Math. Log. Quart. **46** (2001) 3, 1 – 11

Mathematical Logic
Quarterly

On the computational complexity of best L_1-approximation

Paulo Oliva

BRICS, Department of Computer Science, University of Aarhus, DK-8000 Aarhus C, Denmark[1)]

Abstract. It is well known that for a given continuous function $f : [0,1] \to \mathbb{R}$ and a number n there exists a unique polynomial $p_n \in P_n$ (polynomials of degree $\leq n$) which best L_1-approximates f. We establish the first upper bound on the complexity of the sequence $(p_n)_{n\in\mathbb{N}}$, assuming f is polynomial-time computable. Our complexity analysis makes essential use of the modulus of uniqueness for L_1-approximation presented in [13].

Mathematics Subject Classification: 41A52, 41A10, 68Q17, 03F10.

Keywords: L_1-approximation, Strong Unicity, Proof Mining, Computable Analysis.

1 Introduction

It is well known in approximation theory (cf. Jackson's theorem, [1] or [3]) that for a fixed continuous function f on the interval $[0,1]$ (written $f \in C[0,1]$) and a fixed $n \in \mathbb{N}$ there exists a unique element of P_n (polynomials of degree $\leq n$ with real coefficients) which best approximates f with respect to the L_1-norm

$$\|g\|_1 :\equiv \int_0^1 |g(x)|\, dx.$$

More precisely, given $f \in C[0,1]$ and $n \in \mathbb{N}$ there exists a unique polynomial $p_n \in P_n$ such that

$$\|f - p_n\|_1 \leq \|f - p\|_1,$$

for any $p \in P_n$. In this paper we analyze the computational complexity of the sequence $(p_n)_{n\in\mathbb{N}}$, assuming f is a polynomial-time computable function. Since the coefficients of each p_n are potentially real numbers, in our analysis we make use of the concepts and tools developed in *computable analysis* (a brief introduction to computable analysis is presented in Section 2).

BRICS – Basic Research in Computer Science, funded by the Danish National Research Foundation.

1) e-mail:pbo@brics.dk

Our development in this paper follows the pattern used by Ko [4] in the analysis of the sequence $(p_n)_{n\in\mathbb{N}}$ of best Chebysheff approximations. The main difference in our approach is that we make a bold distinction between two steps in the analysis:

i) Finding a modulus of uniqueness Φ (see Section 3).
ii) Using Φ to compute (analyze the complexity of) the sequence $(p_n)_{n\in\mathbb{N}}$.

This distinction is important for understanding the difficulty in computing (or analysing the complexity of) the sequence $(p_n)_{n\in\mathbb{N}}$ of best L_1-approximations: The first modulus of uniqueness for L_1-approximation was presented very recently (cf. [13]), although uniqueness for L_1-approximation was known for over eighty years [3].

In Section 3 we present the notion of *modulus of uniqueness* and the modulus of uniqueness for L_1-approximation (see [13]) which is used by the algorithm described in Section 4. The general idea of the algorithm is taken from [8]. The algorithm is given together with the proof of correctness and complexity analysis.

Remark 1.1. In the following we make use of well-known classical complexity classes **P**, **NP**, **PSPACE**, **FP**, **#P** and classes in the polynomial hierarchy. Moreover, relativized complexity classes are represented by $\mathbf{C}[A]$, where $\mathbf{C}$ is a complexity class and A is an oracle, e.g. $\mathbf{P}[\mathbf{NP}]$ ($= \Delta_2^P$) means polynomial time with **NP** oracle. Readers not familiar with classical complexity theory are referred to e.g. [15].

2 Computable Analysis

While classical complexity theory deals with subsets of (or functions on) countable sets (e.g. $\mathbb{N}$, Σ^* for a finite alphabet Σ, etc.) *computable* (or *effective*) *analysis* deals mainly with operations on uncountable sets (e.g. $\mathbb{R}$, $C[0,1]$, Σ^ω, etc.). In this section we give a brief introduction to Ko's approach to effective analysis as presented in [5] and [6]. Therefore, all the definitions, Theorem 2.7 and Corollary 2.8 in this subsection are taken from [5] with small changes on the notation. For other essentially equivalent approaches to computable analysis see, for instance, [16] and [17].

2.1 Computable real number

Real numbers are represented by converging sequence of dyadic approximations. (A rational number is dyadic if it has a finite binary representation. The set of dyadic numbers is represented as $\mathbb{D}$.) If $d \in \mathbb{D}$ has binary representation $b_m \dots b_1.e_1 \dots e_n$ ($b_i, e_j \in \{0,1\}$) then d is said to have precision n (written $prec(d) = n$). A function $\psi : \mathbb{N} \to \mathbb{D}$ is a *Cauchy name* for a real number x if $|x - \psi(n)| \leq 2^{-n}$, for all $n \in \mathbb{N}$. A real number x is *computable* if it has a computable Cauchy name, i.e. if there exists a Turing machine M_x generating on input $n \in \mathbb{N}$ a $d_n \in \mathbb{D}$ such that $d_0, d_1, \dots$ is a Cauchy sequence converging to x with fixed rate 2^{-n}.

For our complexity analysis we must carefully fix how inputs are given. Natural numbers will be represented by elements of the set $S_1 = \{0\}^*$, and the dyadic numbers by elements of $S_2 \subset \{\cdot, 0, 1\}^*$ in the standard way. For the sake of simplicity we shall confuse the elements of S_1 and S_2 with the numbers they represent.

If there is a Turing machine M_x which on input $n \in S_1$ outputs a string $d_n \in S_2$ such that $\psi(n) :\equiv d_n$ is a Cauchy name for x and moreover the machine M_x works in polynomial time[2)] we say that x is a *polynomial-time computable real number* (written $x \in \mathbf{P}_{\mathbb{R}}$). The class $\mathbf{P}_{\mathbb{R}}$ can be characterized via general left cuts as follows.

