Approximating Convex Functions By Non-Convex Oracles Under The Relative Noise Model

Nir Halman*

February 2, 2014

Abstract

We study succinct approximation of functions that have noisy oracle access. Namely, construction of a succinct representation of a function, given oracle access to an L-approximation of the function, rather than to the function itself. Specifically, we consider the question of the succinct representation of an approximation of a convex function φ that cannot be accessed directly, but only via oracle calls to a general (i.e., not necessarily convex) L-approximation $\tilde{\varphi}$ of φ. We efficiently construct such a succinct $(1+\epsilon)L^2$-approximation for a univariate convex φ, for any $\epsilon > 0$. The algorithms designed in this paper can, and are used as subroutines (gadgets) within other approximation algorithms.

1 Introduction

1.1 Succinct representation of functions given noisy oracles

On succinct representation of data. A broadly successful approach to massive datasets analysis involves understanding and manipulating not the raw data, but the essence of the data. Not all the data is captured, but only a representation suitable for subsequent analysis. Ideally, this representation is succinct, i.e., far smaller than the original data, and adequate at least for approximate analysis.

When dealing with datasets, errors may occur unintentionally during the process of data acquisition (e.g., white noise) or data processing (e.g., roundoff errors). But errors may *intentionally* be allowed in order to speedup data processing (e.g., approximation of the requested value). While unintentional errors are typically of additive nature (stochastic/robust), we consider intentional errors of multiplicative nature. In this setting, it is perhaps more natural to view the dataset as a function φ over a finite domain.

In this paper we consider an ideal function φ that is assumed to satisfy various known structural properties, e.g., it may be monotone. Because of errors, the oracle $\tilde{\varphi}$ may not be such. However, for every point x in the domain, $\tilde{\varphi}(x)$ is limited to be at least $\varphi(x)$ and at most $L\varphi(x)$, where $L > 1$ is a given constant. In this way $L - 1$ is the *relative error* of $\tilde{\varphi}$. Considering monotonicity, we note that sorted lists of numbers are a requirement for all kind of operations. For example, a binary search will easily err if the list is not perfectly sorted. We would like therefore to process the function efficiently and store a succinct representation of it such that for any query point we will be able to return a value that is (i) consistent with a sorted list and (ii) differs from the ideal φ by a factor of at most KL, $(K > 1)$. An immediate application of such a representation is to provide robustness for binary search over functions that are *far* from being monotone (e.g., half of their values need to be changed in order to retain monotonicity), but the relative error in *each* value is bounded by $L - 1$. Another application, which we pursue in this paper, is constructing a bounded-error relative approximation for φ.

*Hebrew University of Jerusalem, halman@huji.ac.il

On noise models. Consider the following "traditional" settings for accessing a function φ (the first two have no error):

1. *Explicit*: One has explicit access to the function, e.g., one is given its closed-form formula.
2. *Implicit*: One accesses the function via queries to an oracle. For each query point x, one gets a value depending on the specific setting:
   (a) *Direct ("black box")*: $\varphi(x)$.
   (b) *Stochastic*: $\tilde{\varphi}(x) := \varphi(x) + \epsilon(x)$, where $\epsilon(x)$ is a random additive sampling error with a given probability distribution.
   (c) *Robust*: $\tilde{\varphi}(x) := \varphi(x) + \epsilon(x)$, where $\epsilon(x)$ is a random additive sampling error with unknown probability distribution.

Consider the problem of convex function minimization. In the explicit setting, if the function is differentiable then using the Karush-Kuhn-Tucker (KKT) conditions may lead to a closed-form solution that can be derived analytically. In the direct (or stochastic) setting, methods for numerically solving the KKT system of equations may be used. We note that among works that primarily consider the robust setting (where $\epsilon(x)$ is arbitrary large for an unknown small fraction of the points, and otherwise is zero) are those on property testing, self correction, and property reconstruction [BLR90, RS96, ACCL08].

