon}{d-1}\right)^{\beta_4(d-1)} \left(\frac{1}{\varepsilon}\right)^{\beta_5}, \tag{2}$$

for certain β_i which are fully determined by the quality of the one dimensional algorithms and the setting.

Note that from (2) we can infer an estimate of K_d in (1). Observe also that

$$C_d \,=\, \limsup_{\varepsilon\to 0^+}\; \mathrm{cost}(A_\varepsilon(d)) \Big/ \left((c(d)+2)(\ln\,1/\varepsilon)^{\beta_4(d-1)}(1/\varepsilon)^{\beta_5}\right) \,\leq\, \beta_1 \left(\frac{\beta_3}{d-1}\right)^{\beta_4(d-1)}.$$

Hence, the asymptotic constant C_d goes to zero super-exponentially with d. Equivalently, for $\varepsilon \leq \varepsilon_0$ with a sufficiently small ε_0, the factor K_d is super-exponentially small in d.

For fixed ε and varying d, or for both ε and d varying, it may, however, happen that the bound (2) depends exponentially on d. It is not clear whether this is a bad property of the algorithm $A_\varepsilon(d)$ or is an indication that the tensor product problem is hard.

To address this issue we raise the question of optimality of the cost bound (2), i.e., how much this bound differs from the complexity $\mathrm{comp}(\varepsilon, d)$. (As already mentioned, the complexity is the minimal cost over all choices of information and algorithms which allow to compute an ε-approximation). Unfortunately, sharp bounds on the complexity $\mathrm{comp}(\varepsilon, d)$ are only known for some specific problems and usually only if arbitrary linear functionals are allowed. If function values only are allowed then we usually know the asymptotic behavior of the complexity

$$\mathrm{comp}(\varepsilon, d) \;=\; K_d^*\,(c(d)+2)\,\varepsilon^{-\beta_5^*}(\ln 1/\varepsilon)^{\beta_4^*(d-1)}\,(1+o(1)) \qquad \text{as} \qquad \varepsilon \to 0 \tag{3}$$

with known β_4^* and β_5^*, but unknown K_d^*, see e.g., [13, 15, 16, 20, 22, 29, 30]. The limited knowledge of the behavior of the complexity makes the question of optimality of the cost bound (2) even harder. A partial answer is provided by comparing the exponents β_5, β_4 with β_5^*, β_4^*. If we use optimal information and algorithms for $d = 1$ then the exponent β_5 in (2) matches the exponent β_5^* in the complexity. The exponent β_4 is usually larger than β_4^*. This means that (2) is not optimal for small ε. However, it is usually necessary to take a pathologically small ε, to observe a significant difference in the cost due to the increase of the power of $\ln 1/\varepsilon$.

Without knowing K_d^* in (3) we are not able to verify optimality of the cost bound (2). To alleviate this difficulty, and yet to get some insight concerning the optimality of (2) and the dependence on d, we follow the approach of tractability and strong tractability for linear multivariate problems introduced in [31, 32]. Tractability means that the complexity is bounded by $c(d)\,K(d,\varepsilon)$ where $K(d,\varepsilon)$ is a polynomial in d and $1/\varepsilon$. Strong tractability means that $K(d,\varepsilon)$ is a polynomial in $1/\varepsilon$ and does not depend on d. An algorithm that computes an ε-approximation with cost bounded in such a way is said to be a polynomial-time (or strongly polynomial-time) algorithm.[1] For tensor product problems for which S_1 is not a linear functional, i.e., $\dim S_1(F_1) \geq 2$, the concepts of tractability and strong tractability coincide, see Section 7 and (i) of the Appendix.

We stress that the cost bound (2) holds for arbitrary tensor product problems which may or may not be strongly tractable. If the tensor product problem is *not* strongly tractable then the cost of *any* algorithm which computes an ε-approximation cannot depend polynomially on d. In our case, this means that the dependence on d in (2) is exponential. We stress that for many applications d is modest, say $d \leq 12$. Then we may afford to use the algorithm $A_\varepsilon(d)$ even for small ε.

Suppose now that the tensor product problem is strongly tractable. This is the case if the following two conditions holds. The first condition is that there exists a one dimensional

[1] Polynomial-time is a property of an algorithm and the information used by it. For simplicity, we only mention the algorithm.

algorithm, $A_\varepsilon(1)$, whose cost is polynomial in $1/\varepsilon$. The second condition is

$$\|S_1\|_{\text{set}} < 1. \tag{4}$$

We add that the first condition is also necessary for strong tractability. The second condition is necessary if S_1 is not a linear functional. If S_1 is a linear functional then $\|S_1\|_{\text{set}} \geq 1$ may or may not yield strong tractability depending on a specific form of S_1. The second condition is also necessary for a linear functional S_1 which cannot be represented as a finite combination of function evaluations, see (ii) and (iii) of the Appendix.

Assuming these two conditions, we show that $A_\varepsilon(d)$ is a *strongly polynomial-time algorithm*,

$$\text{cost}\,(A_\varepsilon(d)) \leq K\ (c(d)+2)\ \varepsilon^{-p}, \quad \forall\, d, \varepsilon \leq 1, \tag{5}$$

for some nonnegative numbers K and p. This means that the dependence on d is only through the cost $c(d)$. This holds even if only function values are used. Previously, such algorithms were known to exist but the proof was non-constructive, see [31, 32].

We stress that the cost bound (2) is usually much smaller than (5). To obtain a bound of the form (5) we must permit the worst relation between d and $1/\varepsilon$. Such a relation between d and $1/\varepsilon$ does not usually hold in computational practice. Hence, in most practical cases the cost bound (2) is much smaller than the bound (5). Clearly, for small ε we loose a power of $\varepsilon^{-(p-\beta_5)}$. If B is close to one then $p-\beta_5$ may be very large.

We now comment on the assumption (4) which may be viewed as the normalization of the tensor product problem. Let $B = \|S_1\|_{\text{set}} < 1$. For $d \geq 2$, the norm of the corresponding operator is B^d, which is exponentially small for large d. As already mentioned, for $\varepsilon \geq B^d$ the problem is trivial since it can be solved with zero cost, whereas $\varepsilon < B^d$ seems to require an unusually high precision which may not be needed in computational practice. This may be interpreted as stating that only trivial or badly normalized tensor product problems are strongly tractable. Still, as we shall see, for certain important problems the assumption $B < 1$ holds.

One may feel uneasy accepting the assumption (4) and prefer to normalize the tensor product problem by insisting that $\|S_1\|_{\text{set}} = 1$. Then some counter-intuitive things happen, see [32] and (iv) of the Appendix. For instance, this normalization criterion contradicts a natural property that the sum of two strongly tractable problems is strongly tractable.

Furthermore, it is not necessarily clear why we should insist on $\|S_1\|_{\text{set}} = 1$. For instance, consider the worst case setting with the unit ball in the Sobolev space F_1 of absolutely continuous functions defined over $[0,1]$, vanishing at zero and with the first derivative in $G_1 = L_2([0,1])$. The norm in F_1 is $(\int_0^1 |f'(t)|^2 dt)^{1/2}$. Let $S_1(f) = \alpha\, f$. Then

$$\|S_1\| = 2\,|\alpha|/\pi.$$

Hence, if we insist that $\|S_1\| = 1$ we must take $|\alpha| = \pi/2$, although it seems that the most natural choice of α is 1. Obviously, the norm $\|S_1\|$ depends also on the length of interval, the bound of the first derivative and so on. One can thus play with the choices of different problem parameters to satisfy or not to satisfy the assumption (4), see Section 8. We prefer not to choose sides in this selection of parameters.

We add that if one normalizes the problem by taking $\|S_1\| = 1$ then the problem becomes intractable and the complexity depends roughly on

$$d^{\beta_6 \ln 1/\varepsilon}$$

for some positive β_6, see [32]. Still, if ε is not too small, this is acceptable even for large d. Obviously, we can always use the cost bound (2) which is reasonable for modest d.

We illustrate the analysis of this paper by a number of applications. We discuss multivariate integration and approximation in the worst case and average case settings. In particular, we study the average case setting for multivariate integration of continuous functions defined over the d dimensional cube $[0, \beta]^d$ and equipped with the classical Wiener sheet measure. In this case, we have $\|S_1\|_{\text{avg}} = \beta^{3/2}/\sqrt{3}$, and both conditions for strong tractability hold iff $\beta < 3^{1/3} = 1.4422\ldots$.

Another interesting application is discrepancy (with unequal weights) in the L_2 norm. This problem is defined as finding n points in the d dimensional unit cube which approximate the volumes of rectangles with minimal error, see [11] for the precise definition and basic properties. Discrepancy has been extensively studied in number theory, see e.g., [3, 11, 17, 18]. It has recently been applied in computer science, see [4, 5] and the references given there. Discrepancy is related to multivariate integration. Indeed, discrepancy is an upper bound on the worst case integration error of functions whose variation in the sense of Hardy and Krause is at most one, see [11]. It is also known, see [29], that discrepancy (with optimally chosen weights) is equal to the minimal average case integration error of continuous functions defined over the d dimensional unit cube and equipped with the Wiener sheet measure. Hence, although discrepancy seems to have nothing in common with tensor products and strong tractability of multivariate problems, we may apply the bounds of the average case integration to discrepancy. More precisely, let $n(\varepsilon, d)$ denote the number of points in the d dimensional cube for which discrepancy is at most ε. Then

$$n(\varepsilon, d) = 0 \quad \text{for } \varepsilon \geq 3^{-d/2}.$$

Indeed, $n(\varepsilon, d) = 0$ means that no points are used in approximating the volumes of rectangles and discrepancy is $\left(\int_{[0,1]^d} (t_1 t_2 \cdots t_d)^2 dt\right)^{1/2} = 3^{-d/2}$, as claimed. Hence, for any n points in the d dimensional cube, discrepancy with optimally chosen weights is no greater than $3^{-d/2}$.

For applications with large d, we have thus an exponentially small initial value of discrepancy, and $\varepsilon < 3^{-d/2}$ does not have to hold.[2] For applications with modest d, the error bound ε is usually much less than $3^{-d/2}$. Then

$$n(\varepsilon,d) \;\le\; 3.304\,\left(1.77959 \,+\, 2.714\,\frac{-1.12167+\ln 1/\varepsilon}{d-1}\right)^{1.5(d-1)}\,\frac{1}{\varepsilon}.$$

We also show that

$$n(\varepsilon,d) \;\le\; \gamma_d\,\varepsilon^{-2.454} \;\le\; 7.26\,\varepsilon^{-2.454}, \quad \forall\, d, \varepsilon \le 1,$$

where $\gamma_1 = 0.39$, $\gamma_2 = 1.76$, $\gamma_3 = 4.76$, $\gamma_4 = 7.26$ and, starting from $d = 4$, γ_d decreases monotonically to zero. Equivalently, this means that the discrepancy $\mathrm{DISC}_n(d)$ of these n points in the d dimensional case satisfies

$$\mathrm{DISC}_n(d) \;\le\; \frac{2.244}{n^{0.408}}, \quad \forall\, d.$$

See Section 8.3 for the definition of the points and weights satisfying these bounds. We stress, however, that these bounds can be applied only if unequal weights are permitted.

We now comment on the above bounds on discrepancy and the average case integration. It is known that the minimal number of points with discrepancy or average case integration error at most ε is of order ε^{-2}. The proof is, however, not constructive. We mention in Section 8.2 that the exponent which we obtained, $p = 2.454$, can be lowered by choosing different parameters for $d = 1$. We tried a number of such parameters but we always obtained exponents greater than two. This indicates that either the algorithms studied in this paper are not optimal, or that analysis of these algorithms can be improved. It is a challenging problem to find points for which the exponent p is at most two. Of course, it would be desirable to find the smallest such exponent.

We believe that there are many possible ways to improve the algorithm $A_\varepsilon(d)$ studied here. The algorithm depends on a number of parameters and in many cases we simplified quite complicated estimates by taking specific values of these parameters or by accepting some overestimates. In a number of remarks presented in the text we indicate possible generalizations which may be useful for future improvements.

We plan to write and test software implementing the algorithm $A_\varepsilon(d)$ for a number of tensor product problems including the problems mentioned in Section 8. We also plan to study rounding error properties of $A_\varepsilon(d)$. This seems especially needed since for the integration problem the weights of these algorithms for $d \ge 2$ may be of different signs even though the weights for $d = 1$ are all positive.

[2] This may suggest that discrepancy is not properly normalized.

2 Formulation of the Problem

In this section, we define a tensor product problem for a class of functions of d variables. We also define information and algorithms as well as their cost and errors for approximating the tensor product problems.

For $d = 1, 2, \ldots$, consider

$$S_d : F_d \to G_d,$$

where F_d is a real separable Banach space of functions $f : \mathcal{D}^d \to \mathbb{R}$, $\mathcal{D} \subset \mathbb{R}$, G_d is a real separable Hilbert space, and S_d is a continuous linear operator.

We assume that G_d is a tensor product,

$$G_d \;=\; G_1 \otimes G_1 \otimes \cdots \otimes G_1.$$

That is, for an orthonormal system $\{\eta_i\}$ of G_1, the space G_d consists of functions g such that

$$g(t_1, t_2, \ldots, t_d) \;=\; \sum_{i_1,\ldots,i_d=1}^{\infty} c_{i_1,\ldots,i_d}\, \eta_{i_1}(t_1) \cdots \eta_{i_d}(t_d), \quad \forall t_i \in \mathcal{D}, \tag{6}$$

whose real coefficients $c_{i_1,\ldots,i_d}$ satisfy $\sum c^2_{i_1,\ldots,i_d} < +\infty$. The inner product in G_d is defined for $g(t_1, \ldots, t_d) = g_1(t_1) \cdots g_d(t_d)$ and $h(t_1, \ldots, t_d) = h_1(t_1) \cdots h_d(t_d)$ with $g_i, h_i \in F_1$ as

$$\langle g, h \rangle_{G_d} \;=\; \prod_{j=1}^{d} \langle g_i, h_i \rangle_{G_1}.$$

Observe that $\{\eta_{i_1}(t_1)\eta_{i_2}(t_2) \cdots \eta_{i_d}(t_d)\}$ is an orthonormal system of G_d and the coefficients $c_{i_1,\ldots,i_d}$ in (6) are equal to $\langle g, \eta_{i_1} \cdots \eta_{i_d} \rangle_{G_d}$.

The specific assumptions concerning F_d depend on a particular setting and are given in Sections 2.1 and 2.2. In both cases we assume that

$$\bigotimes_{k=1}^{d} f_k \in F_d, \quad \forall\, f_k \in F_1.$$

Here $f = \bigotimes_{k=1}^{d} f_k$ is defined by $f(t_1, ..., t_d) = \prod_{k=1}^{d} f_k(t_k)$.

We assume that also S_d has a tensor product form, i.e.,

$$S_d \left(\bigotimes_{k=1}^{d} f_k \right) (t_1, \ldots, t_d) \;=\; \prod_{k=1}^{d} (S_1 f_k)(t_k), \qquad \forall\, f_k \in F_1.$$

We now explain how the elements $S_d(f)$ are approximated. Without loss of generality, we restrict ourselves to nonadaptive information and linear algorithms since for the settings considered in this paper, adaption and nonlinear algorithms are not essentially better, see [28, 23]. Hence, the element $S_d(f)$ is approximated by $A(f) = \phi(N(f))$, where the *information* about f,

$$N(f) = [L_1(f), \dots, L_n(f)],$$

consists of n values of continuous linear functionals L_i, and $\phi : \mathbb{R}^n \to G_d$ is a linear mapping. This results in linearity of A, i.e.,

$$A(f) = \sum_{i=1}^{n} g_i L_i(f), \qquad \text{for some} \quad g_i \in G_d. \tag{7}$$

We refer to such A as a *linear algorithm.*

The error of the algorithm A depends on the setting and is defined in Sections 2.1 and 2.2. The cost of A does not depend on the setting and is defined as follows. We assume that the cost of computing $L_i(f)$ equals $c(d)$ for any $f \in F_d$ and any L_i. We also assume that basic arithmetic operations on reals and multiplication and addition in G_d have a unit cost.

Assuming that the elements g_i can be precomputed, the cost of the algorithm A, $\mathrm{cost}(A)$, is bounded by

$$\mathrm{cost}(A) \le n\,(c(d) + 2) - 1.$$

As we shall see, precomputation of the elements $g_i \in G_d$ is usually easy since they depend only on the corresponding elements for $d = 1$.

2.1 Worst Case Setting

In the worst case setting, we additionally assume that F_1 is a real separable Hilbert space and that F_d is a corresponding tensor product Hilbert space,

$$F_d = F_1 \otimes \cdots \otimes F_1,$$

defined similarly as the space G_d. Under these assumptions, for arbitrary linear operators $T_k : F_1 \to G_1$ we have

$$\|T_1 \otimes T_2 \otimes \cdots \otimes T_d\| = \prod_{k=1}^{d} \|T_k\|. \tag{8}$$

In the worst case setting, the error of a linear algorithm A is given as

$$e(A) = \sup\{\|S_d(f) - A(f)\|_{G_d} \, : \, \|f\|_{F_d} \le 1\}.$$

Due to linearity of S_d and A, we have

$$e(A) \;=\; \|S_d - A\|$$

for the induced operator norm $\|\cdot\|$. Observe that for an arbitrary f, not necessarily from the unit ball,

$$\|S_d(f) - A(f)\|_{G_d} \;\le\; e(A)\|f\|_{F_d}.$$

2.2 Average Case Setting

In the average case setting, we assume that the Banach space F_1 is equipped with a zero-mean Gaussian measure μ_1, see [8, 25]. To preserve the tensor product structure, we assume that F_d is equipped with a tensor product measure μ_d, $\mu_d = \mu_1 \otimes \cdots \otimes \mu_1$. Hence, μ_d is a zero-mean Gaussian measure and

$$\mu_d(B_1 \otimes \cdots \otimes B_d) \;=\; \prod_{k=1}^{d} \mu_1(B_k),$$

for all Borel sets $B_k \subseteq F_1$.