Definition 2.1. Let ψ be a Cauchy name of $x \in \mathbb{R}$. The set

$$L = \{d \in S_2 : d \leq \psi(prec(d))\}$$

is called the *left cut of x associated with ψ* (or *a general left cut of x*).

Lemma 2.2. *Let $x \in \mathbb{R}$. $x \in \mathbf{P}_{\mathbb{R}}$ iff x has a general left cut in $\mathbf{P}$.*

Proof. If $x \in \mathbb{R}$ has a polynomial-time computable Cauchy name ψ, it is clear that the general left cut associated with this ψ will be in $\mathbf{P}$. On the other hand, suppose L is a general left cut of x in $\mathbf{P}$. Given a precision $k \in S_1$, by binary search on L, we can find a d such that $|x - d| \leq 2^{-k}$. Since $L \in \mathbf{P}$, the binary search can be performed in polynomial time. □

In this way we have reduced the problem of the complexity of a real number x to the complexity (in the sense of classical complexity theory) of a general left cut of x. The same idea can be used to define the class of *nondeterministic polynomial-time computable real number* $\mathbf{NP}_{\mathbb{R}}$, i.e. a real number x belongs to $\mathbf{NP}_{\mathbb{R}}$ if x has a general left cut in $\mathbf{NP}$.

Remark 2.3. In Section 4 we make use of a general left cut L of a real number x in order to compute an approximation $d \in \mathbb{D}$ of x with precision k (i.e. $|x-d| \leq 2^{-k}$). As mentioned above, this can be done in polynomial time with oracle access to L.

We shall now define computability and complexity for sequences of polynomials. Here we use Ko's notion of strong computability which is defined as follows. For simplicity we assume that the n-th polynomial has degree n.

Definition 2.4. A sequence of polynomials $(p_n)_{n\in\mathbb{N}}$ is *strongly computable* if there exists a Turing machine M which, for given $n, k \in S_1$, generates an $(n+1)$-tuple $b_0, \ldots, b_n \in S_2$ such that $|a_i - b_i| \leq 2^{-k}$, for $0 \leq i \leq n$, where $p_n(x) = a_0 + \ldots + a_n x^n$.

If the Turing machine M above works in polynomial time we say that the sequence $(p_n)_{n\in\mathbb{N}}$ is *strongly polynomial-time computable*. Strong $\mathbf{NP}$ computability is defined as follows.

Definition 2.5. A sequence of polynomials $(p_n)_{n\in\mathbb{N}}$ is *strongly* $\mathbf{NP}$ *computable* if there exists a polynomial-time non-deterministic Turing machine M such that, for given $n, k \in S_1$ at least one computation path is accepting, and in each accepting path an $(n+1)$-tuple $b_0, \ldots, b_n \in S_2$ is output such that $|a_i - b_i| \leq 2^{-k}$, for $0 \leq i \leq n$, where $p_n(x) = a_0 + \ldots + a_n x^n$.

This definition can be generalized, for instance, as follows:

i) if M is a polynomial-time deterministic oracle Turing machine with an $\mathbf{NP}$ oracle then $(p_n)_{n\in\mathbb{N}}$ is said to be strongly Δ_2^P computable;

ii) if M is a polynomial-time non-deterministic oracle Turing machine with an $\mathbf{NP}$ oracle then $(p_n)_{n\in\mathbb{N}}$ is said to be strongly Σ_2^P computable, etc.

[2)] The running time of a Turing machine is calculated as usual with respect to the size of the input string. It is fair to give the input n in unary since the required output $\psi(n) = d_n$ must be close to x by 2^{-n}, i.e. the string d_n ($\in S_2$) will normally have precision (and consequently length) greater n.

2.2 Computable real valued functions

We now investigate computability of functions $f : \mathbb{R} \to \mathbb{R}$. In this case we are interested in estimating the time required to compute $f(x)$ for any given $x \in \mathbb{R}$ (even non-computable ones). Since we are only interested in the complexity of f, we abstract from the complexity of the input x. That is done by assuming that x is given via an *oracle machine* O_x which on input m returns in constant time a $d_m \in \mathbb{D}$ such that $|x - d_m| \leq 2^{-m}$.

Definition 2.6. A function $f : \mathbb{R} \to \mathbb{R}$ is said to be *computable* if there exists an oracle Turing machines M_f which on input n (and oracle O_x) outputs $d_n \in \mathbb{D}$ such that $|f(x) - d_n| \leq 2^{-n}$.

From the definition above, it follows that any computable function is continuous. Moreover, it can also be proved that any computable f, on a fixed compact interval $[a, b]$, has a computable modulus of uniform continuity, i.e. there exists a computable (in the sense of classical recursion theory) function $\omega_f : \mathbb{N} \to \mathbb{N}$ such that

$$\forall k \in \mathbb{N}; x, y \in [a, b](|x - y| \leq 2^{-\omega_f(k)} \to |f(x) - f(y)| \leq 2^{-k}).$$

Theorem 2.7 ([5]). *If $f : [a, b] \to \mathbb{R}$ is computable on $[a, b]$ then*

- *f is continuous on $[a, b]$ and*
- *f has a computable modulus of uniform continuity on $[a, b]$.*

As a corollary of Theorem 2.7 we get a complete characterization of the computable functions in terms of computability of two number theoretic functions.