The relative noise model. We propose to investigate the following natural noise model, which arises, e.g., when approximating functions recursively: for every query point x, we are provided with $\tilde{\varphi}(x)$, where $\tilde{\varphi}$ is an L-approximation function of φ, for a given constant $L \geq 1$. $\tilde{\varphi}$ is said to be an *L-approximation function* of φ if it returns a value that is between $\varphi(x)$ and $L\varphi(x)$, for every point x in its domain. We call our model the *relative* noise model.

Succinct approximation under the relative noise model. We formalize the problem of succinct approximations of functions. The goal is to efficiently construct a succinct M-approximation of a non-negative function $\varphi : D \to \mathbb{R}^+$ over a linearly ordered finite domain D, while having access only to an L-approximation $\tilde{\varphi} : D \to \mathbb{R}^+$ of it. By *succinct* we mean that the space used for the representation of the approximation must be polylogarithmic in $|D|$ and $\frac{\varphi^{\max}}{\varphi^{\min}}$, where $\varphi^{\max} = \max_{x \in D} \varphi(x)$ and $\varphi^{\min} = \min\{\varphi(x) \mid x \in D \text{ and } \varphi(x) > 0\}$. By *efficient* we mean that the time and number of oracle calls to $\tilde{\varphi}$ needed by an algorithm to create the approximation function must be polylogarithmic in these two terms as well. We would like to have $M > L$ be as small as possible.

If $L = 1$, i.e., in the direct setting where one has a "black box" access to the function itself, this can be done with $M = L$ quite easily for either monotone or convex univariate functions [HKL+08]. Note that if the function is unimodal - this is not possible, e.g., consider a function whose value is always 1 except for one point in the domain, in which its value is 0.

If $L > 1$, i.e., in the relative setting, approximating monotone functions is quite straightforward for $M = (1+\epsilon)L$ and any $\epsilon > 0$, mainly because the exact argmin of φ is known [HKL+13]. The convex case is more involved.

1.2 Our results and contributions

Our main algorithmic result is:

Theorem 1.1 (Succinct approximation of a convex function via an L-approximation general oracle)**.** *Let $\varphi : [A, B] \to \mathbb{Z}^+$ be a univariate convex function, $L > 1$ be a constant, $\tilde{\varphi}$ be an L-approximation function* *of φ and $t_{\tilde{\varphi}}$ be an upper bound on the time needed to evaluate $\tilde{\varphi}(x)$, for any $x \in D$. Then for every $K = 1 + \epsilon > 1$, it is possible to construct in*

$$O\left(\frac{1 + t_{\tilde{\varphi}}}{\epsilon} \log \tilde{\varphi}^{\max} \left[\log\log \tilde{\varphi}^{\max} + \log(1/\epsilon)\right] \left[L^4 + \log(B - A)\right]\right)$$

time a convex piecewise-linear KL^2-approximation function of φ that has $O(\frac{\log \tilde{\varphi}^{\max}}{\epsilon})$ breakpoints.

We note that it may be possible to use this result as a "building block" when approximating multidimensional convex (or submodular) functions, as is already done with corresponding results in the direct setting [CDJ14]. But this result does not extend in a natural way to multidimensional domains, even under the direct setting. I.e., it is impossible, in general, to get succinct K-approximations even for bivariate functions that are both monotone and discretely-convex. We formally state our main impossibility result:

Theorem 1.2 (Non-existence of succinct approximations for multivariate functions)**.** *For any $1 \leq K < 2$, a bivariate monotone discretely-convex function in the sense of Miller [Mil71] does not necessarily admit a succinct K-approximation, regardless of the scheme used to represent the function.*

Our contribution. Our contribution is fourfold. First, we propose a novel perspective on the problem of succinct approximations - we study succinct approximations under the relative setting where one can only access an L-approximation of a function and not the function itself. To the best of our knowledge, this problem was not studied before under this setting. Second, we efficiently construct succinct approximations for univariate convex functions in this setting by means of careful algorithm design. Third, besides this stand-alone algorithmic result, the algorithms designed here can, and are successfully used as subroutines within other approximation algorithms, such as approximation algorithms based on the method of K-approximation sets and functions (see below). Last - we give the first impossibility result for succinctly approximating multidimensional discretely-convex/monotone functions, even under the direct setting.