For an arbitrary linear operator $T : F_d \to G_d$, define

$$\|T\|_{\mu_d} \;=\; \left(\int_{F_d} \|T(f)\|_{G_d}^2 \,\mu_d(df)\right)^{1/2}.$$

Then the property (8) carries over to the norm $\|\cdot\|_{\mu_d}$, i.e.,

$$\|T_1 \otimes T_2 \otimes \cdots \otimes T_d\|_{\mu_d} \;=\; \prod_{k=1}^{d} \|T_k\|_{\mu_1}, \quad \text{where} \quad T_k : F_1 \to G_1. \tag{9}$$

In the average case setting, the error of an algorithm A is given as

$$e(A) \;=\; \left(\int_{F_d} \|S_d(f) - A(f)\|_{G_d}^2 \,\mu_d(df)\right)^{1/2}.$$

Due to linearity of S_d and A, we have

$$e(A) \;=\; \|S_d - A\|_{\mu_d}.$$

3 Algorithms for Tensor Product Problems

We present algorithms for tensor product problems. As already indicated in the introduction, these algorithms were analyzed by Smolyak [19] in the worst case setting but without studying the dependence on d.

The essence of these algorithms is that, for arbitrary $d \geq 2$, they are given by certain combinations of tensor products of one dimensional ($d = 1$) algorithms.

Assume therefore that, for $d = 1$, we know linear algorithms (operators) U_i, $i \geq 1$, which approximate the problem $\{F_1, G_1, S_1\}$ such that $\|S_1 - U_i\| \to 0$ as $i \to +\infty$. We denote

$$\Delta_0 \;=\; U_0 \;=0, \qquad \Delta_i \;=\; U_i - U_{i-1}.$$

For $d > 1$, we approximate the tensor product problem $\{F_d, G_d, S_d\}$ by the algorithm

$$A(q,d) \;=\; \sum_{0 \leq i_1 + i_2 + \cdots + i_d \leq q} \Delta_{i_1} \otimes \cdots \otimes \Delta_{i_d}. \tag{10}$$

Hence, if $f(t_1, t_2, \ldots, t_d) = f_1(t_1)\, f_2(t_2) \cdots f_d(t_d)$ then

$$(A(q,d)\, f)\; (t_1, t_2, \ldots, t_d) \;=\; \sum_{0 \leq i_1 + i_2 + \cdots + i_d \leq q} (\Delta_{i_1} f_1)\; (t_1)\, (\Delta_{i_2} f_2)\; (t_2) \cdots (\Delta_{i_d} f_d)\; (t_d).$$

Here, q is a nonnegative integer. Observe that for $q < d$ one of the indices is zero, say $i_j = 0$, and $\Delta_{i_j} = 0$ implies that $A(q,d) = 0$. Therefore we assume that $q \geq d$.

To find an explicit form of the algorithm $A(q,d)$ we use the following notation. For $\vec{i} = [i_1, \ldots, i_d]$ with nonnegative integers i_j, let $|\vec{i}| = \sum_{k=1}^{d} i_k$. We also write $\vec{i} \geq \vec{j}$ if $i_k \geq j_k$ for all k. Furthermore, by $Q(q,d)$ we mean

$$Q(q,d) = \left\{ \vec{i} = [i_1, i_2, \ldots, i_d] :\; \vec{1} \leq \vec{i},\; |\,\vec{i}\,| \leq q \right\}, \quad \text{with } \vec{1} = [1, 1, \ldots, 1].$$

The cardinality of the set $Q(q,d)$ is equal to the binomial coefficient $\begin{pmatrix} q \\ d \end{pmatrix}$. We have

$$\begin{aligned} A(q,d) \;&=\; \sum_{\vec{i} \in Q(q,d)} \bigotimes_{k=1}^{d} \Delta_{i_k} \;=\; \sum_{\vec{i} \in Q(q-1,d-1)} \left(\bigotimes_{k=1}^{d-1} \Delta_{i_k} \right) \otimes \sum_{i_d = 1}^{q - |\vec{i}|} \Delta_{i_d} && (11) \\ &=\; \sum_{\vec{i} \in Q(q-1,d-1)} \left(\bigotimes_{k=1}^{d-1} \Delta_{i_k} \right) \otimes U_{q - |\vec{i}|}, && (12) \end{aligned}$$

since $\sum_{i=1}^m \Delta_i = U_m$ for any $m \geq 1$.

We now derive an explicit form of $A(q,d)$ in terms of U_{i_k}. Observe that

$$\bigotimes_{k=1}^{d} (U_{i_k} - U_{i_k-1}) \;=\; \sum_{\vec{\alpha}\,\in\,\{0,1\}^d} (-1)^{|\vec{\alpha}\,|} \;\bigotimes_{k=1}^{d} U_{i_k-\alpha_k}.$$

Note that $\bigotimes_{k=1}^d U_{j_k}$ appears in $A(q,d)$ for all indices $\vec{i}$ for which $i_k = j_k + \alpha_k$ with $\vec{\alpha} \in \{0,1\}^d$ and $|\vec{\alpha}\,| \leq q - |\vec{j}\,|$. Furthermore, the sign of $\bigotimes_{k=1}^d U_{j_k}$ in this case is $(-1)^{|\vec{\alpha}\,|}$. Let

$$b(i,d) \;=\; \sum_{\vec{\alpha}\,\in\,\{0,1\}^d,\, |\vec{\alpha}\,| \leq i} (-1)^{|\vec{\alpha}\,|}.$$

This and (11) yield

$$A(q,d) \;=\; \sum_{\vec{j}\in Q(q,d)} b(q - |\vec{j}\,|, d) \bigotimes_{k=1}^{d} U_{j_k}.$$

We now compute $b(i,d)$. Clearly, we can sum up with respect to $|\vec{\alpha}\,| = 0, 1, \ldots, d$. Since $|\vec{\alpha}\,| = j$ corresponds to $\binom{d}{j}$ terms, we have

$$b(i,d) \;=\; \sum_{j=0}^{\min\{i,d\}} \binom{d}{j} (-1)^j = (-1)^i \binom{d-1}{i}.$$

In particular, $b(i,d) = 0$ for $i \geq d$. This yields the explicit form of $A(q,d)$ which is summarized in the following Lemma.

Lemma 1

$$A(q,d) \;=\; \sum_{\vec{i}\,\in\,P(q,d)} (-1)^{q-|\vec{i}|} \binom{d-1}{q-|\vec{i}\,|} \bigotimes_{k=1}^{d} U_{i_k}, \tag{13}$$

where

$$P(q,d) \;=\; \left\{ \vec{i} \,:\, \vec{1} \,\leq\, \vec{i},\; q-d+1 \,\leq\, |\vec{i}\,| \,\leq\, q \right\}.$$

In particular, for

$$U_i(f) \;=\; \sum_{j=1}^{m_i} a_{i,j}\, L_{i,j}(f) \tag{14}$$

with elements $a_{i,j} \in G_1$ *and continuous linear functionals* $L_{i,j}$*, we have*

$$A(q,d)f \;=\; \sum_{\vec{i}\,\in\,P(q,d)} (-1)^{q-|\vec{i}\,|} \binom{d-1}{q-|\vec{i}\,|} \sum_{\vec{j}\,\le\,\vec{m}_{\vec{i}}} L_{\vec{i},\vec{j}}(f)\, g_{\vec{i},\vec{j}}, \tag{15}$$

where $L_{\vec{i},\vec{j}} = \bigotimes_{k=1}^{d} L_{i_k,j_k}$, $g_{\vec{i},\vec{j}} \;=\; \bigotimes_{k=1}^{d} a_{i_k,j_k}$, *and* $\vec{m}_{\vec{i}} = [m_{i_1}, \ldots, m_{i_d}]$. $\square$

We now comment on the information

$$N_{q,d}(f) \;=\; [L_{\vec{i},\vec{j}}(f) : \; \vec{i} \,\in\, P(q,d),\, \vec{j} \,\le\, \vec{m}_{\vec{i}}]$$

used by the algorithm $A(q,d)$. Assume that $m_i \le F^i$ for some F. Then $\vec{j} \le \vec{m}_{\vec{i}}$ yields

$$j_1 j_2 \cdots j_d \;\le\; m_{i_1} m_{i_2} \cdots m_{i_d} \;\le\; F^{|\vec{i}|} \;\le\; F^q. \tag{16}$$

Hence, the indices $\vec{j}$ satisfy the so called *hyperbolic* inequality (16). However, not every index $\vec{j}$ satisfying (16) must correspond to some $L_{\vec{i},\vec{j}}$. For example, take $d = F = 2$ and $m_i = F^i$ with $q = 5$. Then $\vec{j} = [5,5]$ is a counter-example.

Thus, the indices of the functionals of the information $N_{q,d}$ are a subset of the solution of the hyperbolic inequality (16). The solution set of (16) forms part of a hyperbolic cross. That is why the information $N_{q,d}$ is called *hyperbolic cross* information.

The study and power of such hyperbolic cross information has been initiated by Babenko [2] who studied approximation of periodic functions by polynomials that use Fourier coefficients whose indices are from a hyperbolic cross. There are many papers where the power of hyperbolic cross information has been studied for a number of problems in different settings. The reader is referred to [7, 13, 14, 15, 16, 19, 20, 21, 22, 26, 30].

4 Error Analysis

4.1 Worst Case

We analyze the error of the algorithm (10) in terms of its error for $d = 1$. The following assumptions will be used for $d = 1$:

$$\|S_1\| \;\le\; B, \tag{17}$$

$$\|S_1 - U_i\| \;\le\; C\,D^i, \qquad \forall\, i \ge 0, \tag{18}$$

$$\|\Delta_i\| \;=\; \|U_i - U_{i-1}\| \;\le\; E\,D^i, \qquad \forall\, i \ge 1. \tag{19}$$

Here, the constant B bounds a norm of the operator S_1. The constants C, D, and E describe how well U_i approximates S_1. Of course, only $D < 1$ is of interest.

For $i = 0$ in (18), we get $\|S_1\| \le C$. Therefore we can assume that $B \le C$. Similarly, letting $i = 1$ in (19), we get

$$\|S_1\| \;\le\; \|S_1 - U_1\| + \|U_1\| \;\le\; D\,(\,C + E\,).$$

Hence, we can assume that $B \le D(C + E)$. By the same argument, $C(D^{-1} - 1) \le E \le C(D^{-1} + 1)$. To avoid the trivial case, we also assume that $B > 0$. Then C, E, and D have to be positive.

We stress that, in general, we do not assume any optimality properties of linear algorithms U_i for $d = 1$. We also do not assume any relation between information used by successive U_i. In Subsection 4.1.1, we derive an upper bound on the error of $A(q, d)$ in this general case. In Subsection 4.1.2, we assume "nested" information and optimality of U_i. Under these assumptions we improve error bounds of $A(q, d)$ as well as conclude optimality of $A(q, d)$.

4.1.1 General Case

Recall that

$$e(A(q, d)) \;=\; \|S_d - A(q, d)\|,$$

is the error of the algorithm $A(q, d)$ in the d dimensional case. Similarly, $e(U_i) = \|S_1 - U_i\|$ is the error of U_i for the one dimensional case.

For $q < d$ we have $A(q, d) = 0$ and $e(A(q, d)) \;=\; \|S_d\| \;\le\; B^d$. For $q \ge d$ we present the following estimates.

Lemma 2 *If (17), (18), and (19) hold then for $q \ge d$ we have*

$$\begin{aligned} e(A(q,d)) \;&\le\; C\,B^{d-1}\,D^{q-d+1} \sum_{j=0}^{d-1} \left(\frac{E\,D}{B}\right)^j \binom{q-d+j}{j} \\ &\le\; C H^{d-1} \binom{q}{d-1} D^q \qquad \text{with} \qquad H = \max\{B/D, E\}. \end{aligned} \tag{20}$$

Proof: For $d = 1$, Lemma 2 coincides with (18). Assume by induction that Lemma 2 holds for d. Due to (12), we have

$$S_{d+1} - A(q+1, d+1) \;=\; S_{d+1} - \sum_{\vec{i} \in Q(q,d)} \left(\bigotimes_{k=1}^{d} \Delta_{i_k} \right) \otimes U_{q+1-|\vec{i}|}$$

$$
\begin{aligned}
&= S_{d+1} + \sum_{\vec{i}\in Q(q,d)} \left(\bigotimes_{k=1}^{d} \Delta_{i_k}\right) \otimes \left(S_1 - U_{q+1-|\vec{i}|}\right) - A(q,d)\otimes S_1 \\
&= \sum_{\vec{i}\in Q(q,d)} \left(\bigotimes_{k=1}^{d} \Delta_{i_k}\right) \otimes \left(S_1 - U_{q+1-|\vec{i}|}\right) + (S_d - A(q,d))\otimes S_1.
\end{aligned}
$$

Hence, due to (8), (17), (18), and (19),

$$
\begin{aligned}
e(A(q+1,d+1)) &\le \sum_{\vec{i}\in Q(q,d)} E^d D^{|\vec{i}|} C D^{q+1-|\vec{i}|} + Be(A(q,d)) \\
&= CE^d D^{q+1} \begin{pmatrix} q \\ d \end{pmatrix} + Be(A(q,d)).
\end{aligned}
$$

This and the inductive assumption complete the proof of the first inequality. Estimating $(ED/B)^j$ by $(\max\{1, ED/B\})^{d-1}$ and using the fact that $\sum_{j=0}^{d-1} \begin{pmatrix} q-d+j \\ j \end{pmatrix} = \begin{pmatrix} q \\ d-1 \end{pmatrix}$, we obtain the second inequality. □

4.1.2 Nested Information and Optimal Algorithms

In this subsection we assume that the algorithms

$$
U_i(f) = \sum_{j=1}^{m_i} a_{i,j} L_{i,j}(f)
$$

use *nested* information $N_i = [L_{i,1}, L_{i,2}, \ldots, L_{i,m_i}]$. That is,

$$
\{L_{i,1}, L_{i,2}, \ldots, L_{i,m_i}\} \subset \{L_{i+1,1}, L_{i+1,2}, \ldots, L_{i+1,m_{i+1}}\} \quad \forall\, i = 1, 2, \ldots .
$$

Since F_1 is now a Hilbert space, $L_{i,j} = \langle f, f_{i,j}\rangle$ for some element $f_{i,j}$ of F_1. Hence, there exists a sequence $\{f_i\}$ of F_1 such that

$$
N_i(f) = [\langle f, f_1\rangle, \langle f, f_2\rangle, \ldots, \langle f, f_{m_i}\rangle], \quad i = 1, 2, \ldots .
$$

We also assume that the algorithms U_i are optimal, i.e., they minimize the error among all algorithms that use the information N_i. It is well known that U_i is optimal if

$$
U_i = S_1 \mathcal{P}_i, \tag{21}
$$

where $\mathcal{P}_i$ is the orthogonal projection on the linear subspace $\text{span}\{f_j : j = 1, 2, \ldots, m_i\} = (\ker N_i)^\perp$. We show that (21) implies optimality of the algorithm $A(q,d)$ for any d.

Lemma 3 *For nested information N_i and optimal U_i of (21),*

$$A(q,d) = S_d \mathcal{P}(q,d),$$

where $\mathcal{P}(q,d)$ is the orthogonal projection on the linear subspace $(ker(N_{q,d}))^{\perp}$. Thus, in particular, $A(q,d)$ minimizes the error among all algorithms that use the same information $N_{q,d}$.

Proof: From (10) and (21), $A(q,d) = S_d R(q,d)$ for $R(q,d) = \sum_{\vec{i}\in Q(q,d)} \bigotimes_{k=1}^{d} R_{i_k}$ with $R_{i_k} = \mathcal{P}_{i_k} - \mathcal{P}_{i_k - 1}$.

Let h_i be such that $L_j(h_i) = \delta_{j,i}$. Then $R_l h_{j_k} = \delta_{l,j_k} h_{j_k}$. Hence, on the subspace spanned by the functions $h(t_1, t_2, \ldots, t_d) = \prod_{k=1}^{d} h_{j_k}(t_k)$, $\vec{j} \le \vec{m}_i$ and $\vec{i} \in Q(q,d)$, the operator $R(q,d)$ is the identity. Moreover, $R(q,d) = 0$ for the orthogonal complement of this subspace. This proves that $R(q,d) = \mathcal{P}(q,d)$. As for $d = 1$, it is known that this form of $A(q,d)$ yields the minimal error. □

Remark 1 The projection form of an algorithm $A(q,d)$ implies the minimal error also if the range space of S_d is not necessarily a tensor product nor a Hilbert space. In fact, the projection form implies additional error properties such as minimizing all local errors, see, e.g., [10, 24, 23]. This is why such algorithms are sometimes referred to as *central* or *strongly optimal.* □

We now improve the estimate of Lemma 2 when the following additional assumption is made. Suppose that for every i, the information N_i used by U_i is optimal among all information using m_i arbitrary linear continuous functionals. It is well known that N_i is then of the form

$$N_i(f) = \left[\langle f, \eta_1\rangle_{F_1}, \ldots, \langle f, \eta_{m_i}\rangle_{F_1}\right], \tag{22}$$

where $\{\eta_j\}_{j=1}^{\infty}$ are the orthonormal eigenelements of $W_1 = (S_1^* S_1)^{1/2} : F_1 \to F_1$ and the corresponding eigenvalues λ_j are decreasing. That is,

$$W_1 \eta_j \;=\; \lambda_j\, \eta_j \qquad \text{and} \qquad \lambda_j \;\ge\; \lambda_{j+1} \;\ge\; 0.$$

Obviously, such optimal information is also nested. It is known that

$$e(U_i) \;=\; \|S_1 - U_i\| \;=\; \lambda_{m_i+1}.$$

For $d \ge 2$, the algorithm $A(q,d)$ uses information consisting of eigenelements of $W_d = (S_d^* S_d)^{1/2}$. It is also known that

$$e(A(q,d)) = \|S_d - A(q,d)\| \;=\; \lambda(q,d), \tag{23}$$

where $\lambda(q,d)$ is the largest eigenvalue of W_d whose eigenelement $\eta_{q,d}$ is orthogonal to $(\ker N_{q,d})^\perp$, i.e., $N_{q,d}\eta_{q,d} = 0$. We now find a more explicit form on the error of $A(q,d)$.