Corollary 2.8 ([5]). *A function $f : [a, b] \to \mathbb{R}$ is computable iff there exist two computable functions $f_r : \mathbb{D} \cap [a, b] \times \mathbb{N} \to \mathbb{D}$ and $\omega_f : \mathbb{N} \to \mathbb{N}$ such that*

- *$\forall d \in \mathbb{D} \cap [a, b]; n \in \mathbb{N}\ (|f(d) - f_r(d, n)| \leq 2^{-n})$,*
- *ω_f is a modulus of uniform continuity for f on $[a, b]$.*

The restriction to a compact domain here is essential since a continuous function on $\mathbb{R}$ need not to be uniformly continuous on $\mathbb{R}$. Once we have this characterization of computable functions f on compact intervals via a pair of computable number theoretic functions (f_r, ω_f) we can easily define the complexity of real functions on $[a, b]$. A function $f : [a, b] \to \mathbb{R}$ is *polynomial-time computable* if $f_r \in \mathbf{FP}$ and ω_f is a polynomial.

2.3 Complexity of integration

Note that integration is an operation which takes an f (e.g. in $C[0, 1]$) and returns an $x \in \mathbb{R}$. There is no well established notion of complexity classes for such operations. The best we can do is to analyze the complexity of the real number x when the complexity of f is fixed. A result which shows that integration is a difficult operation (in the sense just explained) is due to Friedman [2] and establishes that the integral of a polynomial-time computable function is always a polynomial-time computable real number iff $\mathbf{FP} = \#\mathbf{P}$. In our analysis we abstract from the complexity of integration by the use of an oracle. If we want to take into account the complexity of integration (oracle B_f of Section 4) the best result is given in [7]:

Theorem 2.9. *If $f \in C[0, 1]$ is polynomial-time computable then the real number $\int_0^1 f(x)\, dx$ is in $\mathbf{PSPACE}_{\mathbb{R}}$.*

3 The modulus of uniqueness

Let U and V be Polish spaces (i.e. complete, separable metric spaces) and $G : U \times V \to \mathbb{R}$ a real-valued continuous function. The fact that $G(u, \cdot)$ has at most one root in some compact set $V_u \subseteq V$ (parametrized by u) is expressed as

$$\forall u \in U; v_1, v_2 \in V_u \big(\textstyle\bigwedge_{i=1}^{2} G(u, v_i) = 0 \to v_1 = v_2 \big).$$

A *modulus of uniqueness* (notion introduced in [8]) for the function G is a functional Φ such that

$$\forall u \in U; v_1, v_2 \in V_u; k \in \mathbb{N} \ \big(\textstyle\bigwedge_{i=1}^{2} |G(u, v_i)| \leq 2^{-\Phi(u,k)} \to d_V(v_1, v_2) \leq 2^{-k} \big)$$

where d_V is a metric in V. The functional Φ generally depends on the representation of u as an element of the Polish space U.

Remark 3.1. It turns out that for a broad class of (even non-constructive) proofs of uniqueness theorems one can extract moduli of uniqueness (cf. [9]). Two such extractions have been carried out in the context of approximation theory (namely for the Chebyshev approximation [8, 9] and L_1-approximation [13]) as part of the project of proof mining (extraction of constructive content from prima facie ineffective proofs in mathematical analysis by means of logical analysis). For the case under study (L_1-approximation) the modulus of uniqueness was extracted from Cheney's proof of Jackson's theorem (cf. [1, 3]). Further information about the project of proof mining and other applications can be found in [10, 11, 12].

The main application of the modulus of uniqueness $\Phi(u, k)$ of a function G is its use in the computation of a root of $G(u, \cdot)$ uniformly in u, given that a root exists (see [8]). In the rest of the paper we carry out all the details of this computation for the case of L_1-approximation using the modulus of uniqueness presented in [13]. First, however, we explain how the general picture described above indeed applies to L_1-approximation. We should keep in mind though that the whole procedure is very general, and by no means confined to the area of approximation theory.

3.1 Best L_1-approximation

Let $(C[0, 1], \| \cdot \|_1)$ denote the normed linear space of all continuous functions on the interval $[0, 1]$ with metric $d(f, g) = \|f - g\|_1$. The *distance* of an element $f \in C[0, 1]$ from the subspace P_n (polynomials of degree $\leq n$) with respect to the L_1-norm is defined as $dist_1(f, P_n) :\equiv \inf_{p \in P_n} \|f - p\|_1$. Therefore, an element p^* is a *best L_1-approximation of f from P_n* if $\|f - p^*\|_1 = dist_1(f, P_n)$. If we define a function $G : (C[0, 1], \| \cdot \|_1) \times P_n \to \mathbb{R}$ as $G(f, p) :\equiv \|f - p\|_1 - dist_1(f, P_n)$ it is clear that $G(f, p^*) = 0$, i.e. the best L_1-approximations of f from P_n are precisely the roots of the function $G(f, \cdot)$.

We have not argued so far that any $f \in C[0, 1]$ indeed has a best L_1-approximation. This can be done in the following way. Let $K_{f,n} :\equiv \{p \in P_n : \|p\|_1 \leq 2\|f\|_1\}$. Any best L_1-approximation of f from $K_{f,n}$ is also a best L_1-approximation of f from P_n (cf. Lemma 3.3). Since $K_{f,n}$ is a bounded and closed subset of the finite-dimensional subspace P_n of the normed linear space $(C[0, 1], \| \cdot \|_1)$, it is compact. The existence of a best L_1-approximation then follows from the fact that G is continuous and therefore attains its infimum in the compact set $K_{f,n}$. As shown in [3], the best L_1-approximation of any $f \in C[0, 1]$ from P_n is in fact unique (henceforth called p_n).