1.3 Related work

Property-preserving reconstruction. It is interesting to compare this work and the one on property-preserving reconstruction [ACCL08, SS10]. In monotonicity reconstruction, the function is given in the robust setting and is assumed to be monotone. The (additive) sampling error $\epsilon(x)$ equals zero for an unknown fraction $1 - \epsilon$ of the points in D, and can be arbitrary large otherwise. In other words, one assumes that φ is monotone, and that in general the oracle $\tilde{\varphi}$ must be modified at $\epsilon|D|$ places to become monotone. The goal is to construct in an online fashion a monotone filter f that for any query point $x \in D$ returns a value $f(x)$ that, although not necessarily equal to $\varphi(x)$, differs from it as infrequently as possible. Because f should resemble to φ as much as possible, and $\tilde{\varphi}$ is equal to φ on $(1 - \epsilon)|D|$ of the points, in an offline preprocessing, the filter can always go over the entire domain, compute the "nearest" monotone function, and store it as its filtered function. This is not efficient, however, since the number of queries performed is linear in $|D|$ (and not polylogarithmic). In our work, the function is given in the relative setting. $\tilde{\varphi}(x)$ may differ from a monotone $\varphi(x)$ on *all* of the points in the domain, but its value is *always* "close" to $\varphi(x)$, i.e., it satisfies $\varphi(x) \leq \tilde{\varphi}(x) \leq L\varphi(x)$, for some given constant $L > 1$. The goal is to construct a function ψ that is a monotone approximation of φ. It turns out that an efficient offline construction of a succinct approximation ψ of φ is possible. This is mainly because we allow ψ to differ from φ on "many" points and by using the monotonicity of φ. (E.g., consider of $\varphi(x) := x$, $\forall x \in [10, 20]$, $\tilde{\varphi}(x) := \lceil 1.5x \rceil + 3(-1)^x$ and $L = 1.8$. Then $\tilde{\varphi}$ is an L-approximation of φ, $\tilde{\varphi} \neq \varphi$ in all points of the domain and should be corrected in half of its domain in order to be monotone. Nevertheless, it is possible to construct a monotone succinct 2-approximation for φ such as $\psi(x) := 2x$, while accessing only $\tilde{\varphi}$). We note in passing that [ACCL08, SS10] design reconstructions of d-dimensional monotone functions, while we show that it is impossible in general to get efficient succinct approximations for d-dimensional monotone functions, even under the direct setting.

Discrete convexity. In discrete optimization, discrete analogs of convexity, or "discrete convexity" for short, have been considered. Miller investigated a class of discrete functions, of which local optimality implies global optimality [Mil71]. Favati and Tardella considered a certain special way of extending functions defined over the integer lattice to piecewise-linear functions defined over the real space, and they introduced the concept of "integrally convex functions" [FT90]. Murota introduced the concepts of "L-convexity" and "M-convexity," in which "L" stands for "Lattice" and "M" stands for "Matroid" [Mur03]. L- and M-convex functions possess several desirable properties as discrete convex functions, including extendability to ordinary (continuous) convex functions, duality theorems, and conjugacy between L- and M-convex functions, etc. Our impossibility result, Theorem 1.2, deals with discretely-convex functions in the sense of Miller, which is a specific, although quite general, class of discretely-convex functions.

K-approximation sets and functions. The method of K-approximation sets and functions, introduced in earlier work with coauthors, is used for designing fully polynomial time approximation schemes (FPTAS) for discrete-time finite-time stochastic dynamic programs (DP) where *direct ("black-box")* access to the single-period cost functions is assumed [HKM+09]. The basic idea underlying the FPTAS is to K-approximate the functions involved in a DP by keeping only a logarithmic number of points in their domain (called a K-approximation set). One then uses a step function or linear interpolation to determine the function value at points that have been eliminated from the domain. [HKL+08] give an FPTAS for three classes of problems that fall into this framework: convex DP, nondecreasing DP, nonincreasing DP. This FPTAS is not problem-specific, but relies solely on structural properties of the DP. This was used to give the first FPTAS for several problems, and to improve (running-time wise) upon existing "tailor-made" FPTASs for other problems (e.g., for deterministic single-item capacitated economic lot-sizing problem with a monotone cost, the "general" FPTAS in [HKL+08] runs faster than the currently best ad-hoc FPTAS [CNC10]).