Lemma 4 *For optimal information with N_i of (22) and optimal U_i of (21), we have*

$$e(A(q,d)) \;=\; \max\left\{ \prod_{k=1}^{d} \lambda_{m_{i_k-1}+1} \;:\; |\vec{i}| = q+1 \right\}. \tag{24}$$

If, additionally, (18) holds then

$$e(A(q,d)) \;\le\; C^d\, D^{q-d+1}. \tag{25}$$

Proof: Clearly, $\eta_{j_1,j_2,\ldots,j_d}(t_1,t_2,\ldots,t_d) = \eta_{j_1}(t_1)\eta_{j_2}(t_2)\cdots\eta_{j_d}(t_d)$ is an eigenelement of W_d and its corresponding eigenvalue is $\lambda_{j_1,j_2,\ldots,j_d} = \lambda_{j_1}\lambda_{j_2}\cdots\lambda_{j_d}$. Let $N_{q,d}(\eta_{j_1,j_2,\ldots,j_d}) = 0$. Define the smallest i_k such that $j_k \le m_{i_k}$ for $k = 1,2,\ldots,d$. Then $|\vec{i}| > q$. Let $j_k = m_{i_k-1} + p_k$ with $p_k \ge 1$. Then

$$\lambda_{j_1,j_2,\ldots,j_d} \;\le\; \lambda_{m_{i_1}-1+1}\lambda_{m_{i_2}-1+1}\cdots\lambda_{m_{i_d}-1+1}.$$

Note that $\eta_{m_{i_1}-1+1,\ldots,m_{i_d}-1+1}$ is also orthogonal to $(\ker N_{q,d})^\perp$ since $m_{i_k-1}+1 \le m_{i_k}$ and $|\vec{i}| > q$. Due to (23), $e(A(q,d)) = \max\{\prod_{k=1}^{d}\lambda_{m_{i_k-1}+1} :\; |\vec{i}| = q+1\}$, as claimed.

Due to (18), we have $\|S_1 - U_i\| \;=\; \lambda_{m_i+1} \;\le\; C\,D^i$. Using these estimates on $\lambda_{m_{i-1}+1}$, we get

$$e(A(q,d)) \;\le\; C^d \prod_{k=1}^{d} D^{i_k-1} \;=\; C^d\, D^{|\vec{i}|-d} \;=\; C^d\, D^{q-d+1},$$

as claimed. □

It is easy to check that the information $N_{q,d}$ is, in general, not optimal for $d > 2$ although it consists of optimal information for $d = 1$. How much $N_{q,d}$ differs from optimal information will be checked in Subsection 8.5.

4.2 Average Case

We now analyze the error of the algorithm $A(q,d)$ in the average case setting. As in Section 4.1, this will be done in terms of its error for $d = 1$. The following assumptions will be used for $d = 1$:

$$\|S_1\|_{\mu_1} \;\le\; B, \tag{26}$$

$$\|S_1 - U_i\|_{\mu_1} \;\le\; C\,D^i, \qquad \forall\, i \ge 0, \tag{27}$$

$$\|\Delta_i\|_{\mu_1} \;=\; \|U_i - U_{i-1}\|_{\mu_1} \;\le\; E\,D^i, \qquad \forall\, i \ge 1, \tag{28}$$

for some constants B, C, D, E with $0 < B \le C$ and $D < 1$.

4.2.1 General Case

For the general case of non-nested information and arbitrary algorithms U_i, the same proof technique as in Lemma 2 yields that (26), (27), and (28) imply

$$e(A(q,d)) \;=\; \|S_d - A(q,d)\|_{\mu_d} \;\le\; CH^{d-1}\begin{pmatrix} q \\ d-1 \end{pmatrix} D^q \quad \text{with} \quad H = \max\{B/D, E\}. \quad (29)$$

This upper bound can significantly be improved for nested information and optimal algorithms U_i.

4.2.2 Nested Information and Optimal Algorithms

As in Section 4.1.2, we now assume that, for $d = 1$, the information N_i used by the algorithms U_i is nested. We also assume that the algorithms U_i are optimal, i.e., they minimize the average case error among all algorithms that use the information N_i.

We now recall the form of optimal algorithms in the average case setting for arbitrary d. It is known, see, e.g., [23], that for Gaussian measures and a linear operator S, the optimal algorithm that uses information N equals $Sm(y)$, where $m(y)$ is the mean of the conditional measure given $y = N(f)$. Hence, for $d = 1$, U_i is optimal iff

$$U_i = S_1 \mathcal{P}_i, \quad (30)$$

where $\mathcal{P}_i(y)$ is the mean of the conditional measure $\mu_1(\cdot | N_i(f) = y)$.

For $d \ge 2$, we use the information $N = N(q,d)$, and it is easy to verify that

$$\mathcal{P}(q,d)(g) = \sum_{\vec{i} \in Q(q,d)} \left(\bigotimes_{k=1}^{d} (\mathcal{P}_{i_k} - \mathcal{P}_{i_k - 1}) \right)(g)$$

is the mean of the conditional measure $\mu_d(\cdot | N(f) = N(g))$. Due to (10), we have

$$A(q,d) = S_d \mathcal{P}(q,d).$$

Hence, $A(q,d)$ is optimal for arbitrary d. (In fact, this holds without assuming the tensor product form of S_d and/or Hilbertian properties of G_d). We summarize this in the following lemma.

Lemma 5 *For nested information N_i and optimal U_i of (30), the resulting algorithm $A(q,d)$ is optimal, i.e., it minimizes the average case error among all algorithms that use the same information.*

To estimate the error of $A(q,d)$ we now use the fact that G_d is a Hilbert space and S_d is a tensor product operator. Recall that the error $e(A(q,d))$ of $A(q,d)$ is now given by $e(A(q,d)) = \|S_d - A(q,d)\|_{\mu_d}$. For any continuous linear operator $T : F_d \to G_d$, we have $\|T\|^2_{\mu_d} = \int_{F_d} \|T(f)\|^2_{G_d}\, \mu_d(df)$. Hence, this norm is generated by an inner-product,

$$\langle T, R\rangle_{\mu_d} = \int_{F_d} \langle T(f), R(f)\rangle_{G_d}\; \mu_d(df)$$

for continuous linear $T, R : F_d \to G_d$.

Since U_i are optimal and the information N_i is nested, it is easy to verify that the following formulas hold:

$$\begin{aligned} \langle S_1, U_i\rangle_{\mu_1} &= \|U_i\|^2_{\mu_1} &&= \|S_1\|^2_{\mu_1} - \|S_1 - U_i\|^2_{\mu_1}, && (31)\\ \langle S_1 - U_i, S_1 - U_j\rangle_{\mu_1} &&&= \|S_1 - U_{\max\{i,j\}}\|^2_{\mu_1} = e^2(U_{\max\{i,j\}}), && (32)\end{aligned}$$

where, as before, $e(U_i) = \|S_1 - U_i\|_{\mu_1}$ ($e(U_0) = \|S_1\|_{\mu_1}$).

This yields $\|U_i - U_{i-1}\|^2_{\mu_1} = e^2(U_{i-1}) - e^2(U_i)$ which corresponds to (19) of Section 4.1. Using (9), (31), and (32), it is easy to show that

$$\begin{aligned} e^2(A(q,d)) &= \|S_1\|^{2d}_{\mu_1} - \sum_{\vec{i}\in Q(q,d)} \prod_{k=1}^{d} \left(e^2(U_{i_k-1}) - e^2(U_{i_k})\right) && (33)\\ &= \|S_1\|^2_{\mu_1}\, e^2(A(q-1,d-1)) \\ &\quad + \sum_{\vec{i}\in Q(q-1,d-1)} e^2(U_{q-|\vec{i}|}) \prod_{k=1}^{d-1} \left(e^2(U_{i_k-1}) - e^2(U_{i_k})\right). && (34)\end{aligned}$$

Observe that $e^2(A(q,d))$ increases with $e(U_i)$. Indeed, consider

$$f_d(x_{0,1},\dots,x_{q-d+1,1},\dots,x_{0,d},\dots,x_{q-d+1,d}) = \prod_{k=1}^{d} x_{0,k} - \sum_{\vec{i}\in Q(q,n)} \prod_{k=1}^{d} (x_{i_k-1,k} - x_{i_k,k}). \qquad (35)$$

Obviously, for $x_{j,k} = e^2(U_j)$ we have that f_d equals $e^2(A(q,d))$. It is easy to observe that f_d is increasing in each of the variable as long as $0 \le x_{j,k} \le x_{j-1,k}$ for all j,k holds. This yields the desired property.

Remark 2 The error formulas (33) and (34), and their monotonicity hold under more general conditions. For instance, let $F_d = \bigotimes_{k=1}^d F_{1,k}$, $S_d = \bigotimes_{k=1}^d S_{k,1}$ with $S_{k,1} : F_{k,1} \to G_{k,1}$, $G_d = \bigotimes_{k=1}^d G_{k,1}$, and $\mu_d = \bigotimes_{k=1}^d \mu_{1,k}$. Hence, we can use now different spaces and operators for each $k = 1, 2, \ldots$. Assume that for each scalar problem $S_{1,k}$ we use optimal algorithms $U_{i,k}$. Then the error of the resulting algorithm $A(q,d)$ satisfies

$$e^2(A(q,d)) = f_d(x_{0,1}, \ldots, x_{q-d+1,1}, \ldots, x_{0,d}, \ldots, x_{q-d+1,d})$$

with $x_{j,k} = \|S_{1,k} - U_{1,k}\|^2_{\mu_{1,k}}$ and f_d given by (35). $\square$

Due to the monotonicity property, (26), and (27), we estimate the error of $A(q,d)$ by

$$\begin{aligned} e^2(A(q,d)) \;&\le\; C^2 e^2(A(q-1,d-1)) + C^{2d}(1-D^2)^{d-1} D^{2(q-d+1)} \sum_{\vec{i} \in Q(q-1,d-1)} 1 \\ &=\; C^2 e^2(A(q-1,d-1)) + C^{2d}(1-D^2)^{d-1} D^{2(q-d+1)} \binom{q-1}{d-1}. \end{aligned}$$

Therefore,

$$e^2(A(q,d)) \;\le\; C^{2d} D^{2(q-d+1)} \sum_{i=0}^{d-1} \left(1-D^2\right)^{(d-1-i)} \binom{q-i-1}{d-i-1}. \tag{36}$$

It is easy to check that for $q > (d-D^2)/(1-D^2)$, the largest term in the last sum is for $i = 0$. Therefore, in this case, one can estimate the last sum by d times the term for $i = 0$.

To bound the last sum by a closed form formula, we proceed as follows. Denoting $x = D^2$, we have

$$e^2(A(q,d)) \;\le\; C^{2d} x^{q-d+1} s(x), \tag{37}$$

where

$$s(x) \;=\; \sum_{i=0}^{d-1} \binom{q-i-1}{q-d} (1-x)^{d-1-i} \;=\; \sum_{j=0}^{d-1} \binom{q+j-d}{q-d} (1-x)^j.$$

Using the binomial equation for $(1-x)^j$ and changing the order of summation, we get

$$s(x) \;=\; \sum_{k=0}^{d-1} (-x)^k \sum_{j=k}^{d-1} \binom{j}{k} \binom{q+j-d}{q-d}.$$

Since the product of the two binomial coefficients in the second sum equals

$$\binom{q+k-d}{k} \binom{q-d+j}{q-d+k},$$

and since the sum with respect to j of the latter coefficient is $\begin{pmatrix} q \\ d-1-k \end{pmatrix}$, we get

$$s(x) \;=\; \sum_{k=0}^{d-1} (-x)^k \begin{pmatrix} q+k-d \\ k \end{pmatrix} \begin{pmatrix} q \\ d-1-k \end{pmatrix}.$$

The product of the last two coefficients is $q \begin{pmatrix} q-1 \\ d-1 \end{pmatrix}$ times $\begin{pmatrix} d-1 \\ k \end{pmatrix} /(q+k-d+1)$. Hence,

$$s(x) \;=\; q \begin{pmatrix} q-1 \\ d-1 \end{pmatrix} x^{-q+d-1}\, g(x) \qquad \text{where} \qquad g(x) \;=\; \sum_{k=0}^{d-1} (-1)^k \frac{x^{k+q-d+1}}{k+q-d+1} \begin{pmatrix} d-1 \\ k \end{pmatrix}.$$

Obviously, $x^{k+q-d+1}/(k+q-d+1) = \int_0^x t^{k+q-d}\, dt$. Therefore,

$$g(x) \;=\; \int_0^x t^{q-d} (1-t)^{d-1}\, dt. \tag{38}$$

In summary, we have the following lemma.

Lemma 6 *Let (26) and (27) hold. For nested information N_i, optimal U_i of (30), and $q \geq d$, we have*

$$e(A(q,d)) \;\leq\; C^d \sqrt{ q \begin{pmatrix} q-1 \\ d-1 \end{pmatrix} \int_0^{D^2} t^{q-d}(1-t)^{d-1}\, dt } \;<\; C^d D^{q-d+1} \sqrt{ \begin{pmatrix} q \\ d-1 \end{pmatrix} }. \tag{39}$$

Moreover, if $\|S_1 - U_i\|_{\mu_1} = C D^i$ for all $i \geq 0$, then the first inequality in (39) becomes an equality. If $q \geq (d - D^2)/(1-D^2)$ then

$$e(A(q,d)) \;\leq\; \sqrt{d}\, C^d\, D^{q-d+1}\, (1-D^2)^{(d-1)/2} \sqrt{ \begin{pmatrix} q-1 \\ d-1 \end{pmatrix} }. \tag{40}$$

Remark 3 In Lemma 6 we assume that (26) and (27) hold. Thus, in particular, we assume that $B \leq C$. It is sometimes better not to assume any relation between the constants B and C. This corresponds to requiring that (27) holds for $i \geq 1$.

Under the assumptions of Lemma 6 with (27) replaced by

$$\|S_1 - U_i\| \;\leq\; C\, D^i, \quad \forall\, i \geq 1,$$

we have

$$\frac{e^2(A(q,d))}{C^{2d}D^{2(q-d+1)}} \le \sum_{i=0}^{d-1} \beta^{2i}(1-D^2)^{d-i-1} \sum_{k=0}^{d-1} \binom{d-i-1}{k} \binom{q-d}{d-i-k-1} \left(1+\frac{\beta^2-1}{1-D^2}\right)^k .$$

Here $\beta = B/C$ and we assume that $\beta \ge D$. The last assumption $\beta \ge D$ is quite natural since optimality of U_i yields $\|S_1 - U_i\| \le \|S_1\|$. We add that if $\|S_1\| = B$ and $\|S_1 - U_i\| = C\,D^i$ for $i \ge 1$ then the inequality in the formula for the error of $A(q,d)$ becomes an equality.

To prove this we use (34). Let $\vec{i} \in Q(q-1,d-1)$. For $i_k > 1$, we can replace $e^2(U_{i_k-1}) - e^2(U_{i_k})$ by $C^2D^{2(i_k-1)}(1-D^2)$, whereas for $i_k = 1$, we can replace $e^2(U_0) - e^2(U_1)$ by $B^2 - C^2D^2$. Let $k(\vec{i})$ denote the number of indices in $\vec{i}$ which are equal to 1. Then we have

$$e^2(A(q,d)) \le B^2 e^2(A(q-1,d-1)) + a(q-1,d-1), \tag{41}$$

where

$$a(q-1,d-1) = \sum_{\vec{i}\in Q(q-1,d-1)} C^2 D^{2(q-|\vec{i}|)} D^{2(|\vec{i}|-d+1)} C^{2(d-1-k(\vec{i}))} (1-D^2)^{d-1-k(\vec{i})} (B^2 - C^2D^2)^{k(\vec{i})}.$$

Since $B^2 - C^2D^2 = C^2(1-D^2)\gamma$ with $\gamma = (1 + (\beta^2-1)/(1-D^2))$, we can simplify the last equality to

$$a(q-1,d-1) = C^{2d}D^{2(q-d+1)}(1-D^2)^{d-1} \sum_{\vec{i}\in Q(q-1,d-1)} \gamma^{k(\vec{i})}.$$

The summation for $\vec{i} \in Q(q-1,d-1)$ can be performed with respect to the length p of vectors $\vec{i}$, and with respect to the number k of indices in $\vec{i}$ that equal 1. Observe that for fixed p and k with $p \ge 2d-k-2$, we have $\binom{d-1}{k}\binom{p-d}{d-2-k}$ vectors $\vec{i}$ of length p and with k indices equal to 1. Hence,

$$\sum_{\vec{i}\in Q(q-1,d-1)} \gamma^{k(\vec{i})} = \sum_{k=0}^{d-1} \gamma^k \binom{d-1}{k} \sum_{p=2d-k-2}^{q-1} \binom{p-d}{d-k-2}.$$

Solving (41) in terms of $a(q,d)$ we get

$$e^2(A(q,d)) \le B^{2(d-1)} e^2(A(q-d+1,1)) + \sum_{i=0}^{d-2} B^{2i} a(q-1-i,d-1-i) = \sum_{i=0}^{d-1} B^{2i} a(q-1-i,d-1-i),$$

since $a(q-d,0)=1$. Using the form of $a(q-1-i,d-1-i)$ and the identity

$$\sum_{p=2(d-i)-k-2}^{q-i-1} \binom{p-d+i}{d-2-i-k} = \binom{q-d}{d-i-k-1},$$

see 0.151.1 of [6], we obtain the needed estimate.