A *modulus of uniqueness for L_1-approximation* is a functional Φ such that, for all f in $C[0,1]$,

$$\forall n \in \mathbb{N}; p_1, p_2 \in K_{f,n}; k \in \mathbb{N}\big(\bigwedge_{i=1}^{2}(|G(f,p_i)| \leq 2^{-\Phi(f,n,k)}) \to \|p_1 - p_2\|_1 \leq 2^{-k}\big).$$

We note that a modulus for the space $K_{f,n}$ can be easily extended to a modulus for the whole space P_n.

As pointed out in Remark 3.1, one can try to extract from a proof of the uniqueness of best L_1-approximation such a functional Φ. Such extraction is carried for Cheney's proof (cf. [1]) in [13] where a modulus of uniqueness Φ for L_1-approximation is obtained. The logical meta-theorems which guarantee the extraction of moduli of uniqueness, however, can only be applied when the function G (whose uniqueness of the root has been proved) is explicitly definable by a term of the underlying formal system and in particular continuous as a function on a Polish space. Since the space $(C[0,1], \|\cdot\|_1)$ is not complete, for the extraction of Φ we use the Polish space $U = (C[0,1], \|\cdot\|_\infty)$ instead. The functional G being continuous w.r.t. the uniform topology in $C[0,1]$ follows from the fact that $\|\cdot\|_1$ is continuous in $(C[0,1], \|\cdot\|_\infty)$ (which follows from $\|f\|_1 \leq \|f\|_\infty$).[3)]

As already mentioned, it is important that the functional Φ will in general depend on f through its representation as an element of $(C[0,1], \|\cdot\|_\infty)$. Such f is represented as a pair (f_r, ω_f), where the first element is the restriction of f to the dyadic numbers and ω_f is the modulus of uniform continuity of f (cf. Corollary 2.8).

3.2 Modulus of uniqueness for L_1-approximation

As mentioned above, the computation of the sequence[4)] $(p_n)_{n\in\mathbb{N}}$ for a given function $f \in C[0,1]$ makes essential use of the modulus of uniqueness for best L_1-approximation. We present the modulus (taken from [13], cf. Remark 3.1) in this section.

Theorem 3.2 ([13]). *Let*

$$\Phi(\omega_f, n, k) :\equiv 2k + (4n^2+10n+18)\log(n+2) + \omega_f(k + (2n^2+5n+6)\log(n+2)).$$

The functional Φ is a modulus of uniqueness for the best L_1-approximation of any $f \in C[0,1]$, having modulus of uniform continuity ω_f, from P_n, i.e. for all $n \in \mathbb{N}$, $p_1, p_2 \in P_n$,

$$\forall k \in \mathbb{N}\big(\bigwedge_{i=1}^{2} \|f - p_i\|_1 - dist_1(\omega_f, P_n) \leq 2^{-\Phi(\omega_f,n,k)} \to \|p_1 - p_2\|_1 \leq 2^{-k}\big).$$

Throughout the rest of the article, Φ will denote the modulus of uniqueness defined in Theorem 3.2.

It is important to notice that (besides being independent of p_1 and p_2) the modulus of uniqueness Φ depends on f only through its modulus of uniform continuity ω_f and does not depend on any particular value of the function f. Moreover, the above

[3)] The continuity of G (w.r.t. the uniform topology in $C[0,1]$) also follows from the fact that G is primitive recursively definable in (f_r, ω_f) and n (cf. [8]). Actually, this is the fact which guarantees the applicability of the meta-theorems of [8] (cf. Remark 3.1) to Cheney's proof of Jackson's theorem yielding the results of [13].

[4)] Throughout the rest of the paper p_n will denote the best L_1-approximation of $f \in C[0,1]$ from P_n, for a fixed f.

modulus has optimal k-dependency (as follows from [14]). In the following sections we will make use of some facts about the L_1-norm which we present here. For the rest of this section we let f and n be fixed.

L e m m a 3.3. *Let* $K_{f,n} :\equiv \{p \in P_n : \|p\|_1 \leq 2\|f\|_1\}$. *The zero polynomial (which belongs to* $K_{f,n}$*)* L_1*-approximates* f *better than any* $p \notin K_{f,n}$.

P r o o f. Let $p \notin K_{f,n}$ be fixed, i.e. $\|p\|_1 > 2\|f\|_1$. Therefore, by the triangle inequality for the L_1-norm, $\|f - p\|_1 > \|f\|_1$. □

As a consequence of Lemma 3.3 we get that $dist_1(f, P_n) = dist_1(f, K_{f,n})$. Therefore, any polynomial p^* such that $\|f - p^*\|_1 = dist_1(f, K_{f,n})$ is a best L_1-approximation of f from P_n.

Markov's inequality states that, for any given $p \in P_n$, $\|p'\|_\infty \leq 2n^2\|p\|_\infty$, where p' denotes the first derivative of p.

L e m m a 3.4. *If* $p \in P_n$ *then* $\|p\|_\infty \leq 2(n+1)^2\|p\|_1$.

P r o o f. Let $p(x) = a_0 + a_1x + \ldots + a_nx^n$. Define $\tilde{p}(x) :\equiv a_0x + \frac{a_1}{2}x^2 + \ldots + \frac{a_n}{n+1}x^{n+1}$. It is clear that for any $x \in [0, 1]$,

$$|\tilde{p}(x)| = |\textstyle\int_0^x p(y)dy| \leq \int_0^x |p(y)|dy \leq \|p\|_1,$$

therefore, $\|\tilde{p}\|_\infty \leq \|p\|_1$. Since the derivative of $\tilde{p}$ equals p, by Markov's inequality, we have $\|p\|_\infty \leq 2(n+1)^2\|p\|_1$. □

L e m m a 3.5. *Let* $p(x) = a_0 + a_1x + \ldots + a_nx^n$ *be an element of* P_n *and* $M \in \mathbb{R}^*_+$. *If* $\|p\|_1 \leq M$ *then* $|a_i| \leq 4(n+1)^{2(i+1)}M$, $0 \leq i \leq n$.