1.4 Applications of our model

Generalized binary search. Suppose one is interested in minimizing a univariate convex function φ. If there is direct access to φ, then by applying binary search one can efficiently minimize φ. Our current work enables one to apply binary search in the relative setting, when access to φ is via a (not necessarily convex) L-approximation $\tilde{\varphi}$ of φ: all we need is to apply Theorem 1.1 in order to get a convex succinct approximation of φ and then perform binary search over it.

Generalized K-approximation sets and functions. The FPTAS framework discussed above assumes direct ("black-box") access to the single-period functions which we denote by g_t. This work generalizes this FPTAS framework to work under the relative setting: suppose that the single-period cost functions g_t of a convex DP are not known, but an FPTAS $\tilde{g}_t^\epsilon$ for them is available. If $\tilde{g}_t^\epsilon$ is convex then the convex structure of the DP is maintained and the framework of [HKL+08] applies. Otherwise, by Theorem 4.1 below, one gets *convex* FPTASs for the various g_t, and again the aforementioned framework applies. We note in passing that any cost function that requires simulation can be computed approximately with high probability when the lowest and greatest nonzero probability is bounded away from zero and one.

1.5 Notation.

Let $\mathbb{R}, \mathbb{R}^+, \mathbb{Z}, \mathbb{Z}^+, \mathbb{N}$ denote the set of real numbers, nonnegative real numbers, integers, nonnegative integers and positive integers, respectively. For every pair A, B of integers, $A < B$ let $[A, B] = \{A, A+1, \dots, B\}$ denote the set of integers between A and B. For simplicity, we assume throughout that $B - A > 1$. A function $\varphi : D \to \mathbb{R}$ over a linearly ordered set D is called *increasing* if it is nondecreasing on D. Similarly, φ is called *decreasing* if it is nonincreasing on D. We denote by t_φ the time needed to evaluate φ on a single point in its domain. Given a finite set $D \in \mathbb{R}$ and $x \in [D^{\min}, D^{\max}]$, for $x < D^{\max}$ let $\mathrm{next}(x, D) := \min\{y \in D \mid y > x\}$, and for $x > D^{\min}$ let $\mathrm{prev}(x, D) := \max\{y \in D \mid y < x\}$. Given a function defined over a finite set $\varphi : D \to \mathbb{R}$, we define $\sigma_\varphi(x) := \frac{\varphi(\mathrm{next}(x,D)) - \varphi(x)}{\mathrm{next}(x,D) - x}$ as the slope of φ at x for any $x \in D \setminus \{D^{\max}\}$, $\sigma_\varphi(D^{\max}) := \sigma_\varphi(\mathrm{prev}(D^{\max}, D))$. For any subset $D' \subseteq D$, we define the *piecewise linear extension of* φ *induced by* D' as the function obtained by making φ linear between successive values of D'. For any subset $D' \subseteq D$, a function φ over D is called *convex over* D' if its piecewise linear extension induced by D' is convex. For any subset $D' \subseteq D$, we define the *convex extension of* φ *induced by* D' as the function defined as the lower envelope of the convex hull of $\{(x, \varphi(x)) \mid x \in D'\}$. The base two logarithm of z is denoted by $\log z$. In our setting the problem input is given as A, B and an oracle $\tilde{\varphi}$. We define the input size to be $\log A + \log B + \log \tilde{\varphi}^{\max}$.