Note that for $B=C$ we have $\beta=1$ and the sum with respect to k is $\binom{q-i-1}{d-i-1}$ which agrees with Lemma 6. $\square$

5 Cost Analysis

Similarly as in the previous section, we estimate the cost of the algorithm $A(q,d)$ for arbitrary d by the cost of the algorithms U_i for $d=1$. Since $A(q,d)$ is a linear algorithm, its cost is estimated by $m(q,d)(c(d)+2)-1$, see Section 2. Here $m(q,d)$ is the cardinality of the information $N(q,d)$ used by the algorithm $U(q,d)$, and $c(d)$ is the cost of computing one linear functional.

We now discuss the cardinality m_i of information N_i used by the algorithms U_i for $d=1$, see (14). In Section 4 we assume that the error of U_i is of order D^i. Hence, we want to define m_i such that this error estimate holds. For many problems, the error depends on some power of the reciprocal of m_i, $\|S_1-U_i\| = O(m_i^{-p})$ for some positive p. Hence, to satisfy (18) or (27) we have to take $m_i = O(D^{-1/p})^i$ which means that m_i depends exponentially on i. More specifically, we assume that

$$m_i \le F_0\left(F^i-1\right) \tag{42}$$

for some numbers $F>1$ and $F_0>0$. The minus 1 in the above formula is taken to simplify initial estimates. Moreover, it makes the bound sharp for $i=0$ since for $U_i=0$ we have $m_0=0$. A more general case is considered in Remark 4.

Of course, the cardinality $m(q,d)$ of information used by the algorithm $A(q,d)$ depends on F_0 through F_0^d. Thus, for the purpose of the following estimates, we can assume that $F_0=1$. For non-nested information,

$$m(q,d) \le \sum_{\vec{i}\in P(q,d)} \prod_{k=1}^{d} F^{i_k} = \sum_{\vec{i}\in P(q,d)} F^{|\vec{i}|} = \sum_{i=\max\{d,q-d+1\}}^{q} F^i \binom{i-1}{d-1}.$$

To estimate the last sum, observe that $\left(\begin{array}{c} i-1 \\ d-1 \end{array}\right)$ is a nondecreasing function of i and we can replace $\left(\begin{array}{c} i-1 \\ d-1 \end{array}\right)$ by $\left(\begin{array}{c} q-1 \\ d-1 \end{array}\right)$. From this we have

$$m(q,d) \le \frac{F^{q+1} - F^{\max\{d,q-d+1\}}}{F-1} \left(\begin{array}{c} q-1 \\ d-1 \end{array}\right) \le \frac{F}{F-1} F^q \left(\begin{array}{c} q-1 \\ d-1 \end{array}\right).$$

We now analyze the case of nested information. For any $\vec{s} = [s_1, \ldots, s_d]$, let $U_{\vec{s}} = \bigotimes_{i=1}^d U_{s_i}$. Since the algorithm $A(q,d)$ is a combination of $U_{\vec{s}}$'s, see Lemma 1, and for $|\vec{s}| < q$, $U_{\vec{s}}$ uses information contained in another $U_{\vec{r}}$ with $\vec{s} \le \vec{r}$ and $|\vec{r}| = q$, we only need to consider $|\vec{s}| = q$.

For $|\vec{s}| = q$, let $\vec{s}\,' = [s_1, \ldots, s_{d-1}]$. If $|\vec{s}\,'| = d-1$ then $s_d = q-d+1$ and hence $U_{\vec{s}}$ requires at most $(F-1)^{d-1}(F^{q-d+1}-1)$ functionals. For $|\vec{s}\,'| = p \ge d$, there are at most

$$(F^{q-p}-1)\prod_{i=1}^{d-1}(F^{s_i} - F^{s_i-1}) = (F^{q-p}-1)F^{p-d+1}(F-1)^{d-1}$$

functionals used by $U_{\vec{s}}$ that are not used by any other $U_{\vec{v}}$ with $|\vec{v}| = q$. For a fixed p, there are $\left(\begin{array}{c} p-1 \\ d-2 \end{array}\right)$ of different $\vec{s}\,'$ with $|\vec{s}\,'| = p$. Since $p \le q-1$, the cardinality $m(q,d)$ is bounded by

$$\overline{m}(q,d) = (F-1)^{d-1} \sum_{p=d-1}^{q-1} \left(F^{q-d+1} - F^{p-d+1}\right) \left(\begin{array}{c} p-1 \\ d-2 \end{array}\right).$$

Hence

$$\overline{m}(q,d) - \overline{m}(q-1,d) = F^{q-d}(F-1)^d \sum_{p=d-1}^{q-1} \left(\begin{array}{c} p-1 \\ d-2 \end{array}\right) = F^{q-d}(F-1)^d \left(\begin{array}{c} q-1 \\ d-1 \end{array}\right),$$

the latter equality follows from 0.151.1 of [6]. Hence

$$\overline{m}(q,d) = (F-1)^d \sum_{j=0}^{q-d} F^j \left(\begin{array}{c} j+d-1 \\ d-1 \end{array}\right) \le (F-1)^{d-1} F^{q-d+1} \left(\begin{array}{c} q-1 \\ d-1 \end{array}\right).$$

We summarize the cost estimate in the following Lemma.

Lemma 7 *Let (42) hold and $q \ge d$. Then we have*

$$cost(q,d) \;\le\; (c(d)+2)\, m(q,d),$$

where for non-nested information

$$m(q,d) \;\le\; F_0^d \frac{F}{F-1}\, F^q \left(\begin{array}{c} q-1 \\ d-1 \end{array} \right), \tag{43}$$

and for nested information

$$m(q,d) \;\le\; F_0^d (F-1)^d \sum_{j=0}^{q-d} F^j \left(\begin{array}{c} j+d-1 \\ d-1 \end{array} \right) \;\le\; F_0^d \left(\frac{F-1}{F}\right)^{d-1} F^q \left(\begin{array}{c} q-1 \\ d-1 \end{array} \right). \tag{44}$$

For $m_i = F_0(F^i - 1)$ the first inequality in (44) becomes an equality.

As already mentioned in Section 2, we use precomputation to obtain the cost bound $(c(d)+2)m(q,d)$. In our case, we need to precompute the elements

$$(-1)^{q-|\vec{i}\,|} \left(\begin{array}{c} d-1 \\ q-|\vec{i}\,| \end{array} \right) g_{\vec{i},\vec{j}}.$$

We stress that this precomputation is sometimes very easy. Indeed, if $G_i = \mathbb{R}$ then $g_{\vec{i},\vec{j}}$ is the product of d numbers and can be computed in $d-1$ scalar multiplications of numbers. On the other hand, if $G_i = L_2([0,1])$ then the function $g_{\vec{i},\vec{j}}$ is the product of d one dimensional functions and $g_{\vec{i},\vec{j}}(t) = \prod_{k=1}^d a_{i_k,j_k}(t_k)$ can be computed in $d-1$ scalar multiplications of one dimensional functions. Hence, if we know the elements for $d=1$ then usually it is easy to obtain the corresponding elements for arbitrary d.

Remark 4 We now generalize the estimates on the cardinality $m(q,d)$ assuming that for $d=1$ we use nested information N_i with cardinality

$$m_i \;\le\; F_0(F^i - b), \qquad \forall i \ge 1,$$

where $F_0 > 0$ is positive and $F > \max\{1,b\}$. We prove that

$$m(q,d) \le F_0^d (F-1)^{d-1} F^{q-d+1} \sum_{k=0}^{d-1} \left(\begin{array}{c} d-1 \\ k \end{array} \right) \left(\frac{F-b}{F-1}\right)^k \sum_{p=2d-k-2}^{q-1} \left(\begin{array}{c} p-d \\ d-k-2 \end{array} \right) \left(1 - \frac{b}{F^{q-p}}\right).$$

Clearly, the dependence on F_0 is through F_0^d. Hence, we can assume that $F_0 = 1$. Repeating the analysis of this section for nested information, we need only to consider vectors $|\vec{s}| = q$. As before, let $\vec{s}' = [s_1, \dots, s_{d-1}]$. For $|\vec{s}'| = p \geq d-1$, there are exactly

$$(F^{q-p} - b) \prod_{i=1}^{d-1} (F^{s_i} - F^{s_i - 1} + (1-b)\delta_{s_i,1}) \;=\; (F^{q-p} - b) F^{p-d+1} (F-1)^{d-1} \left(\frac{F-b}{F-1} \right)^k$$

functionals used by $U_{\vec{s}}$ that are not used by any other $U_{\vec{v}}$ with $|\vec{v}| = q$. Here k is the number of $s_i = 1$. As already mentioned in Remark 3, we have $\left(\begin{array}{c} d-1 \\ k \end{array} \right) \left(\begin{array}{c} p-d \\ d-2-k \end{array} \right)$ vectors $\vec{s}$ of length p with k indices equal to 1, $p \geq 2d-k-2$. Summing up we obtain the needed formula. □

6 ε-Cost Analysis

In this section, we analyze the cost of the algorithm $A(q,d)$ with error at most ε. That is, we determine a possibly minimal q for which $e(A(q,d)) \leq \varepsilon$ and estimate the cost of $A(q,d)$.

First of all observe that $e(U_0) \leq B^d$ implies $\mathrm{cost}(\varepsilon, d) = 0$ for $\varepsilon \geq B^d$. We also mention the easy case of $d=1$. For $d=1$ we have $e(U_q) \leq C\, D^q$ and $\mathrm{cost}(A(q,1)) \leq (c(1)+2)\, F_0 F^q$. Hence, $e(A(q,1)) \leq \varepsilon$ for $q = \lceil (\ln\, C/\varepsilon)/(\ln\, D^{-1}) \rceil$ and

$$\mathrm{cost}(A(q,1)) \;\leq\; (c(1)+2)\, F_0\, F\, \left(\frac{C}{\varepsilon} \right)^{\ln F / \ln D^{-1}} .$$

In what follows, we consider the remaining case when $q \geq d \geq 2$.

We begin with the general case of non-nested information. Let $q = q(d,\varepsilon)$ be the minimal integer for which the error bound (20) does not exceed ε. Hence

$$\left(\begin{array}{c} q \\ d-1 \end{array} \right) \;\leq\; \frac{\varepsilon}{C H^{d-1} D^q} \tag{45}$$

and (43) imply

$$m(q,d) \leq \frac{F_0^d F}{F-1} \, \frac{q-d+1}{q} \, \frac{\varepsilon}{C H^{d-1}} \left(\frac{F}{D} \right)^q . \tag{46}$$

To estimate q we proceed as follows. Let $q = x(d-1)$ with $x \geq d/(d-1)$. It is easy to check that

$$\left(\begin{array}{c} q \\ d-1 \end{array} \right) \quad\leq\quad \left(x \left(1 + \frac{1}{x-1} \right)^{x-1} \right)^{d-1} \sqrt{\frac{x}{2\pi(d-1)(x-1)}} \tag{47}$$

$$\leq \quad (xe)^{d-1}\frac{1}{\sqrt{(2\pi(d-1))}}\sqrt{\frac{x}{x-1}} \leq (xe)^{d-1}\sqrt{\frac{d}{2\pi(d-1)}}. \tag{48}$$

Let $x = t/\ln D^{-1}$. Using (48) instead of the left-hand-side of (45) we get

$$t \geq \ln t + \ln h \qquad \text{with} \qquad h = h(\varepsilon, d) = \frac{eH}{\ln D^{-1}}\left(\frac{C}{\varepsilon}\sqrt{\frac{d}{2\pi(d-1)}}\right)^{1/(d-1)}. \tag{49}$$

Observe that $h > e\,D^{-1}/\ln\, D^{-1} \geq e^2$ since

$$\varepsilon \leq B^d \leq CB^{d-1} \leq C(HD)^{d-1} < C(eHD)^{d-1}\sqrt{d/(2\pi(d-1))}.$$

Consider the following sequence of $t_k = t_k(h)$:

$$t_0 = \frac{e}{e-1}\ln h \qquad \text{and} \qquad t_{k+1} = \ln(ht_k). \tag{50}$$

It is easy to verify that t_0 satisfies (49), and then to show by a simple induction that all t_k's satisfy this inequality. Hence

$$t^* \leq t_k, \quad \forall k,$$

where t^* is a unique solution, $t^* = \ln\, t^* + \ln\, h$. Clearly, $t^* > \ln\, D^{-1}$ since $x > 1$.

Remark 5 The sequence of t_k's converges monotonically to t^*. Indeed, consider $y_k = \ln(hy_{k-1})$ with $y_0 = \ln h$. Then $y_k \leq t^*$. It can be easily checked that $t_k - y_k$ converges to zero. Moreover the convergence is quite fast, and thus this can be used in an algorithm implementation when computing t^*. $\square$

Hence, $q(\varepsilon, d) \leq \lceil t_{k+1}(d-1)/\ln D^{-1}\rceil = x(d-1)$ with $x \geq d/(d-1)$, and

$$q(\varepsilon, d) \leq 1 + t_{k+1}\frac{d-1}{\ln D^{-1}}, \quad \forall k \geq 0.$$

Using this in (46), we get

$$m(q, d) \leq \frac{F_0^d F^2}{F-1}\frac{1}{DH^{d-1}}\left(\sqrt{\frac{d}{2\pi(d-1)}}\right)^{\alpha+1}\left(\frac{eH}{\ln D^{-1}}t_k\right)^{(\alpha+1)(d-1)}\left(\frac{C}{\varepsilon}\right)^{\alpha},$$

where

$$\alpha = \frac{\ln F}{\ln D^{-1}}.$$

For simplicity we use the last inequality only for t_0. Then

$$m(q,d) \le \frac{F_0 F^2}{(F-1)D} \left(\frac{d}{2\pi(d-1)}\right)^{(\alpha+1)/2} \left(\frac{C}{\varepsilon}\right)^{\alpha}$$
$$\times \left(\frac{e^2 F_0^{1/(\alpha+1)} H^{\alpha/(\alpha+1)}}{(e-1)\ln D^{-1}} \left(\ln\left(\frac{eH}{\ln D^{-1}}\right) + \frac{\ln(\sqrt{d/(2\pi(d-1))}) + \ln(C/\varepsilon)}{d-1} \right)\right)^{(\alpha+1)(d-1)}.$$

For nested information, the cardinality $m(q,d)$ has the same bounds as above multiplied by $((F-1)/F)^d$.

We now consider the average case setting. For a general case of non-nested information, the cost of $A(q,d)$ is bounded by the same formula as in Theorem 1 for non-nested information. For nested information with optimal U_i, to guarantee $A(q,d) \le \varepsilon$ we can take $q = q(\varepsilon, d)$ such that

$$C^{2d} D^{2(q-d+1)} \binom{q}{d-1} = C^2 \left(\frac{C^2}{D^2}\right)^{d-1} D^{2q} \binom{q}{d-1} \le \varepsilon^2,$$

see (39). This leads to the same analysis as in the worst case setting with ε, C, D, and H replaced by ε^2, C^2, D^2, and C^2/D^2, respectively. In particular, $h(\varepsilon, d)$ from (49) and (50) is now $eC^2/(2\,D^2 \ln D^{-1})(C^2\varepsilon^{-2}(d/(2\pi(d-1)))^{1/2})^{1/(d-1)}$.

The above analysis and the relation between the cardinality $m(q,d)$ and the cost lead to the following theorem. In this theorem, we use the following abbreviations: *worst case nested* when the worst case setting and nested information are considered, *average case nested* when the average case setting with nested information and optimal U_i are considered, and *general* when non-nested information (either in worst case or average case setting) is considered.