P r o o f. Let $p(x) = a_0 + a_1x + \ldots + a_nx^n$ and assume $\|p\|_1 \leq M$. By Lemma 3.4 we have, (1) $\|p\|_\infty \leq 2(n+1)^2M$. Let $p^{(i)}$ denote the i-th derivative of p. It is clear that $a_i = \frac{p^{(i)}(0)}{i!}$. By applying Markov's inequality i times and by (1) we have (2) $\|p^{(i)}\|_\infty \leq 2^{i+1}(n+1)^{2(i+1)}M$, and therefore,

$$|a_i| = \frac{|p^{(i)}(0)|}{i!} \overset{(2)}{\leq} \frac{2^{i+1}(n+1)^{2(i+1)}M}{i!} \leq 4(n+1)^{2(i+1)}M.$$

□

Let $p(x) = a_0 + a_1x + \ldots + a_nx^n$ be any polynomial in $K_{f,n}$ (which, by Lemma 3.3, includes p_n). By the definition of $K_{f,n}$ and Lemma 3.5 we have that $|a_i| \leq 8(n+1)^{2(i+1)}\ M$, for $0 \leq i \leq n$, where $M \in \mathbb{D}$ is an upper bound on $\|f\|_1$. Since we will use this bound on the coefficients of the elements of $K_{f,n}$ we give it a name, $C_{n,i} :\equiv 8(n+1)^{2(i+1)}\ M$. We will also need a function $\Theta(n, k)$ such that for polynomials $p(x) = a_0 + a_1x + \ldots + a_nx^n$,

$$\text{(1)} \qquad \|p\|_1 \leq 2^{-\Theta(n,k)} \rightarrow \bigwedge_{i=0}^{n} |a_i| \leq 2^{-k},$$

which can be easily derived from Lemma 3.5, for instance $\Theta(n, k) :\equiv 2(n+1)\log(n+1) + k + 2$.

D e f i n i t i o n 3.6. A set of elements $N_{n,k} \subset P_n$ is called a (n, k)*-net (for* $K_{f,n}$*)* if for any element $\tilde{p} \in K_{f,n}$ there exists an element $p \in N_{n,k}$ which is $(k+1)$-close to $\tilde{p}$, i.e. $\|p - \tilde{p}\|_1 \leq 2^{-k-1}$.

We want to choose a net based on the representation of the dyadic numbers so that we have control over the precision of the elements.

Lemma 3.7. *Let* $C_{n,k,i} :\equiv \{a \in S_2 : prec(a) \leq k + \log(\frac{n+1}{i+1})$ *and* $|a| \leq C_{n,i}\}$. *The space of polynomials* $N_{n,k} :\equiv \{a_0 + \ldots + a_n x^n : a_i \in C_{n,k,i}, 0 \leq i \leq n\}$ *is a* (n,k)*-net.*

Proof. Take an arbitrary element of $K_{f,n}$, say $\tilde{p}(x) = b_0 + \ldots + b_n x^n$. In the way we have chosen the coefficients of the elements of $N_{n,k}$ we are able to find $p(x) = a_0 + \ldots + a_n x^n \in N_{n,k}$ such that $|a_i - b_i| \leq \frac{2^{-k-1}(i+1)}{n+1}$, $0 \leq i \leq n$, i.e.

$$\begin{aligned} \|p - \tilde{p}\|_1 &= \|(a_0 - b_0) + \ldots + (a_n - b_n)x^n\|_1 \\ &= \int_0^1 |(a_0 - b_0) + \ldots + (a_n - b_n)x^n| dx \\ &\leq |a_0 - b_0| + \ldots + \frac{|a_{n-1} - b_{n-1}|}{n} + \frac{|a_n - b_n|}{n+1} \\ &\leq \frac{2^{-k-1}}{n+1} + \ldots + \frac{2^{-k-1}n}{n(n+1)} + \frac{2^{-k-1}(n+1)}{(n+1)(n+1)} = 2^{-k-1}. \end{aligned}$$

□

4 The complexity of $(p_n)_{n\in\mathbb{N}}$

For the rest of the article f denotes a fixed polynomial-time computable function. As mentioned before, we will analyze the complexity of the sequence $(p_n)_{n\in\mathbb{N}}$ relative to the complexity of integration. Therefore, in the following we will make use of an oracle B_f which is supposed to answer queries about integration.

In the case of the best Chebyshev approximation the value $dist_\infty(f, P_n)$ can be computed beforehand, and that value can be used in the computation of the best Chebyshev approximation of f from P_n. For the sake of comparison between the two cases of Chebyshev and L_1-approximation, in the first part of this section we first analyze the complexity of $(p_n)_{n\in\mathbb{N}}$ relative to an oracle A_f for $dist_1(f, P_n)$ (as done in [4] for the Chebyshev case). Then, in the last section we present an algorithm which does not need the values of $dist_1(f, P_n)$ in advance. From this algorithm we obtain a complexity upper bound for the sequence $(p_n)_{n\in\mathbb{N}}$ relative solely to the oracle B_f.

4.1 Using oracle A_f for $dist_1(f, P_n)$

Let L_n be a general left cut of the real number $dist_1(f, P_n)$. The oracle A_f decides the set

$$\{\langle n, d\rangle : d \in L_n\}$$

where $n \in S_1$ and $d \in S_2$. The second oracle B_f answers queries about general left cuts of the real numbers $\|f - p\|_1$, uniformly in p. More precisely, let $L_{n,p}$ denote a general left cut of the real number $\|f - p\|_1$. The oracle B_f decides the set [5)]

$$\{\langle n, p, e\rangle : e \in L_{n,p}\}$$

where $n \in S_1$ and $a_0, \ldots, a_n, e \in S_2$. As done in [4] for the Chebyshev case, we first show how to decide a certain set $\mathcal{G}_f$ using the oracles A_f and B_f. (The oracles are used as mentioned in Remark 2.3.)