2 Non-approximability of multivariate convex functions

A function $f : [1, U]^d \to \mathbb{R}^+$ is said to be a "Miller's discretely convex function" (*discretely convex function* for short) if

$$\min\{f(z) \mid z \in N(\alpha x + (1-\alpha) y)\} \le \alpha f(x) + (1-\alpha) f(y) \tag{1}$$

holds for any $x, y \in [1, U]^d$ and any $0 \le \alpha \le 1$, where $N(t) = \{t' \in \mathbb{Z}^d \mid \|t - t'\|_\infty < 1\}$ for $t \in \mathbb{R}^d$, [Mil71]. Note that Miller's discretely convex functions is a class of convex functions which is fairly broad [Mur03, p. 37], and they have the characteristic that local optimality implies global optimality [Mil71].

We now state the following proposition, where its validity follows directly from the definition of K-approximation functions.

Proposition 2.1. *For every* $1 \le K < 2$, $d \in \mathbb{N}$, *and binary function* $\varphi : [1, U]^d \to \{0, 1\}$, *any integer-valued* K*-approximation function of* φ *coincides with* φ.

By Proposition 2.1, approximating φ instead of calculating it exactly does not reduce the complexity of the problem.

We next calculate a lower bound on the number of nondecreasing discretely convex functions $\varphi : [1, U]^2 \to [0, U]$. We say that $\varphi_1, \varphi_2 : [1, U]^2 \to [0, U]$ are *binary distinct* if $\varphi_1^{-1}(0) \ne \varphi_2^{-1}(0)$; that is, the domains on which their values are zero are different. (An equivalent definition is as follows: φ_1, φ_2 are binary distinct if $bin(\varphi_1) \ne bin(\varphi_2)$, where $bin(\varphi)$ is a function such that $bin(\varphi)(x, y) = 0$ if $\varphi(x, y) = 0$ and $bin(\varphi)(x, y) = 1$ otherwise.)

Proposition 2.2. *There are* $\Omega(2^{\sqrt{U}})$ *binary distinct nondecreasing discretely convex functions* $\varphi : [1, U]^2 \to [0, U]$.

Proof. Let

$$\Phi = \left\{ \varphi_{r_1,\dots,r_U} : [1, U]^2 \to [0, U] \;\middle|\; r_1, \dots, r_U \in [0, U];\ \sum_{i=1}^{U} r_i = U;\ r_1 \le r_2 \le \dots \le r_U \right\}$$

be a family of bivariate functions defined as follows: For every function $\varphi_{r_1,\ldots,r_U} \in \Phi$ and every $i = 1, \ldots, U$,

$$\varphi_{r_1,\ldots,r_U}(x,i) = \begin{cases} 0, & \text{for } x = 1,\ldots,U - \sum_{j=1}^{i} r_j; \\ x - U + \sum_{j=1}^{i} r_j, & \text{for } x \geq U - \sum_{j=1}^{i} r_j + 1 \text{ and } \sum_{j=1}^{i} r_j \geq 1. \end{cases}$$

Clearly, any pair of elements of Φ are binary distinct. We refer to $\{\varphi_{r_1,\ldots,r_U}(x,i) \mid x = 1,\ldots,U\}$ as the ith row of $\varphi_{r_1,\ldots,r_U}$. Note that the first row contains $U - r_1$ zeros, and for $i > 1$, the ith row contains r_i less zeros than the $(i-1)$st row. Note also that since $\sum_{i=1}^{U} r_i = U$, we have $\varphi_{r_1,\ldots,r_U}(x,U) = x$.

Consider any $\varphi_{r_1,\ldots,r_U} \in \Phi$. Clearly, $\varphi_{r_1,\ldots,r_U}$ is a nondecreasing function. In addition, it is not difficult to check that $\varphi_{r_1,\ldots,r_U}$ is discretely convex (and the detailed convexity proof is omitted).