Theorem 1 *Let the problem satisfy assumptions (17), (18), and (19) if the worst case setting is considered, and assumptions (26), (27), and (28) if the average case setting is considered. Moreover, we assume (42). Let $d \ge 2$ and $\varepsilon \le B^d$. Let*

$$t_{k+1} = \ln(h t_k) \qquad \textit{with} \qquad t_0 = \frac{e}{e-1} \ln h.$$

For

$$q = \left\lceil t_{k+1} \frac{d-1}{\ln\, D^{-1}} \right\rceil$$

with an arbitrary $k \ge 0$, the algorithm $A_\varepsilon(d) = A(q,d)$ has error at most ε and its cost is bounded by

$$\mathrm{cost}(A_\varepsilon(d)) \le (c(d)+2)\alpha_0(d)\left(\alpha_1 + \alpha_2 \frac{\ln(\sqrt{d/(2\pi(d-1))}) + \ln(C/\varepsilon)}{d-1}\right)^{\alpha_3(d-1)} \left(\frac{C}{\varepsilon}\right)^{\alpha}.$$

Here

$$\alpha = \frac{\ln F}{\ln D^{-1}},$$

and depending on the specific case, the values of h, α_1, α_2, and α_3 are given by:

$$h = h(\varepsilon,d) = \begin{cases} \frac{eH}{\ln D^{-1}} \left(\frac{C}{\varepsilon}\sqrt{\frac{d}{2\pi(d-1)}}\right)^{1/(d-1)}, & \textit{general} \\ \frac{eC^2}{2D^2 \ln D^{-1}} \left(\frac{C^2}{\varepsilon^2}\sqrt{\frac{d}{2\pi(d-1)}}\right)^{1/(d-1)}, & \textit{average case nested} \end{cases}$$

$$\alpha_0(d) = \begin{cases} \left(\frac{d}{2\pi(d-1)}\right)^{(\alpha+1)/2} \frac{F_0 F^2}{(F-1)D}, & \textit{general} \\ \left(\frac{d}{2\pi(d-1)}\right)^{(\alpha+1)/2} \frac{F_0 F}{D}, & \textit{worst case nested} \\ \left(\frac{d}{2\pi(d-1)}\right)^{(\alpha+2)/4} \frac{F_0 F}{D^2}, & \textit{average case nested} \end{cases}$$

$$\alpha_1 = \begin{cases} \alpha_2 \ln\left(\frac{eH}{\ln D^{-1}}\right), & \textit{general} \\ \alpha_2 \ln\left(\frac{e\,C^2}{2D^2 \ln D^{-1}}\right)/2, & \textit{average case nested} \end{cases}$$

$$\alpha_2 = \begin{cases} \frac{e^2 F_0^{1/(\alpha+1)} H^{\alpha/(\alpha+1)}}{(e-1)\ln D^{-1}}, & \textit{general} \\ \left(\frac{F_0(F-1)}{F}\right)^{1/(\alpha+1)} \frac{e^2 H^{\alpha/(\alpha+1)}}{(e-1)\ln D^{-1}}, & \textit{worst case nested} \\ \left(\frac{F_0(F-1)}{F}\right)^{2/(\alpha+2)} \frac{e^2}{(e-1)\ln D^{-1}} \left(\frac{C}{D}\right)^{2\alpha/(\alpha+2)}, & \textit{average case nested} \end{cases}$$

$$\alpha_3 = \begin{cases} \alpha+1, & \textit{general} \\ \alpha/2+1, & \textit{average case nested,} \end{cases}$$

where $H = \max\{B/D, E\}$.

We now comment on Theorem 1. The essence of the estimate of Theorem 1 is that for arbitrary d the cost of computing an ε-approximation is fully determined by the constants from the one dimensional case, $d = 1$. To focus on the dependence on d, we slightly simplify the estimate on $\mathrm{cost}(A_\varepsilon(d))$. Since $\alpha_0(d)$ is decreasing in d we have

$$\mathrm{cost}(A_\varepsilon(d)) \le (c(d)+2)\ \beta_1 \left(\beta_2 + \beta_3 \frac{\ln 1/\varepsilon}{d-1}\right)^{\beta_4(d-1)} \left(\frac{1}{\varepsilon}\right)^{\beta_5}, \tag{51}$$

where $\beta_1 = \alpha_0(2)C^\alpha$, $\beta_2 = \alpha_1 + \alpha_2\left(\ln\, C/\sqrt{2\pi}\right)$, $\beta_3 = \alpha_2$, $\beta_4 = \alpha_3$ and $\beta_5 = \alpha$. (This is the formula mentioned in the abstract.)

Observe that the leading factor, $(1/\varepsilon)^{\beta_5}$, of the cost has the same exponent for all d. The value of β_5 depends on the quality of the information N_i and the algorithms U_i for $d = 1$. Sometimes we can choose them in such a way that β_5 is minimized. The next leading factor of the cost is of the form

$$\mathrm{cost}_2(\varepsilon, d) \;=\; \left(\beta_2 \,+\, \beta_3\,\frac{\ln\, 1/\varepsilon}{d-1}\right)^{\beta_4(d-1)}.$$

For fixed d and ε tending to zero we have

$$\mathrm{cost}_2(\varepsilon, d) \;=\; \left(\frac{\beta_3}{d-1}\right)^{\beta_4(d-1)} \left(\ln\,\frac{1}{\varepsilon}\right)^{\beta_4(d-1)} (1 + o(1)),$$

where the constant in the o notation may depend on d. Observe that the asymptotic constant goes to zero super-exponentially with d. We stress that the exponent β_4 is sometimes too large. That is, for some problems, there exist algorithms for which the cost of computing an ε-approximation is

$$(c(d) + 2)\, C_d \left(\ln\,\frac{1}{\varepsilon}\right)^{\gamma(d-1)} \left(\frac{1}{\varepsilon}\right)^{\beta_5} \quad \text{with } \gamma < \beta_4.$$

This indicates that the algorithm does not, in general, minimize the cost of computing an ε-approximation although the loss is usually only by a power of $\ln\, 1/\varepsilon$. Moreover, for those problems, the dependence of C_d on d is unknown and it could happen that C_d grows super-exponentially with d.

We now fix ε and vary d. Observe the very interesting dependence of $\mathrm{cost}_2(\varepsilon, d)$ on d. With increasing d, the power grows but $\beta_3 \ln\, 1/\varepsilon$ is divided by increasing numbers. For $\beta_2 > 0$, due to $(1 + x/(d-1))^{d-1} \le e^x$ we have

$$\mathrm{cost}_2(\varepsilon, d) \;\le\; \beta_2^{\beta_4(d-1)} \left(\frac{1}{\varepsilon}\right)^{\beta_3\beta_4/\beta_2}$$

Hence, for $\beta_2 = 1$ the dependence on d disappear, for $\beta_2 < 1$ it goes exponentially fast to zero, and for $\beta_2 > 1$ it goes exponentially fast to infinity.

This indicates that the quality of the algorithm $A(q, d)$ may depend on β_2. However, it is not clear whether the condition $\beta_2 > 1$ is an overestimate of the cost of the algorithm $A(q, d)$, or that the algorithm $A(q, d)$ is not good, or that the tensor product problem is hard. This is discussed in the next section.

7 Strong Tractability

In this section we utilize the concept of strong tractability, see [32]. A tensor product problem $\{F_d, G_d, S_d\}$ is *strongly tractable* iff the complexity of computing an ε-approximation is bounded by $c(d)K(\varepsilon)$ where $K(\varepsilon)$ is a polynomial in $1/\varepsilon$. This is equivalent to the condition that there are two nonnegative numbers K and p such that for every $d = 1, 2, \dots$ and $\varepsilon \le 1$ there exists an algorithm A which computes an ε-approximation for the d dimensional case with cost

$$\mathrm{cost}(A; \varepsilon, d) \;\le\; (c(d) + 2)\, K\, \varepsilon^{-p}. \tag{52}$$

Hence, the only dependence on the dimension d is through the cost $c(d)$ of one evaluation of a linear functional. This also means that the number of linear functionals needed for an ε-approximation is independent of d and depends polynomially on $1/\varepsilon$. An algorithm A which satisfies (52) is said to be *strongly polynomial-time*. The smallest (or the infimum of) p for which (52) holds is called the *strong exponent* of the tensor product problem.

Let Λ denote the class of continuous linear functionals which can be used by the algorithms $A(q, d)$. We consider two classes. The first class $\Lambda = \Lambda^{\mathrm{std}}$ consists of function evaluations, whereas the second class $\Lambda = \Lambda^{\mathrm{all}}$ consists of *all* continuous linear functionals.

We now discuss the class Λ^{std}. Recall that in the worst case setting we assume that F_d is a Hilbert space. Then continuity of functionals in Λ^{std} is equivalent to F_d being a reproducing kernel space, see [1]. Obviously, F_d is a reproducing kernel space iff F_1 has this property. In the average case setting, the continuity of function evaluations is equivalent to continuity of the covariance kernel function of the measure μ_d. Strong tractability in the class Λ^{std} is equivalent to strong tractability in the class Λ^{all} if $\lambda(\mathcal{D})^d K_{1,d}^2 K_{2,d}$ is uniformly bounded in d. Here, $\lambda(\mathcal{D})$ is the Lebesgue measure of the domain $\mathcal{D} \subset \mathbb{R}$, $K_{1,d}$ is a bound of the operator S_d in the $L_2(\mathcal{D}^d)$ norm, and $K_{2,d}$ is a bound of the function $f(t) = R_d(t, t)$ in the $L_\infty(\mathcal{D}^d)$ norm, where R_d is the reproducing kernel of the space F_d in the worst case setting, and the covariance kernel function in the average case setting, see [32].

For the class Λ^{all}, it is known which problems are strongly tractable. Obviously, if S_1 is a continuous linear functional then the problem is strongly tractable. If S_1 is not a linear functional, i.e., $\dim S_1(F_1) \ge 2$ or, equivalently, $W_1 = (S_1^* S_1)^{1/2}$ has at least two positive eigenvalues, then strong tractability depends on the setting.

In the worst case setting, strong tractability holds iff

$$\lambda_1 \;=\; \|S_1\| < 1 \qquad \text{and} \qquad \lambda_n \;=\; O(n^{-k})$$

for some positive k. Here, $\lambda_1 \ge \lambda_2 \ge \cdots$ are eigenvalues of W_1, see Theorem 3.1 of [31].

In the average case setting, strong tractability holds iff

$$\|S_1\|^2_{\mu_1} \; = \; \sum_{n=1}^{\infty} \gamma_n \; < \; 1 \qquad \text{and} \qquad \gamma_n \; = \; O(n^{-1-k})$$

for some positive k. Here, $\gamma_1 \geq \gamma_2 \geq \cdots$ are eigenvalues of the covariance operator C_ν of the measure $\nu = \mu_1 S_1^{-1}$, $C_\nu : G_1 \to G_1$ and $C_\nu g = S_1(C_\mu L_g S)$, $\forall g \in G$, where $L_g(h) = \langle g, h \rangle_{G_1}$, see (i) of the Appendix.

It is also known that if strong tractability does not hold then the cost of any algorithm for computing an ε-approximation has stronger than polynomial dependence on d.

We now relate the assumptions of strong tractability with the assumptions we have made concerning the algorithm $A(q,d)$. Observe that the assumption on λ_1 or $\sum_{n=1}^{\infty} \gamma_n$ means that we can assume $B < 1$ in (17) or (26). The assumption on polynomial decay of λ_n or γ_n means that (18), (19) and (42) hold in the worst case setting, and (27), (28) and (42) hold in the average case setting.

This discussion on strong tractability motivates the assumptions of the next theorem which states that the algorithm $A(q,d)$ is strongly polynomial-time.

Theorem 2 *Let the problem satisfy assumptions (17), (18), and (19) if the worst case setting is considered, and assumptions (26), (27), and (28) if the average case setting is considered. Moreover, we assume (42).*

If $B < 1$ then the algorithm $A(q,d)$ is strongly polynomial-time. More precisely, the algorithm $A_\varepsilon(d) = A(q,d)$ with q defined as in Theorem 1 has error not exceeding ε and its cost is bounded by

$$\mathrm{cost}(A_\varepsilon(d)) \; \leq \; (c(d)+2)\, K \left(\frac{1}{\varepsilon}\right)^p, \quad \forall\, \varepsilon \; \leq \; B^d$$

where $K = \max\{FC^\alpha, K_1\}$ with

$$K_1 \; = \; \alpha_0(2) \max\left\{1, \left(\frac{C}{\sqrt{2\pi}B}\right)^{\alpha_2\alpha_3/(\alpha_1+\alpha_2 \ln B^{-1})}\right\} \left(\frac{C}{B}\right)^\alpha B^p$$

and

$$p \; = \; \alpha + \alpha_2\alpha_3\, q^*$$

where q^* is given by

$$q^* \; = \; \begin{cases} \frac{\ln \gamma_1}{\ln \gamma_2} & \text{if} \quad \gamma_1 \ln \gamma_1 - \ln \gamma_2 \; \geq \; 0, \\ q^{**} & \text{if} \quad \gamma_1 \ln \gamma_1 - \ln \gamma_2 \; < \; 0, \end{cases}$$

with $\gamma_1 = \alpha_1 + \alpha_2 \ln B^{-1}$ *and* $\gamma_2 = B^{-\alpha_2}$ *(we always have* $\gamma_1 \geq 0$ *and* $\gamma_2 > 1$*), and* q^{**} *is a unique solution from* $(0,1)$ *of the equation*

$$q\,\alpha_1 \;=\; 1 + \ln q.$$

Proof: For $d = 1$, it is trivial. For $d \geq 2$, Theorem 1 yields

$$\mathrm{cost}(A_\varepsilon(d)) \;\leq\; (c(d)+2)\,K_1\,\left(\alpha_1 + \alpha_2 \frac{\ln\,B/\varepsilon}{d-1}\right)^{\alpha_3(d-1)} \left(\frac{B}{\varepsilon}\right)^{\alpha} \frac{1}{B^p}.$$

Letting $B/\varepsilon = (x/B^{\alpha_2})^{(d-1)/\alpha_2}$, it is enough to verify whether

$$\gamma_1 \,+\, \ln\,x \;\leq\; (\gamma_2\,x)^{q^*}\,, \quad \forall x \geq 1. \tag{53}$$

Observe that $\gamma_1 \geq 0$ due to the definitions of α_1 and α_2, and the relations between B, C, D, H. Clearly, $B < 1$ and $\alpha_2 > 0$ imply $\gamma_2 > 1$. Thus, (53) has a solution. We now show that q^* is the smallest solution of (53).

Substituting $y = \ln x + \gamma_1$ and $a = \gamma_1 - \ln\gamma_2$, we need to show that

$$q^* = \sup_{y \geq \gamma_1} g(y) > 0 \quad \text{with} \quad g(y) := \frac{\ln y}{y-a}.$$

Note that $\lim_\infty g(y) = 0$ and $g'(y) = h(y)/(y(y-a)^2)$ with $h(y) = y - a - y\ln y$. Moreover, $h'(y) = -\ln y$.

Consider first $\ln\gamma_2 - \gamma_1 \ln\gamma_1 \leq 0$ which implies $\gamma_1 \geq 1$. Then $y \geq \gamma_1 \geq 1$ and the function g attains its maximum at $y^* = \gamma_1$ since $h'(y)$ is always negative and $h(\gamma_1) \leq 0$. Hence $q^* = g(\gamma_1) = (\ln\gamma_1)/(\ln\gamma_2)$, as claimed.

Consider now $\ln\gamma_2 - \gamma_1\ln\gamma_1 > 0$. Since $g'(\gamma_1) > 0$, the maximum of g is attained at a critical point y^* which is a root of h. Such a root is unique. Indeed, for $\gamma_1 \geq 1$, $h'(y) \leq 0$; and for $\gamma_1 < 1$, h does not have a root in $[\gamma_1, 1]$ since h' is positive and $h(\gamma_1) > 0$. Therefore, $q^* = g(y^*)$ where y^* is the unique solution of $y^* - a = y^* \ln y^*$; or equivalently of $(\ln y^*)/(y^* - a) = 1/y^*$. From the definition of g, we get $g(y^*) = (\ln y^*)/(y^* - a) = 1/y^*$. Substituting y^* by $1/g(y^*)$ in the definition of $g(y^*)$ we finally conclude that $g(y^*) = q^*$ is a unique solution of $1 - aq = \ln q$, as claimed. □

As before, the constant K and the exponent p in Theorem 2 are fully determined by the parameters for $d = 1$. The exponent p is, however, usually too large since we are using some overestimates on the error and on the cardinality of the algorithm $A_\varepsilon(d)$. We now show that sometimes the exponent p can be lowered by a different approach.

For simplicity we only consider the average case setting for nested information N_i and optimal U_i satisfying (26), (27) and (42). For $x \geq 1/(1-D^2)$ and $d \geq 2$, let

$$f(x) \;=\; F_0(F-1)F^{x-1}\frac{x^x}{(x-1)^{x-1}}, \qquad g(x) \;=\; C\,D^{x-1}\sqrt{\frac{x^x(1-D^2)}{(x-1)^{x-1}}},$$

and

$$a(d) \;=\; C\,\max\left\{\left(\frac{D^2d^2}{2\pi(d-1)}\right)^{1/4}, \left(\frac{B}{C}\right)^{d}\right\}, \qquad b(d) \;=\; \frac{F_0F^2}{D^3\sqrt{2\pi(d-1)}}.$$

Let

$$p^* = \max_{x\geq 1/(1-D^2)} \frac{\ln f(x)}{\ln\left(1/g(x)\right)}. \tag{54}$$

Since for $h(x) = \ln f(x)/\ln\left(1/g(x)\right)$ we have $h(x) > \ln F/\ln D^{-1} = \alpha$ for large x and $\lim_{x\to\infty} h(x) = \alpha$, the number p^* exists and $p^* \in (\alpha, +\infty)$.