5) If $p(x) = a_0 + \ldots + a_n x^n \in P_n$ we also write, for convenience, $\langle \ldots, p, \ldots\rangle$ instead of $\langle \ldots, a_0, \ldots, a_n, \ldots\rangle$.

$\langle n, k, a_0, \dots, a_n\rangle \in \mathcal{G}_f; n, k \in S_1$ and $a_0, \dots, a_n \in S_2$ Oracles: A_f, B_f
Let $s :\equiv \Phi(\omega_f, n, \Theta(n,k))$; If $p \notin N_{n,s+1}$ output *no*; (cf. Lemma 3.7) Compute $dist_1(f, P_n)$ with precision $s+3$ (let the resulting value be $d \in S_2$); Compute $\|f-p\|_1$ with precision $s+3$ (let the resulting value be $e \in S_2$); Output *yes* iff $|d-e| \leq 2^{-s-1}$.

Theorem 4.1. *Let $f \in C[0,1]$ be polynomial-time computable and ω_f a polynomial modulus of uniform continuity of f. There exists a multi-valued function α_f which on input n and k ($\in S_1$) produces a non-empty set of $(n+1)$-tuples ($\in S_2^{n+1}$) (representing elements of P_n) such that for each $\langle a_0, \dots, a_n\rangle \in \alpha_f(n,k)$,*

(*i*) *for $0 \leq i \leq n$, $prec(a_i) \leq \Phi(\omega_f, n, \Theta(n,k)) + \log(\frac{n+1}{i+1}) + 1$;*

(*ii*) *for $0 \leq i \leq n$, $|b_i - a_i| \leq 2^{-k}$ (where $p_n(x) = b_0 + \dots + b_n x^n$).*

Moreover,

(*iii*) Graph$(\alpha_f) \in \mathbf{P}[A_f, B_f]$.

Proof. Let s be a shorthand for $\Phi(\omega_f, n, \Theta(n,k))$. We define α_f to be the function that maps each $n, k \in S_1$ to all $(n+1)$-tuples $\langle a_0, \dots, a_n\rangle \in S_2^{n+1}$ such that $\langle n, k, a_0, \dots, a_n\rangle \in \mathcal{G}_f$, i.e. α_f is the function whose graph is $\mathcal{G}_f$. We first have to argue that α_f is total. Let n, k be fixed. By Lemma 3.7 and the fact that $p_n \in K_{f,n}$, there exists a $p \in N_{n,s+1}$ such that $\|p_n - p\|_1 \leq 2^{-s-2}$. By the triangle inequality for the L_1-norm we get $\|f-p\|_1 - dist_1(f, P_n) \leq 2^{-s-2}$. By the computation of d and e we have

$$|\, d - dist_1(f, P_n)\,| \leq 2^{-s-3} \quad \text{and} \quad |\, e - \|f-p\|_1\,| \leq 2^{-s-3},$$

which implies $|d-e| \leq 2^{-s-1}$, and the input p is accepted.

(*i*) Immediate consequence of the definition of a net (3.6) and the definition of α_f.

(*ii*) Suppose $\langle a_0, \dots, a_n\rangle \in \alpha_f(n,k)$ (let $p(x) :\equiv \sum_{i=0}^n a_i x^i$). This implies $|e-d| \leq 2^{-s-1}$, d and e as above. We then obtain $\|f-p\|_1 - dist_1(f, P_n) \leq 2^{-s}$. By Theorem 3.2 we get $\|p_n - p\|_1 \leq 2^{-\Theta(n,k)}$. And by (1) of Section 3.2, $|b_i - a_i| \leq 2^{-k}$, for $0 \leq i \leq n$, where $p_n(x) = b_0 + \dots + b_n x^n$.

(*iii*) Since ω_f is a polynomial (cf. Section 2.2), $\Phi(\omega_f, n, \Theta(n,k))$ is also a polynomial and the procedure $\mathcal{G}_f$ above can be performed in polynomial time (in A_f and B_f). Notice also that since f is fixed the net $N_{n,s+1}$ has size exponential on the input. □

Corollary 4.2. *Let $f \in C[0,1]$ be polynomial-time computable. The sequence of best L_1-approximation $(p_n)_{n\in\mathbb{N}}$ is strongly $\mathbf{NP}[A_f, B_f]$ computable.*

Proof. Let $n, k \in S_1$ be given. We define a non-deterministic oracle Turing machine M as follows. The oracles of M will be the sets A_f and B_f. Each computation path of M takes into consideration one element $p \in N_{n,s+1}$ (s as above). The machine (in each path) decides whether $\langle n, k, p\rangle$ belongs to $\mathcal{G}_f$ (i.e. Graph(α_f)) or not. If yes then the path is accepted and the machine outputs p. Note that, by Theorem 4.1 (*i*), the size of p is a polynomial on n and k. □

We obtain, for instance, that if A_f and B_f are in **NP** then Graph$(\alpha_f) \in \Delta_2^P$ and $(p_n)_{n\in\mathbb{N}}$ is strongly Σ_2^P computable.