Lemma 8 *Consider the average case setting with nested information N_i and optimal U_i for which (26), (27), and (42) hold with $B < 1$. For $\varepsilon < B^d$ with $d \geq 2$, define the algorithm $A_\varepsilon(d) = A(q,d)$ with $q = \lceil x(d-1)+1\rceil$ where x is a unique solution of*

$$g^{d-1}(x)\,a(d) \;=\; \varepsilon.$$

Then the error of $A_\varepsilon(d)$ is at most ε, and for any positive η the cost of $A_\varepsilon(d)$ is bounded by

$$cost(A_\varepsilon(d)) \;\leq\; (c(d)+2)\,K_d\left(\frac{1}{\varepsilon}\right)^{p^*+\eta} \;\leq\; (c(d)+2)\,C_\eta\left(\frac{1}{\varepsilon}\right)^{p^*+\eta}, \tag{55}$$

where

$$K_d \;=\; b(d)\,a^{p^*}(d)\,B^{d\eta} \qquad \textit{and} \qquad C_\eta \;=\; \max_{d\geq 2} K_d \;<\; +\infty. \tag{56}$$

Moreover, if

$$\frac{F_0(F-1)}{1-D^2}\left(\frac{F}{D^2}\right)^{D^2/(1-D^2)} \;>\; C^2 \tag{57}$$

then

$$p^* = \frac{\ln F + \ln\frac{x^*}{x^*-1}}{\ln 1/D - \frac{1}{2}\ln\frac{x^*}{x^*-1}}, \tag{58}$$

where $x^* > 1/(1-D^2)$ is a unique solution of

$$\frac{\ln f(x)}{\ln g(x)} \;=\; \frac{\ln(Fx/(x-1))}{\ln(D(x/(x-1))^{1/2})}. \tag{59}$$

Proof: Note that the function g is decreasing and $g(1/(1-D^2)) = C$. Hence, $g^{d-1}(1/(1-D^2))a(d) \geq B^d > \varepsilon$ and the equation $g^{d-1}(x)a(d) = \varepsilon$ has a unique solution $x > 1/(1-D^2)$. From the definition of q we obtain that $q = (x+y)(d-1)+1$ with $x \geq 1$ and $y \in [0, 1/(d-1)]$. We now show that

$$e(A(q,d)) \leq g^{d-1}(x)a(d) \qquad \text{and} \qquad m(q,d) \leq f^{d-1}(x)b(d). \tag{60}$$

Indeed, for $x \geq 1/(1-D^2)$ we have $q \geq (d-D^2)/(1-D^2)$, and (40) of Lemma 6 yields

$$\begin{aligned} e(A(q,d)) &\leq C^d D^{(x-1)(d-1)+1}(1-D^2)^{(d-1)/2}\sqrt{d\begin{pmatrix} x(d-1) \\ d-1 \end{pmatrix}} \\ &\leq g^{d-1}(x) C D \left(\frac{d^2 x}{2\pi(x-1)(d-1)}\right)^{1/4} \leq g^{d-1}(x) C \left(\frac{D^2 d^2}{2\pi(d-1)}\right)^{1/4}, \end{aligned}$$

as claimed in the first inequality of (60).

To show the second inequality, observe that for $h(x) = x^x/(x-1)^{x-1}$ we have

$$h(x+y) \leq h(x)\left(\frac{x}{x-1}\right)^y.$$

Using this and Lemma 7, we obtain

$$\begin{aligned} m(q,d) &\leq F_0(F-1)^{d-1}F^{(x+y-1)(d-1)+1}\begin{pmatrix} (x+y)(d-1) \\ d-1 \end{pmatrix} \\ &\leq f^{d-1}(x) F_0 F^2 \left(\frac{x}{x-1}\right)^{y(d-1)}\sqrt{\frac{x+y}{2\pi(d-1)(x+y-1)}} \\ &\leq f^{d-1}(x)\frac{F_0F^2}{D^2}\sqrt{\frac{1}{2\pi D^2(d-1)}}. \end{aligned}$$

This proves (60).

We now prove (55). It follows from (60) that the error of the algorithm $A_\varepsilon(d)$ is at most ε. To estimate the cardinality $m(q,d)$ of $A_\varepsilon(d)$ observe that the definition of p^* yields $f(x) \leq (1/g(x))^{p^*}$. Hence, (60) yields

$$\begin{aligned} m(q,d) &\leq b(d)\, g^{-p^*(d-1)}(x) = b(d)\, a(d)^{p^*}\, \varepsilon^{-p^*} \\ &\leq b(d)\, a(d)^{p^*}\, \varepsilon^{\eta}\, \varepsilon^{-p^*-\eta} \leq b(d)\, a(d)^{p^*}\, B^{d\eta}\, \varepsilon^{-p^*-\eta} \leq C_\eta\, \varepsilon^{-p^*-\eta}. \end{aligned}$$

Clearly, C_η is finite since $B < 1$. By direct calculations we can bound C_η as in (56).

To show the second part of the Lemma, denote $p(x) = -\ln f(x)/\ln g(x)$. Of course,

$$p^* = \max_{x \geq 1/(1-D^2)} p(x).$$

Observe that

$$p'(x) = -\frac{c(x)}{\ln^2 g(x)} \quad \text{with} \quad c(x) = \ln\left(F\frac{x}{x-1}\right)\ln(g(x)) - \ln\left(D\sqrt{\frac{x}{x-1}}\right)\ln(f(x)).$$

Obviously, $c(1/(1-D^2)) = \ln(F/D^2)\ln(C)$ is negative. On the other hand,

$$c(x) = \ln(x)\ln(F^{1/2}/D) + \ln(Fx/(x-1))\ln(C(1-D^2)^{1/2}) - \ln(D(x/(x-1))^{1/2})\ln(F_0(F-1)),$$

which implies $\lim_{x\to\infty} c(x) = +\infty$. This means that the equation $p'(x) = 0$ has a solution x^*. Of course, this equation is equivalent to (59). To show the uniqueness of x^*, note that $c'(x) = -(x(x-1))^{-1}\ln\left(g(x)/\sqrt{f(x)}\right)$ is always positive since $g(x)/\sqrt{f(x)} < 1$ due to (57). This completes the proof. □

Remark 6 Observe that for $p^* \leq 2$ we can set $\eta = 0$ in (55). Indeed, it easily follows from (56), since $C_0 \leq \max_d\{b(d)a^{p^*}(d)\}$ and $b(d)a^{p^*}(d) = O(d^{(p^*-2)/4})$. For $p^* > 2$, we need $\eta > 0$. However, then, the maximum in (56) is attained for $d \leq (p^* - 2)/(4\eta \ln B^{-1})$. □

Remark 7 Theorem 1 and Lemma 8 describe two different definitions of the parameter q for which the algorithm $A_\varepsilon(d) = A(q,d)$ is strongly polynomial-time. As already mentioned, the exponent $p^* + \eta$ given by Lemma 8 is usually smaller. For a fixed d, we can estimate the cost of the algorithm $A_\varepsilon(d)$ in Theorem 1 by

$$\operatorname{cost}(A_\varepsilon(d)) \;\leq\; (c(d)+2)\, C_d\, \varepsilon^{-p^*-\eta},$$

where

$$C_d \;=\; \max_{\varepsilon \leq B^d} \alpha_0(d)\, C^\alpha \left(\alpha_1 + \alpha_2 \frac{\ln(\sqrt{d/(2\pi(d-1))}) + \ln(C/\varepsilon)}{d-1}\right)^{\alpha_3(d-1)} \varepsilon^{p^*+\eta-\alpha}.$$

Clearly, C_d is finite since $p^* > \alpha$. On the other hand, C_d goes to infinity with d. Still, for some small d we can have that $C_d < K_d$.

Hence, if we define $A_\varepsilon(d) = A(q,d)$ with q as in Theorem 1 if $C_d < K_d$, and with q as in Lemma 8 if $C_d \geq K_d$ then

$$\operatorname{cost}(A_\varepsilon(d)) \;\leq\; (c(d)+2)\, \min\{C_d, K_d\}\, \varepsilon^{-p^*-\eta}.$$

As we shall see in Section 8.2, the last estimate will be used to lower the estimates of the cost. □

8 Applications

In this section we illustrate the results of the previous sections for four problems of integration, approximation and a general tensor product problem in the worst case and average case settings. Since integration for continuous functions with the classical Wiener sheet measure is related to discrepancy, we also obtain bounds on discrepancy in the L_2 norm.

Tensor product problems are defined by the one dimensional case for scalar functions. In general, one may consider functions $f : [a, b] \to \mathbb{R}$ for an arbitrary interval $[a, b]$. Clearly, with the obvious change of variables we can assume that $a = 0$ and the new interval becomes $[0, \beta] = [0, b - a]$. That's why we choose to work with functions defined over $[0, \beta]$ with $\beta > 0$. Then for $d \geq 2$, the domain of the functions is $[0, \beta]^d$.

As already mentioned, we are interested in both large and modest d. For large d, we would like to have strong tractability. As we shall see it will depend on β. For modest d, strong tractability is irrelevant and the parameter β does not matter.

8.1 Integration of Smooth Periodic Functions

In this subsection we consider an integration problem. We begin with the average case setting and then comment on the worst case setting.

Let $F_1 = \widetilde{C}^r([0, \beta])$ be the Banach space of periodic r-times continuously differentiable functions with period β and equipped with the norm $\|f\|_{F_1} = \max_{t\in[0,\beta]} |f(t)|$ for $r = 0$, and $\|f\|_{F_1} = |f(0)| + \max_{t\in[0,\beta]} |f^{(r)}(t)|$ for $r \geq 1$. We now explain how a Gaussian measure μ_1 is chosen. First we take w_r as the classical Wiener measure w placed on rth derivatives,

$$w_r(B) \;=\; w(\{f^{(r)} :\; f \in B\}), \quad \forall\, \text{Borel set } B \subset F_1.$$

Recall that w is a Gaussian measure with mean zero and covariance function $R_w(x, t) = \min\{x, t\}$. Observe that the set

$$B \;=\; \{f \in F_1 :\; f^{(j)}(0) = 0,\; j = 0, 1, \ldots, r\}$$

has measure one. Since we deal with periodic functions we also must have $f^{(j)}(\beta) = 0$ for $j = 0, 1, \ldots, r$ with probability 1. To satisfy these boundary conditions, we take the measure μ_1 as the conditional measure $w_r\{\cdot\; |f^{(j)}(\beta) = 0,\; j = 0, 1, \ldots, r\}$.

Let $G_1 = \mathbb{R}$ and

$$S_1(f) \;=\; \int_0^\beta f(t)\, dt, \quad f \in F_1.$$

For $d \geq 2$, the tensor product problem $\{F_d, G_d, S_d\}$ is obtained as in Section 2. That is, F_d is now the Banach space of periodic (in each variable) functions with continuous mixed derivatives $f^{(j_1,j_2,\dots,j_d)}$ for $j_i \leq r$. The measure μ_d is Gaussian with mean zero and covariance function $R_{\mu_d}(x,t) = \prod_{j=1}^d R_{\mu_1}(x_j,t_j)$, where R_{μ_1} is the covariance function of the measure μ_1 and x_j, t_j are the jth components of x and t. With probability 1, the following boundary conditions hold: $f^{(j_1,j_2,\dots,j_d)}(t) = 0$ for all t with at least one component equal to zero or β, $j_i \leq r$. For such functions, $\|f\|_{F_d} = \max_{t\in[0,\beta]^d} |f^{(r,r,\cdots,r)}(t)|$. Clearly, $G_d = \mathbb{R}$ and

$$S_d(f) \;=\; \int_{[0,\beta]^d} f(t)\,dt.$$

We now turn to the algorithm $A(q,d)$. For $d = 1$, we need to define the information N_i and the algorithms U_i. We take

$$N_i(f) \;=\; \left[f\left(\frac{\beta}{m_i+1}\right), f\left(\frac{2\beta}{m_i+1}\right), \dots, f\left(\frac{m_i\beta}{m_i+1}\right)\right]$$

and U_i as the trapezoidal algorithm,

$$U_i(f) \;=\; \frac{\beta}{m_i}\sum_{j=1}^{m_i} f\left(\frac{j\beta}{m_i+1}\right).$$

From Sections 2.1 of Chapters 5 and 7 of [23], it follows that the algorithm U_i is optimal and its average case error is given by

$$e(U_i) \;=\; \|S_1 - U_i\|_{\mu_1} \;=\; \frac{C_r\beta^{(2r+3)/2}}{(m_i+1)^{r+1}}, \quad i \geq 0,$$

where $C_r = \sqrt{|B_{2r+2}|/(2r+2)!}$ with the Bernoulli constant B_{2r+2}. (Recall that $U_0 = 0$.) Observe that for

$$m_i \;=\; 2^i - 1$$

the information N_i is nested, and the assumptions (26), (27) and (42) hold with equalities. Indeed, we have

$$B \;=\; C \;=\; C_r\beta^{(2r+3)/2}, \quad D \;=\; 2^{-(r+1)}, \quad F \;=\; 2, \quad \text{and} \quad F_0 \;=\; 1.$$

Hence, the algorithm $A_\varepsilon(d)$ is strongly polynomial-time iff

$$\beta \;<\; C_r^{-2/(2r+3)}.$$

For instance, for $r = 0$, this holds iff

$$\beta \;<\; 12^{1/3} \;=\; 2.2894\ldots \;.$$

We now estimate the cost of the algorithm $A_\varepsilon(d)$. We first compute the constants which appear in Theorem 1. We have

$$\begin{aligned}
\alpha &= \frac{1}{r+1},\\
\alpha_0(d) &= \left(\frac{d}{2\pi(d-1)}\right)^{\frac{2r+3}{4(r+1)}} 2^{2r+3} \;\le\; \left(\frac{1}{\pi}\right)^{\frac{2r+3}{4(r+1)}} 2^{2r+3},\\
\alpha_1 &= \frac{\alpha_2}{2}\ln\left(\frac{e\,2^{2r+1}C_r^2\beta^{2r+3}}{(r+1)\ln 2}\right),\\
\alpha_2 &= \frac{e^2\beta}{(e-1)(r+1)\ln 2}\,C_r^{\frac{2}{2r+3}},\\
\alpha_3 &= \frac{2r+3}{2(r+1)}.
\end{aligned}$$

Then we can use the estimates of Theorems 1 and 2 with these constants.

We specialize these estimates for $r = 0$ and assuming for simplicity that $\beta = 1$. We now have

$$B = C = 1/(2\sqrt{3}) = 0.288675\ldots, \quad D^{-1} = F = 2, \quad F_0 = 1,$$

and

$$\begin{aligned}
\alpha &= 1,\\
\alpha_0(d) &= 8\left(\frac{d}{2\pi(d-1)}\right)^{\frac{3}{4}} \;\le\; 8\left(\frac{1}{\pi}\right)^{\frac{3}{4}} \;=\; 3.39021\ldots,\\
\alpha_1 &= \left(\frac{1}{12}\right)^{1/3}\frac{e^2}{2(e-1)\ln 2}\,\ln\frac{e}{6\ln 2} \;=\; -0.57617289\ldots,\\
\alpha_2 &= \left(\frac{1}{12}\right)^{1/3}\frac{e^2}{(e-1)\ln 2} \;=\; 2.7098298\ldots,\\
\alpha_3 &= 1.5.
\end{aligned}$$

As explained before, we need only to consider $\varepsilon < B^d = 12^{-d/2}$. This inequality corresponds to modest d or, if d is large, to an unusually high precision. For $d \ge 2$, Theorem 1 yields

$$\mathrm{cost}(A_\varepsilon(d)) \le 0.9787\,(c(d)+2)\left(-0.576 + 2.71\,\frac{-1.8148+\ln 1/\varepsilon}{d-1}\right)^{1.5(d-1)}\frac{1}{\varepsilon}.$$

It is known[3] that the average case complexity of computing an ε-approximation for this integration problem is $\Theta(\varepsilon^{-1}(\ln \varepsilon^{-1})^{(d-1)/2})$ where the factors in the Θ notation depend on d. Hence, for fixed d and ε tending to zero the cost of the algorithm $A_\varepsilon(d)$ agrees with the leading term of the complexity, however, the exponent of $\ln 1/\varepsilon$ is too large.

Computing the constants K and p from Theorem 2 we obtain for $\varepsilon \le 12^{-d/2}$,

$$\mathrm{cost}(A_\varepsilon(d)) \le 0.2064\ (c(d)+2)\ \varepsilon^{-2.253}. \tag{61}$$

We now comment on the last estimate. Since the covariance function $R_{\mu_d}(t,t)$ is uniformly bounded in d, it is well known that the complexity of computing an ε-approximation for integration in the average case setting is $O(\varepsilon^{-2})$ with the factor in the O notation independent of d. This means that (61) is not satisfactory. Indeed, it is possible to improve this bound. It can be verified that we can now use (40) of Lemma 6. This formally corresponds to replacing C by $C(1-D^2)^{1/2} = 1/4$. Computing p for this new value of C we get

$$\mathrm{cost}(A_\varepsilon(d)) = O\left((c(d)+2)\,\varepsilon^{-2.0569}\right)$$

which is better but still not satisfactory. (Of course, the factor in the O notation does not depend on d.) This can be improved by using Lemma 8. Indeed, using Newton's iteration we found out that $p^* \le 1.850698$. Hence, as explained in Remark 6, we can take $\eta = 0$. We computed C_0 to get

$$\mathrm{cost}(A_\varepsilon(d)) \le 1.28068\,(c(d)+2)\,\varepsilon^{-1.850698}.$$

We now briefly comment on the worst case setting. It is well known that for integration (as well as for arbitrary linear functional S_1), the average case and worst case settings are related. That is, the average case setting for the space F_1 with the measure μ_1 corresponds to the worst case setting for the unit ball of the reproducing kernel Hilbert space H_1 whose reproducing kernel is the covariance function of the measure μ_1, see e.g., [27]. For arbitrary β, the space H_1 is the Sobolev space of periodic functions vanishing at 0 and β whose rth derivatives are absolutely continuous and $f^{(r+1)}$ is in $L_2([0,\beta])$ with the norm $\|f\|_{H_1} = \left(\int_0^\beta f^{(r+1)}(t)^2dt\right)^{1/2}$. Hence, all the estimates presented in this subsection are also valid for the worst case with the tensor product problem generated by $\{H_1, \mathbb{R}, S_1\}$.