Remark 4.3. Note that the set

$$L :\equiv \{\langle n, d\rangle \in S_1 \times S_2 : \forall p \in N_{n,k}\ (d \le \|f - p\|_1^{k+1})\}$$

where the k above abbreviates $prec(d)$ and $\|f - p\|_1^{k+1}$ is a $(k+1)$-approximation of the value $\|f - p\|_1$, does the job of the oracle A_f. In other words, the set

$$L_n :\equiv \{d \in S_2 : \langle n, d\rangle \in L\}$$

is a general left cut of $dist_1(f, P_n)$. An algorithm for deciding the complement of L can be given as follows. On input $d \in S_2$ (with precision k) and $n \in S_1$, non-deterministically choose a polynomial from $N_{n,k}$ and compute the value of $\|f - p\|_1$ with precision $k + 1$ (say e). Then, answer *yes* (i.e. $\langle n, d\rangle \notin L$) when $d > e$. In this way, using the oracle B_f for integration, we obtain an upper bound $\mathbf{coNP}[B_f]$ on the complexity of the oracle A_f. Note also that the above procedure does not make use of the fact that the best L_1-approximation of f is unique.

4.2 Absolute complexity of $(p_n)_{n\in\mathbb{N}}$

In this section we present another algorithm which only uses the oracle B_f for a general left cuts of $\|f - p\|_1$ (and does not make use of the oracle A_f). We first use B_f to define the set $\tilde{\mathcal{G}}_f$,

$\langle n, k, p, \tilde{p}\rangle \in \tilde{\mathcal{G}}_f$; $n, k \in S_1$ and $p, \tilde{p} \in S_2^{n+1}$ Oracles: B_f
Let $s :\equiv \Phi(\omega_f, n, \Theta(n, k))$; If $p \notin N_{n,s+1}$ output *no*; (cf. Lemma 3.7) Compute $\|f - p\|_1$ with precision $s + 3$ (let the resulting value be $e \in S_2$); Compute $\|f - \tilde{p}\|_1$ with precision $s + 3$ (let the resulting value be $\tilde{e} \in S_2$); Output *yes* iff $e \le \tilde{e} + 2^{-s-1}$.

Note that deciding membership for the set $\tilde{\mathcal{G}}_f$ can be done in polynomial-time using the oracle B_f, i.e. $\tilde{\mathcal{G}}_f \in \mathbf{P}[B_f]$. Let

$$\mathcal{G}_f :\equiv \{\langle n, k, p\rangle : \forall \tilde{p} \in N_{n,\Phi(\omega_f,n,\Theta(n,k))+1}(\langle n, k, p, \tilde{p}\rangle \in \tilde{\mathcal{G}}_f)\}.$$

Theorem 4.4. *Let $f \in C[0, 1]$ be polynomial-time computable and ω_f a polynomial modulus of uniform continuity of f. There exists a multi-valued function β_f which on input n and k ($\in S_1$) produces a non-empty set of $(n + 1)$-tuples ($\in S_2^{n+1}$) (representing elements of P_n) such that for each $\langle a_0, \ldots, a_n\rangle \in \beta_f(n, k)$,*

(i) for $0 \le i \le n$, $prec(a_i) \le \Phi(\omega_f, n, \Theta(n, k)) + \log(\frac{n+1}{i+1}) + 1$;

(ii) for $0 \le i \le n$, $|b_i - a_i| \le 2^{-k}$ (where $p_n(x) = b_0 + \ldots + b_n x^n$).

Moreover,

(iii) Graph$(\beta_f) \in \mathbf{coNP}[B_f]$.

Proof. Let s be a shorthand for $\Phi(\omega_f, n, \Theta(n, k))$. We define β_f to be the function that maps each $n, k \in S_1$ to all $(n + 1)$-tuples $\langle a_0, \ldots, a_n\rangle \in S_2^{n+1}$ such that

$\langle n,k,a_0,\ldots,a_n\rangle \in \mathcal{G}_f$, i.e. β_f is the function whose graph is $\mathcal{G}_f$. First we have to prove that β_f is total. Let p be an element of $N_{n,s+1}$ such that $\|f-p\|_1 \leq \min_{\tilde{p}\in N_{n,s+1}} \|f-\tilde{p}\|_1$. Then, clearly, $\langle n,k,p,\tilde{p}\rangle \in \tilde{\mathcal{G}}$, for all $\tilde{p} \in N_{n,s+1}$. Therefore, $\langle n,k,p\rangle \in \mathrm{Graph}(\beta_f)$.

(i) Immediate consequence of the definition of a net (3.6) and the definition of β_f.

(ii) Assume $\langle n,k,p,\tilde{p}\rangle \in \tilde{\mathcal{G}}_f$, for all $\tilde{p} \in N_{n,s+1}$. That implies

$$(*) \quad \forall \tilde{p} \in N_{n,s+1}\big(\|f-p\|_1 \leq \|f-\tilde{p}\|_1 + 3\cdot 2^{-s-2}\big).$$

Since $p_n \in K_{f,n}$ (and by the definition of (n,k)-net) there is an element $\tilde{p} \in N_{n,s+1}$ such that $\|p_n-\tilde{p}\|_1 \leq 2^{-s-2}$ and by triangle inequality we get, $\|f-\tilde{p}\|_1 \leq dist(f,P_n)+2^{-s-2}$. By $(*)$ we get, $\|f-p\|_1 \leq dist_1(f,P_n)+2^{-s}$. Hence, by Theorem 3.2 we have $\|p_n-p\|_1 \leq 2^{-\Theta(n,k)}$. And by (1) of Section 3.2 $|b_i-a_i| \leq 2^{-k}$, for $0 \leq i \leq n$, where $p_n(x) = b_0+\ldots+b_nx^n$.

(iii) Similar to Theorem 4.1 (iii). □

Corollary 4.5. *Let $f \in C[0,1]$ be polynomial-time computable, then the sequence $(p_n)_{n\in\mathbb{N}}$ is strongly* **NP** *computable in* $\mathbf{NP}[B_f]$.