[3] This follows from the fact that periodicity does not change the dependence on ε, and without periodicity the bound on the average case complexity is derived in [29].

8.2 Integration of Nonperiodic Functions

In this subsection we also consider integration in the average case setting. We define $\{F_1, G_1, S_1\}$ as in Subsection 8.1 with $r = 0$ but without assuming periodicity of functions. That is, $F_1 = C([0,\beta])$ is now the Banach space of continuous functions equipped with the norm $\|f\|_{F_1} = \max_{t\in[0,\beta]} |f(t)|$. As the measure μ_1 we take the Wiener measure $w_0 = w$.

For $d \geq 2$, the Banach space F_d is the class of continuous functions with the sup norm, the measure μ_d is the classical Wiener sheet measure which is Gaussian with mean zero and covariance function $R_{\mu_d}(x,t) = \prod_{j=1}^d \min\{x_j, t_j\}$.

To define the algorithm $A(q,d)$, we take for $d = 1$ the information

$$N_i(f) \;=\; \left[f\left(\frac{2\beta}{2m_i+1}\right), f\left(\frac{4\beta}{2m_i+1}\right), \dots, f\left(\frac{2m_i\beta}{2m_i+1}\right)\right] \tag{62}$$

and the algorithms

$$U_i(f) \;=\; \frac{2\beta}{2m_i+1} \sum_{j=1}^{m_i} f\left(\frac{2j\beta}{2m_i+1}\right). \tag{63}$$

It is known, see [9], that the algorithm U_i is optimal and its average case error is

$$e(U_i) \;=\; \|S_1 - U_i\|_{\mu_1} \;=\; \frac{\beta^{3/2}}{\sqrt{3}(2m_i+1)}, \quad i \geq 0.$$

Observe that for

$$m_i \;=\; \frac{1}{2}\left(3^i - 1\right)$$

the information N_i is nested, and the assumptions (26) and (27) hold with equalities. Indeed, we have

$$B \;=\; C \;=\; \frac{\beta^{3/2}}{\sqrt{3}}, \quad D \;=\; 1/3, \quad F \;=\; 3, \quad \text{and} \quad F_0 \;=\; 1/2.$$

Hence, the algorithm $A_\varepsilon(d)$ is strongly polynomial-time iff

$$\beta \;<\; 3^{1/3} \;=\; 1.4422\dots\,.$$

Assume for simplicity that $\beta = 1$. To estimate the cost of the algorithm, we first compute the constants of Theorem 1. We have

$$\alpha \;=\; 1,$$

$$
\begin{aligned}
\alpha_0(d) &= 13.5\left(\frac{d}{2\pi(d-1)}\right)^{\frac{3}{4}} \leq 13.5\left(\frac{1}{\pi}\right)^{\frac{3}{4}} = 5.72099...,\\
\alpha_1 &= \frac{e^2}{2\cdot 3^{1/3}(e-1)\ln 3}\ln\frac{3\,e}{2\ln 3} = 1.77958...,\\
\alpha_2 &= \frac{e^2}{3^{1/3}(e-1)\ln 3} = 2.71399...,\\
\alpha_3 &= 1.5.
\end{aligned}
$$

Similarly as in Subsection 8.1, for $\varepsilon \geq 3^{-d/2}$ the problem is trivial. For $\varepsilon < 3^{-d/2}$ and $d \geq 2$, Theorem 1 yields

$$
\mathrm{cost}(A_\varepsilon(d)) \leq 3.304\,(c(d)+2)\left(1.77959 + 2.714\,\frac{-1.12167+\ln 1/\varepsilon}{d-1}\right)^{1.5(d-1)}\frac{1}{\varepsilon}.
$$

We compare this bound with the average case complexity $\Theta(\varepsilon^{-1}(\ln \varepsilon^{-1})^{(d-1)/2})$, where the factors in the Θ notation depend on d, see [29]. As in Subsection 8.1, the exponent of $1/\varepsilon$ in the cost estimate of the algorithm $A(q,d))$ agrees with the power of $1/\varepsilon$ in the average case complexity, however, the power of $\ln 1/\varepsilon$ is too large.

Computing the constants K and p from Theorem 2 we obtain for $\varepsilon \leq 3^{-d/2}$,

$$
\mathrm{cost}(A_\varepsilon(d)) \leq 0.558477\;(c(d)+2)\;\varepsilon^{-4.23568}.
$$

As in Subsection 8.1 the last estimate is not satisfactory since the average case complexity of computing an ε-approximation is $O(\varepsilon^{-2})$ with the factor in the O notation independent of d. As before it is possible to improve this bound. For instance, by using (40) of Lemma 6 we can replace C by $C(1-D^2)^{1/2} = 2\sqrt{2}/(3\sqrt{3})$ to get

$$
\mathrm{cost}(A_\varepsilon(d)) = O\left((c(d)+2)\,\varepsilon^{-4}\right).
$$

The exponent p can be lowered by using a modified Lemma 8. We computed the corresponding p^* and found out that $p^* \leq 2.452616$. We also computed C_η and K_d for $\eta = 10^{-3}$ following Remark 6 and we got

$$
\mathrm{cost}(A_\varepsilon(d)) \leq K_d\,(c(d)+2)\,\varepsilon^{-2.454},
$$

where $K_1 = 0.39$ and $\{K_d\}$, for $d \geq 2$, is monotonically decreasing to zero, and $C_\eta = K_2 = 12.59$, $K_3 = 8.9$, $K_4 = 7.26$ and $K_5 = 6.29$.

Following Remark 7, we can improve these estimates for $d = 2$ and $d = 3$. Indeed, we computed $C_2 = 1.76$ and $C_3 = 4.76$ of Remark 7. Hence, if we define $A_\varepsilon(d) = A(q,d)$ with q from Theorem 1 for $d = 2$ and $d = 3$, and with q from Lemma 8 for $d \geq 4$ then

$$\mathrm{cost}(A_\varepsilon(d)) \leq\ 7.26\,(c(d)+2)\,\varepsilon^{-2.454}, \quad \forall\, d, \varepsilon \leq 1.$$

Still the exponent is larger than 2.

We tried a number of different choices of the parameters m_i as well as different information N_i and algorithms U_i. For $m_i \approx F^i - 1$ with noninteger $F \in (1,2)$, we obtained exponents smaller than 2.45; however, still larger than two. We do not present the corresponding analysis since it is very messy and, more importantly, since we do believe that there exists a choice of N_i, m_i, and U_i for which the exponent p is less than two. We leave it as a challenging problem for future work.

As explained in Subsection 8.1, we can use the estimates of this subsection also for the corresponding worst case integration problem. For arbitrary β, this integration problem is now given by the Sobolev space H_1 which consists of absolutely continuous functions vanishing at zero with square integrable first derivative. The norm is given by $\|f\|_{H_1} = \left(\int_0^\beta f'(t)^2 dt\right)^{1/2}$.

8.3 Discrepancy

We utilize relations between integration in the average case setting with the Wiener sheet measure and discrepancy in the L_2 norm, see [11] for the definition and basic properties of discrepancy. It was shown in [29] that the minimal discrepancy (with optimal weights) of n points in the d dimensional unit cube is equal to the minimal average case error of integration algorithms. This, of course, allows us to apply the bounds obtained in Subsection 8.2 to discrepancy.

More precisely, take N_i and U_i as in (62) and (63) with $\beta = 1$ and $m_i = (3^i - 1)/2$. Then (15) yields

$$A(q,d)f \;=\; \sum_{\vec{i}\in P(q,d)} c_{\vec{i}} \sum_{\vec{j}\leq \vec{m}_{\vec{i}}} f\left(x_{\vec{i},\vec{j}}\right),$$

where $P(q,d) \;=\; \left\{\vec{i} \,:\, \vec{1} \,\leq\, \vec{i},\; q-d+1 \,\leq\, |\vec{i}\,| \,\leq\, q\right\}$,

$$c_{\vec{i}} \;=\; \frac{(-1)^{q-|\vec{i}|}\,2^d}{3^{|\vec{i}|}} \begin{pmatrix} d-1 \\ q-|\vec{i}\,| \end{pmatrix} \quad \text{and} \quad x_{\vec{i},\vec{j}} \;=\; \left[\frac{2\,j_1}{3^{i_1}}, \frac{2\,j_2}{3^{i_2}}, \ldots, \frac{2\,j_d}{3^{j_d}}\right]$$

with $\vec{m}_{\vec{i}} = [(3^{i_1}-1)/2, (3^{i_2}-1)/2, \ldots, (3^{i_d}-1)/2]$.

Let $n = m(q,d)$ be the number of function values used by the algorithm $A(q,d)$; we have due to (44),

$$n \;=\; \sum_{j=0}^{q-d} 3^j \binom{j+d-1}{d-1}.$$

Define n points $z_{\vec{i},\vec{j}} = \vec{1} - x_{\vec{i},\vec{j}}$ and let

$$\mathrm{DISC}_n(t) \;=\; \sum_{\vec{i}\in P(q,d)} c_{\vec{i}} \sum_{\vec{j}\,\le\,\vec{m}_{\vec{i}}} \chi_{[0,t)}(z_{\vec{i},\vec{j}}) \;-\; t_1 t_2 \cdots t_d,$$

where $\chi_{[0,t)}$ is the characteristic (indicator) function of $[0,t) = [0,t_1) \times [0,t_2) \times \cdots \times [0,t_d)$. The discrepancy of the n points $z_{\vec{i},\vec{j}}$ is given by

$$\|\mathrm{DISC}_n\| \;=\; \left(\int_{[0,1]^d} \mathrm{DISC}_n^2(t)\,dt\right)^{1/2}$$

It is known that

$$\|\mathrm{DISC}_n\| \;=\; e(A(q,d)).$$

As already discussed in the introduction, for $n = 0$ we have $\|\mathrm{DISC}_0\| = 3^{-d/2}$. Hence, the above equality also holds for $q < d$. From Lemma 6 we have

$$\|\mathrm{DISC}_n\| \;\le\; \left(\frac{1}{\sqrt{3}}\right)^d \left(\frac{1}{3}\right)^{q-d+1} \sqrt{\binom{q}{d-1}}.$$

If we choose q to guarantee that $e(A(q,d)) \le \varepsilon \le 3^{-d/2}$ as in Theorem 1 then

$$\|\mathrm{DISC}_n\| \;\le\; \varepsilon$$

for $n = n(\varepsilon,d)$ such that

$$n(\varepsilon,d) \;\le\; 3.304\, \left(1.77959 \;+\; 2.714\, \frac{-1.12167 + \ln 1/\varepsilon}{d-1}\right)^{1.5(d-1)} \frac{1}{\varepsilon}.$$

From Subsection 8.2 we also have

$$n(\varepsilon,d) \;\le\; \gamma_d\, \varepsilon^{-2.454} \;\le\; 7.26\, \varepsilon^{-2.454}, \quad \forall\, d, \varepsilon \le 1,$$

with $\gamma_1 = 0.39$, $\gamma_2 = 1.76$, $\gamma_3 = 4.76$, $\gamma_4 = 7.26$ and $\{\gamma_d\}$ monotonically decreasing to zero starting with $d = 4$.

For every ε, there exist $n(\varepsilon) = O(\varepsilon^{-2})$ points with discrepancy at most ε for all d. Construction of these points is open.

8.4 Approximation Using Function Values

In this subsection we consider the following function approximation problem in the average case setting. We define the spaces F_d and the measures μ_d as in Subsection 8.2, i.e., F_d is the class of continuous functions defined over $[0,\beta]^d$ with the sup norm, and μ_d is the classical Wiener sheet measure. We let $G_d = L_2([0,\beta]^d)$ and

$$S_d(f) \;=\; f.$$

For $d=1$, we take the information

$$N_i(f) \;=\; \left[f\left(\frac{3\beta}{3m_i+1}\right), f\left(\frac{6\beta}{3m_i+1}\right), \ldots, f\left(\frac{3m_i\beta}{3m_i+1}\right)\right]$$

and the algorithms

$$U_i(f) \;=\; \sum_{j=1}^{m_i} f\left(\frac{3j\beta}{3m_i+1}\right) g_{i,j},$$

where $g_{i,j}$ are chosen in such a way that $U_i(f)$ is a piecewise linear function passing through $f(0)=0$ and $f(3j\beta/(3m_i+1))$. For $t \in [3m_i\beta/(3m_i+1), 1]$ we set $U_i(f)(t) \equiv f(3m_i\beta/(3m_i+1))$.

It is known, see [9], that the algorithm U_i is optimal and its average case error is

$$e(U_i) \;=\; \|S_1 - U_i\|_{\mu_1} \;=\; \frac{\beta}{\sqrt{2(3m_i+1)}}, \quad i \geq 0.$$

Observe that for

$$m_i \;=\; \frac{1}{3}\left(4^i - 1\right)$$

the information N_i is nested, and the assumptions (26) and (27) hold with equalities. Indeed, we have

$$B \;=\; C \;=\; \frac{\beta}{\sqrt{2}} \quad D \;=\; 1/2, \quad F \;=\; 4, \quad \text{and} \quad F_0 = 1/3.$$

Hence, the algorithm $A_\varepsilon(d)$ is strongly polynomial-time iff

$$\beta \;<\; \sqrt{2} \;=\; 1.4142\ldots\,.$$

Assume for simplicity that $\beta = 1$. To estimate the cost of the algorithm we first compute the constants of Theorem 1. We have

$$\begin{aligned}
\alpha &= 2, \\
\alpha_0(d) &= \frac{16}{3}\left(\frac{d}{2\pi(d-1)}\right) \le \frac{16}{3\,\pi} = 1.69765..., \\
\alpha_1 &= \frac{e^2}{2\sqrt{2}\,(e-1)\ln 2}\ln\frac{e}{\ln 2} = 2.99735..., \\
\alpha_2 &= \frac{e^2}{\sqrt{2}\,(e-1)\ln 2} = 4.38686..., \\
\alpha_3 &= 2.
\end{aligned}$$

Hence, for $d \ge 2$ and $\varepsilon \le 2^{-d/2}$, Theorem 1 yields

$$\mathrm{cost}(A_\varepsilon(d)) \le 0.8489\,(c(d)+2)\left(2.9974 + 4.3869\,\frac{-0.9189 + \ln 1/\varepsilon}{d-1}\right)^{2(d-1)}\left(\frac{1}{\varepsilon}\right)^2.$$

We compare this bound with the average case complexity $\Theta(\varepsilon^{-2}(\ln \varepsilon^{-1})^{2(d-1)})$, where the factors in the Θ notation depend on d, see [30]. Hence, the exponents of $1/\varepsilon$ as well as of $\ln 1/\varepsilon$ in the cost estimate of the algorithm $A_\varepsilon(d)$ agree with the corresponding exponents in the average case complexity.

Computing the constants K and p from Theorem 2 we obtain for $\varepsilon \le 3^{-d/2}$,

$$\mathrm{cost}(A_\varepsilon(d)) \le 0.041588\;(c(d)+2)\;\varepsilon^{-10.705}.$$

As before it is possible to improve this bound. For instance, by using (40) of Lemma 6 we replace C by $C(1-D^2)^{1/2} = \sqrt{3}/(2\sqrt{2})$ and we get

$$\mathrm{cost}(A_\varepsilon(d)) = O\left((c(d)+2)\,\varepsilon^{-9.0042}\right).$$

The exponent p can be once more lowered by using a modified Lemma 8. We now need to divide $b(d)$ and the argument of the logarithm in the numerator of (54) by 3. Then we get $p^* \le 5.661135$, and for $\eta = 10^{-2}$,

$$\mathrm{cost}(A_\varepsilon(d)) \le 2.37632\,(c(d)+2)\,\varepsilon^{-5.672}, \quad \forall\, d, \varepsilon \le 1.$$

The minimal value of the exponent p is not known. Of course, it must be a number from the interval $(2, 5.672]$. Unlike for integration, a better upper bound on the exponent p for approximation is unknown.

8.5 Arbitrary Linear Functionals

In the previous subsections we considered specific tensor product problems and algorithms that use information consisting of function values. In this subsection we deal with a general tensor product problem and algorithms that use information consisting of arbitrary continuous linear functionals. As already mentioned in the introduction, in this case, optimal information and algorithms are known. Nevertheless, we wish to verify the quality of the algorithm $A_\varepsilon(d)$ by comparing them with the optimal algorithms.

We consider the worst case setting for an arbitrary tensor product problem for which the operator $W_1 = (S_1^* S_1)^{1/2}$ has the eigenvalues

$$\lambda_n \;=\; B\, n^{-r}, \quad n \;=\; 1, 2 \ldots$$

for some positive r. As in Subsection 4.1.2, we take for $d = 1$ the optimal information N_i and optimal algorithms U_i, see (22).

For $m_i = F^i - 1$, Lemma 4 yields that the error of the algorithm $A(q,d)$ for arbitrary d is given by

$$e(A(q,d)) \;=\; B^d\, F^{-r(q-d+1)}.$$

To guarantee that the error is at most $\varepsilon \le B^d$ we take

$$q \;=\; \left\lceil d - 1 \;+\; \frac{\ln\, B^d/\varepsilon}{r\, \ln\, F} \right\rceil .$$

Proceeding exactly as in Sections 5 and 6 one can show that the cost is bounded by

$$\mathrm{cost}(A_\varepsilon(d)) \;\le\; (c(d)+2)\,\frac{F}{\sqrt{\pi}} \left(\alpha_1 + \alpha_2 \frac{r\, \ln\, F \,+\, \ln(B/\varepsilon)}{d-1} \right)^{d-1} \left(\frac{B}{\varepsilon} \right)^{1/r},$$

where

$$\begin{aligned} \alpha_1 &\;=\; B^{1/r}\, e\,(F-1) \left(1 + \frac{\ln\, B}{r\, \ln\, F} \right), \\ \alpha_2 &\;=\; \frac{B^{1/r}}{r\, \ln\, F}\, e\,(F-1). \end{aligned}$$

We now compare this estimate with the worst case complexity, see [31],

$$\mathrm{comp}(\varepsilon, d) \;=\; \frac{B^{d/r}}{r^{d-1}(d-1)!} \left(\frac{1}{\varepsilon} \right)^{1/r} \left(\ln\, \frac{1}{\varepsilon} \right)^{d-1} (1 + o(1))\,, \quad \text{as } \varepsilon \to 0.$$

Hence, the exponents of $1/\varepsilon$ and $\ln 1/\varepsilon$ are the same. Even though the algorithm $A_\varepsilon(d)$ is not using the optimal information for $d \geq 2$, its cost differs asymptotically in ε only by a multiplicative constant. Furthermore, even the asymptotic constants are related. Indeed, their ratio is roughly

$$\sqrt{2\pi}\,\left(\frac{F-1}{r\,\ln\,F}\right)^{d-1}.$$

Hence, if we can choose $F - 1 \simeq r \ln F$ then they are comparable.

9 Appendix

In this appendix we show some relations between the concepts of tractability and strong tractability which have been mentioned in the introduction and in Section 7.

(i) First, we show that if S_1 is not a linear functional then tractability and strong tractability coincide for tensor product problems. This holds in the worst case and average case settings for both classes Λ^{all} and Λ^{std}. Obviously, it is enough to show that tractability implies strong tractability.

We begin with the worst case setting. For the class Λ^{all} it is proven in [31]. For the class Λ^{std} we proceed as follows. Tractability in Λ^{std} implies tractability in Λ^{all}. The latter yields that $B = \|S_1\| < 1$, see (i) of Theorem 3.1 of [31]. Tractability in Λ^{std} means that for $d = 1$ there exists a one dimensional algorithm U_n which uses n function values with error $O(n^{-p})$ for some positive p. Hence, we can apply the construction of Section 3 with the one dimensional algorithm U_i to get a strongly polynomial-time algorithm $A_\varepsilon(d)$. This proves that the problem is strongly tractable.

For the average case setting, consider first the class Λ^{all}. Let $\gamma_{1,d} \geq \gamma_{2,d} \geq \cdots \geq 0$ be eigenvalues of the covariance operator C_{ν_d} of the measure $\nu_d = \mu_d S_d^{-1}$. Clearly, for tensor product problems, we have $\gamma_{j,d} = \gamma_{i_1,1}\gamma_{i_2,1}\cdots\gamma_{i_d,1}$ for some indices i_j. Hence,

$$B^{2d} \;=\; \|S_1\|_{\mu_1}^{2d} \;=\; \|S_d\|_{\mu_d}^{2} \;=\; \left(\sum_{j=1}^{\infty}\gamma_{j,1}\right)^{d} \;=\; \sum_{j=1}^{\infty}\gamma_{j,d}.$$

Observe that $\gamma_{1,d} = \gamma_{1,1}^d$ and since S_1 is not a linear functional, $\gamma_{2,1} > 0$.

We show that tractability in Λ^{all} implies $B < 1$. Suppose on the contrary that $B \geq 1$. It is enough to consider the case $B = 1$. Then $\gamma_{1,1} < 1$. Let $e^*(n,d)$ be the nth minimal average case error for the d dimensional case. It is known that

$$e^*(n,d)^2 \;=\; \sum_{j=1}^{\infty}\gamma_{j,d} - \sum_{j=1}^{n}\gamma_{j,d} \;\geq\; 1-\gamma_{1,d}n \;=\; 1-\gamma_{1,1}^{d}n.$$

Thus, to guarantee that, say, $e^*(n,d) \leq 1/\sqrt{2}$ we have to take $n \geq \gamma_{1,1}^{-d}/2$. This contradicts tractability. Hence, $B<1$, as claimed.

Tractability in Λ^{all} implies that for $d=1$ we have $\gamma_{j,1}=O(j^{-1-k})$ for some positive k, see Theorem 5.1 of [32]. We now take a one dimensional algorithm U_n that uses n inner products such that its average case error is $\sqrt{\sum_{j=n+1}^{\infty}\gamma_{j,1}} \;=\; O(n^{-k/2})$. Since $B<1$, the construction of Section 3 yields a strongly polynomial-time algorithm. This means that the problem is strongly tractable.

This also proves that for the class Λ^{all} in the average case setting, strong tractability holds iff

$$\|S_1\|_{\mu_1}^2 = \sum_{n=1}^{\infty}\gamma_{n,1} < 1 \quad \text{and} \quad \gamma_{n,1}=O(n^{-1-k})$$

for some positive k.

For the class Λ^{std} we use the obvious fact that tractability in Λ^{std} implies tractability in Λ^{all}. This yields $B<1$. The existence of a one dimensional algorithm that uses n function values with the average case error of order n^{-k} for some positive k follows from tractability in Λ^{std} for $d=1$. The rest of the proof is the same as for the class Λ^{all}.

(ii) We now show that if S_1 is a linear functional then $\|S_1\|_{\mathrm{set}} \geq 1$ may or may not yield strong tractability. Of course, this holds only for the class Λ^{std} since for Λ^{all} the problem is trivially strongly tractable. Due to equivalence between the worst and average case settings for linear functionals, it is enough to consider only the worst case setting.

Clearly, if $S_1(f)=\alpha f(t^*)$ then $S_d(f)=\alpha^d f(t^*,t^*,\dots,t^*)$ and the problem can be solved by using just one function value for all d. This holds for all α and the norm of S_1 can be arbitrary large for sufficiently large α.

We now present an example of a linear functional S_1 with norm 1 which yields an intractable tensor product problem.

Let h_1 and h_2 be defined on $[0,1]$ with the support $(0,1/2)$ and $(1/2,1)$, respectively. Take $F_1=\mathrm{lin}\{h_1,h_2\}$ with inner-product defined by the condition that h_i are orthonormal. Define the linear functional S_1 such that $Sh_i=1/\sqrt{2}$. This yields $B=\|S_1\|=1$.

For the d dimensional case, F_d is spanned by 2^d functions $h_{\vec{i}}$ ($i_k\in\{1,2\}$), each with a different support. Clearly, $S_d h_{\vec{i}} = 2^{-d/2}$. If we use information $N(f)=[f(x_1),f(x_2),\dots,f(x_n)]$ of cardinality $n\leq 2^d$ then we do not sample in 2^d-n regions. Let I be the set of indices

$\vec{i}$ for which we do not sample $h_{\vec{i}}$. Letting $f^* = \sum_{\vec{i}\in I} c_{\vec{i}} h_{\vec{i}}$ with $c_{\vec{i}} = c = 1/\sqrt{2^d - n}$, we have $\|f^*\| = 1$ and $f^*(x_j) = 0$. It is known that the worst case error of any algorithm that uses the information N is at least

$$S_d f = (2^d - n)c/2^{d/2} = \sqrt{1 - n/2^d}.$$

This means that to compute an ε-approximation, $\varepsilon < 1$, we need to use at least $\lceil 2^d(1-\varepsilon^2)\rceil$ function values. Hence, the problem is intractable.

(iii) Finally, we show that if S_1 is a linear functional which cannot be represented as a finite combination of function evaluations and $\|S_1\| = 1$ then the tensor product problem is not strongly tractable. As in (ii) it is enough to consider the worst case setting. Let $S_1 f = \langle f, h\rangle$ with $\|h\| = 1$, and let $e^*(n,1)$ denote the minimal worst case error of approximating S_1 using function values at n optimally chosen points. Our assumption means that $e^*(n,1) > 0$.

For arbitrary d, we have $S_d f = \langle f, h_d\rangle$ with $h_d(t) = h(t_1)\cdots h(t_d)$. Let $e^*(n,d)$ denote the minimal worst case error of approximating S_d using $[f(x_1), f(x_2), \dots, f(x_n)]$ for optimally chosen points x_i in the d dimensional case. We prove that for any fixed n

$$\liminf_{d\to\infty} e^*(n,d) \ge 1.$$

This, of course, implies that the problem is not strongly tractable.

It is known that

$$e^*(n,d) \;=\; \sup_{f:\ f(x_j)=0, j=1,2,\dots,n, \|f\|=1} |\,\langle f, h_d\rangle\,|.$$

We select a special f to bound $e^*(n,d)$ from below. Let $x_{j,k}$ denote the kth component of x_j. Choose $f_k : [0,1] \to \mathbb{R}$ such that $f_k(x_{j,k}) = 0$ for $j = 1, 2\dots, n$, $\|f_k\| = 1$ and $e_k = S_1 f \ge e^*(n,1)$. Define

$$g(t) \;=\; f_1(t_1)h(t_2)\cdots h(t_d) + h(t_1)f_2(t_2)h(t_3)\cdots h(t_d) + \cdots + h(t_1)\cdots h(t_{d-1})f_k(t_d).$$

Observe that g vanishes at any x_j. We have

$$\langle g, h_d\rangle \;=\; \sum_{k=1}^d e_k \qquad\text{and}\qquad \|g\|^2 \;=\; d + \sum_{i\ne j} e_i e_j \;\le\; d + \left(\sum_{k=1}^d e_k\right)^2.$$

Finally, take $f = g/\|g\|$. Then $f(x_j) = 0$, $\|f\| = 1$, and $\langle f, h_d\rangle \ge a/\sqrt{d+a^2}$ with $a = \sum_{k=1}^d e_k$. Since $a/\sqrt{d+a^2}$ is an increasing function of a and $a \ge d e^*(n,1)$, we conclude that

$$\langle f, h_d\rangle \;\ge\; \frac{d\, e^*(n,1)}{\sqrt{d + d^2 e^*(n,1)^2}} \to 1 \quad \text{as } d\to\infty.$$

This completes the proof.

(iv) We add one more reason why the concept of strong tractability for approximating a sequence of linear operators $\{W_d\}$ with the normalized condition $\|W_d\| = 1, \ \forall\, d$, may lead to some strange conclusions. Indeed, suppose we have a tensor product problem $\{S_d\}$ with $\alpha = \|S_1\| \in (0,1)$. If we define $W_d = \alpha^{-d} S_d$ then the normalized condition is satisfied and, as already explained, the problem $\{W_d\}$ is *not* strongly tractable. However, we may proceed differently. For every d we choose a point x_d from the domain of functions at which not all functions vanish. Define $W_d(f) = \alpha_d f(x_d) + S_d(f)$, where α_d is chosen such that the normalized condition $\|W_d\| = 1$ is satisfied. Then the problem $\{W_d\}$ *is* strongly tractable. Indeed, it is enough to take the algorithm $\alpha_d f(x_d) + A_\varepsilon(d)(f)$, where $A_\varepsilon(d)$ is a strongly polynomial-time algorithm of Section 7 for the nonnormalized problem $\{S_d\}$. In both cases, we are really approximating a sequence $\{S_d\}$; however, depending on the way how the normalized condition is satisfied, we obtain an intractable or strongly tractable problem.

References

[1] N. Aronszajn, Theory of reproducing kernels, *Trans. Amer. Math. Soc.*, 68, p. 337-404, 1950.

[2] K. I. Babenko, On the approximation of a class of periodic functions of several variables by trigonometric polynomials, *Dokl. Akad. Nauk SSSR* (English trans. in *Soviet Math. Dokl*, 1, 1960), 132, p. 247-250, 982-985, 1960.

[3] J. Beck and W. W. L. Chen, Irregularities of Distribution, *Cambridge University Press*, Cambridge, 1987.

[4] B. Chazelle, Geometric Discrepancy Revisited, *34th Annual Symposium on Foundations of Computer Science*, p. 392-399, 1993.

[5] D. P. Dobkin and D. P. Mitchell, Random-Edge Discrepancy of Supersampling Patterns, *Graphics Interface '93*, York, Ontario, 1993.

[6] I. S. Gradshteyn and I. Ryzhik, Table of Integrals, Series and Products, *Academic Press*, New York, 1980.

[7] S. Heinrich, Complexity of integral equations and relations to s-numbers, *J. Complexity*, 9, p. 141-153, 1993.

[8] H.-H. Kuo, H.-H. Gaussian Measures in Banach Spaces, *Lectures Notes in Mathematics*, 463, *Springer Verlag*, Berlin, 1975.

[9] D. Lee, Approximation of linear operators on a Wiener space, *Rocky Mount. J. Math*, 16, p. 641-659, 1986.

[10] C. A. Micchelli and T. J. Rivlin , A survey of optimal recovery, in *Optimal Estimation in Approximation Theory* (C. A. Micchelli and T.J. Rivlin, eds.) p. 1-54, *Plenum Press*, New York, 1977.

[11] H. Niederreiter, Random number generator and quasi-Monte Carlo methods, *CBMS-NSF Reg. Conf. Series Appl. Math.*, 63, *SIAM*, Philadelphia, 1992.

[12] A. Papageorgiou and G. W. Wasilkowski, On the average complexity of multivariate problems, *J. Complexity*, 6, p 1-23, 1990.

[13] S. Paskov, Average case complexity of multivariate integration for smooth functions, *J. Complexity*, 9, p.291-312, 1993.

[14] S. V. Pereverzev, On the complexity of the problem of finding solutions of Fredholm equations of the second kind with differentiable kernels (in Russian), *Ukrain. Mat. Sh.*, 41, p. 1422-1425, 1989.

[15] K. Ritter, G. W. Wasilkowski, and H. Woźniakowski, On multivariate integration for stochastic processes, in *Numerical Integration* (H. Braß, G. Hämmerlin, eds.), p. 331-347, International Series of Numerical Mathematics, vol. 112, *Birkhäuser*, Basel, 1993.

[16] K. Ritter, G. W. Wasilkowski, and H. Woźniakowski, Multivariate integration and approximation for random fields satisfying Sacks-Ylvisaker conditions, *submitted for publication*, 1993.

[17] K. F. Roth, On irregularities of distribution, *Mathematika*, 1, p. 73-79, 1954.

[18] K. F. Roth, On irregularities of distribution, IV, *Acta Arith.*, 37, p. 67-75, 1980.

[19] S. A. Smolyak, Quadrature and interpolation formulas for tensor products of certain classes of functions, *Dokl. Akad. Nauk SSSR*, p. 240-243, 1964.

[20] V. N. Temlyakov, Approximate recovery of periodic functions of several variables, *Math. USSR Sbornik*, 56, p. 249-261, 1987.

[21] V. N. Temlyakov, On a way of obtaining lower estimates for the errors of quadrature formulas errors, *Math. USSR Sbornik*, 181, p. 1403-1413, 1990 (in Russian), *Math. USSR Sbornik*, 71, p. 247-257, 1992 (in English).

[22] V. N. Temlyakov, On approximate recovery of functions with bounded mixed derivative, *J. Complexity*, 9, p. 41-59, 1993.

[23] J. F. Traub, G. W. Wasilkowski, and H. Woźniakowski, Information-based Complexity, *Academic Press*, New York, 1988.

[24] J. F. Traub and H. Woźniakowski, A General Theory of Optimal Algorithms, *Academic Press*, New York, 1980.

[25] N. N. Vakhania, V. I. Tarieladze, and S. A. Chobanyan, Probability Distributions on Banach Spaces, *Reidel, Dordrecht*, 1987.

[26] G. Wahba, Interpolating Surfaces: High Order Convergence Rates and Their Associated Designs, with Application to X-Ray Image Reconstruction, *Dept. of Statistics*, University of Wisconsin, 1978.

[27] G. Wahba, Spline Models for Observational Data, *CBMS-NSF Reg. Conf. Series Appl. Math.*, 59, *SIAM*, Philadelphia, 1990.

[28] G. W. Wasilkowski, Information of varying cardinality, *J. of Complexity*, 2, p. 204-228, 1986.

[29] H. Woźniakowski, Average Case Complexity of Multivariate Integration, *Bull. Amer. Math. Soc. (N.S)*, 24, p. 185-194, 1991.

[30] H. Woźniakowski, Average case complexity of linear multivariate problems, Part 1: Theory, Part 2: Applications, *J. Complexity*, 8, p. 337-372, 373-392, 1992.

[31] H. Woźniakowski, Tractability and strong tractability of multivariate tensor product problems, in Proceedings of 2nd Gauss Symposium, Muenchen, 1993, to appear in *J. Computing and Information.*

[32] H. Woźniakowski, Tractability and strong tractability of linear multivariate problems, to appear in *J. Complexity*, March, 1994.

[33] D. Ylvisaker, Designs on random fields, in *A Survey of Statistical Design and Linear Models* (J. Srivastava, ed.), p. 593-607, *North-Holland*, Amsterdam, 1975.

Author's Addresses:

Grzegorz W. Wasilkowski, Department of Computer Science, University of Kentucky, Lexington, KY 40506, USA, email: greg@ms.uky.edu

Henryk Woźniakowski, Department of Computer Science, Columbia University, New York, NY 10027, USA, and Institute of Applied Mathematics, University of Warsaw, ul. Banacha 2, 02-097 Warszawa, Poland, email: henryk@cs.columbia.edu