Proof. Let $n,k \in S_1$ be given. We define a non-deterministic oracle Turing machine M as follows. The oracle of M will be the set $\mathrm{Graph}(\beta_f)$ (which is in $\mathbf{coNP}[B_f]$). Each computation path of M takes into consideration one element $p \in N_{n,s+1}$ (s as above). The machine (in each path) decides whether $\langle n,k,p\rangle$ belongs to $\mathrm{Graph}(\beta_f)$ or not. If yes then the path is accepted and the machine outputs p. We also note that, as our oracle we can as well use the complement of the set $\mathrm{Graph}(\beta_f)$. □

5 Conclusion

We have established the first complexity upper bound on the sequence $(p_n)_{n\in\mathbb{N}}$ of best L_1-approximations of a polynomial time computable $f \in C[0,1]$. For the complexity analysis we made use of two oracles A_f and B_f solving generalized left cuts of $dist_1(f,P_n)$ and $\|f-p\|_1$ respectively in two different ways:

1) *Relative to both oracles A_f and B_f.* We have shown that the sequence $(p_n)_{n\in\mathbb{N}}$ is strongly **NP** computable relative to those oracles. Since the oracle A_f has a trivial $\mathbf{coNP}[B_f]$ upper bound (cf. Remark 4.3) we obtain that $(p_n)_{n\in\mathbb{N}}$ is strongly $\mathbf{NP}[\mathbf{NP}[B_f],B_f]$ computable, i.e. strongly **NP** computable relative to an $\mathbf{NP}[B_f]$ oracle.
2) *Relative to oracle B_f.* We have also analyzed the complexity of $(p_n)_{n\in\mathbb{N}}$ without first computing the value $dist_1(f,P_n)$. In this case we concluded directly that the sequence $(p_n)_{n\in\mathbb{N}}$ is strongly **NP** computable relative to an $\mathbf{NP}[B_f]$ oracle.

One should note that our complexity analysis strongly relies on the modulus of uniqueness for L_1-approximation, first presented in [13].

In [4] a relation is established between the sequence $(p_n)_{n\in\mathbb{N}}$ (of best Chebysheff approximations of a polynomial time computable $f \in C[0,1]$) and separation of well known complexity classes. It is not known whether similar results also hold in the case under study of L_1-approximation.

Acknowledgement. I would like to thank Ulrich Kohlenbach for interesting discussions and many helpful suggestions.

References

[1] E.W. Cheney. *Approximation Theory.* AMS Chelsea Publishing, 1966.

[2] H. Friedman. On the computational complexity of maximization and integration. *Advances in Mathematics*, 53:80–98, 1984.

[3] D. Jackson. Note on a class of polynomials of approximation. *Transactions of the American Mathematical Society*, 22:320–326, 1921.

[4] K.-I. Ko. On the computational complexity of best Chebycheff approximation. *Journal of Complexity*, 2:95–120, 1986.

[5] K.-I. Ko. *Complexity theory of real functions.* Birkhäuser, Boston-Basel-Berlin, 1991.

[6] K.-I. Ko. Polynomial-time computability in Analysis. In Yu L. Ershov, S.S. Goncharov, A. Nerode, and J.B. Remmel, editors, *Handbook of recursive mathematics (vol. 2)*, volume 139 of *Studies in Logic and the Foundations of Mathematics*, pages 1271–1317. North-Holland, Amsterdam, 1998.

[7] K.-I. Ko and H. Friedman. Computational complexity of real functions. *Theoretical Computer Science*, 20:323–352, 1982.

[8] U. Kohlenbach. Effective moduli from ineffective uniqueness proofs. An unwinding of de La Vallée Poussin's proof for Chebycheff approximation. *Annals of Pure and Applied Logic*, 64:27–94, 1993.

[9] U. Kohlenbach. New effective moduli of uniqueness and uniform a–priori estimates for constants of strong unicity by logical analysis of known proofs in best approximation theory. *Numerical Functional Analysis and Optimization*, 14:581–606, 1993.

[10] U. Kohlenbach. Analysing proofs in Analysis. In W. Hodges, M. Hyland, C. Steinhorn, and J. Truss, editors, *Logic: from Foundations to Applications*, pages 225–260. European Logic Colloquium (Keele, 1993), Oxford University Press, 1996.

[11] U. Kohlenbach. On the computational content of the Krasnoselski and Ishikawa fixed point theorems. In J. Blanck, V. Brattka, and P. Hertling, editors, *Computability and Complexity in Analysis, (CCA'2000)*, volume 2064 of *Lecture Notes in Computer Science*, pages 119–145. Springer, 2001.

[12] U. Kohlenbach. A quantitative version of a theorem due to Borwein-Reich-Shafrir. *Numerical Functional Analysis and Optimization*, 22:641–656, 2001.

[13] U. Kohlenbach and P. Oliva. Proof mining in L_1-approximation. Submitted (35 pages), 2001. An extended technical report version can be found in: BRICS Report Series (http://www.brics.dk/RS/01/14/BRICS-RS-01-14.ps.gz), RS-01-14 (38 pages), 2001.

[14] A. Kroó. On the continuity of best approximations in the space of integrable functions. *Acta Mathematica Academiae Scientiarum Hungaricae*, 32:331–348, 1978.

[15] C. Papadimitriou. *Computational Complexity.* Addison-Wesley, 1994.

[16] M. Pour-El and J. Ian Richards. *Computability in Analysis and Physics.* Perspectives in Mathematical Logic. Springer-Verlag, 1989.

[17] K. Weihrauch. *Computable Analysis.* Springer-Verlag, 2000.