We now determine the cardinality of the family Φ. Because $r_1 \leq r_2 \leq \cdots \leq r_U$ and $\sum_{i=1}^{U} r_i = U$, each combination of $r_1, \ldots, r_U$ is a partition of the integer U into at most U positive integers. Let $p(U)$ be the number of partitions of U (note: a partition of a positive integer U is a set consisting of positive numbers whose sum is U). The number of combinations in our case is $p(U)$; that is, $|\Phi| = p(U)$. Note that $p(U) > \frac{H}{U} e^{2\sqrt{U}}$ for some positive constant H [HR18, eq. (2.11)]. Hence, $p(U) > He^{\sqrt{U}} \geq H \cdot 2^{\sqrt{U}}$, and the validity of the proposition follows. $\square$

Let $\mathcal{F}$ be the family consisting of all nondecreasing discretely convex functions $\varphi : [1,U]^2 \rightarrow [0,U]$, and let $\text{bin}\mathcal{F} = \{bin(f) \mid f \in \mathcal{F}\}$. Proposition 2.2 tells us that we need in general $\Omega(\sqrt{U})$ space to represent a function from $\text{bin}\mathcal{F}$. Since this term is not polylogarithmic in the domain size U^2, nor in $\frac{\varphi^{\max}}{\varphi^{\min}} < U^2$, there exists a function $\varphi' \in \text{bin}\mathcal{F}$ that does not admit a succinct representation. This, together with Proposition 2.1, implies that there exists a function $\varphi \in \mathcal{F}$ with $bin(\varphi) = \varphi'$ which does not admit a succinct K-approximation for any $1 \leq K < 2$. Hence, we have proved Theorem 1.2.

3 Approximating φ

3.1 The monotone case

A related (and easier) problem is when φ is known to be monotone (and not necessarily convex). This problem was solved in [HKL+13] as follows.

Proposition 3.1 (Succinct approximation of a monotone function via a K-approximation general oracle, Proposition 4.7 in [HKL+13])**.** *Let $\varphi : [A,B] \rightarrow \mathbb{R}^+$ be a monotone function of real numbers, $L > 1$ be a constant, and $\tilde{\varphi}$ be a (not necessarily monotone) L-approximation function of φ. Then for every $K = 1 + \epsilon > 1$, it is possible to construct in $O(\frac{1+t_{\tilde{\varphi}}}{\epsilon}(1 + \log \frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}}) \log(B - A))$ time a monotone step KL-approximation function of φ with $O(1 + \frac{1}{\epsilon} \log \frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}})$ steps.*

This result was proved in [HKL+13] with general finite domains D. We choose to present it with $D = [A,B]$ for the sake of simplicity.

For the sake of completeness we provide a self-contained proof of this theorem, together with a statement of function IndirectApxInc, by proving in the Appendix the following somewhat stronger result:

Proposition 3.2. *Let $\varphi : [A,B] \rightarrow \mathbb{R}^+$ be a function of real numbers, $L > 1$ be a constant, and $\tilde{\varphi}$ be a (not necessarily monotone) L-approximation function of φ. Let $t_{\tilde{\varphi}}$ be an upper bound on the time needed to evaluate $\tilde{\varphi}(x)$. Then for every $K = 1 + \epsilon > 1$, Function* IndirxtApxInc *(Algorithm 6) runs in $O(\frac{1+t_{\tilde{\varphi}}}{\epsilon}(1 + \log \frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}}) \log(B - A))$ time and constructs a set W of size $O(1 + \frac{1}{\epsilon} \log \frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}})$ and a function $\psi : [A,B] \rightarrow \mathbb{R}^+$ such that*

$$\psi(y) \leq K\psi(x), \quad \forall \text{ consecutive } x < y \in W \text{ with } y - x \geq 2.$$

Moreover, if φ is increasing then ψ is an increasing step KL-approximation function of φ with $O(|W|)$ steps.

For the analogue case of decreasing functions we design in a similar way function IndirectApxDec, and thus get a result similar to Proposition 3.2.

3.2 The convex case

3.2.1 A high-level description of the algorithm

The algorithm for approximating a convex φ is surprisingly much more involved than in the monotone case, maybe because one needs to first approximate an argmin of φ, and then use it to build an approximation function of $\tilde{\varphi}$. Note that finding (exactly) an argmin of φ is generally not possible because an L-approximation function $\tilde{\varphi}$ of φ induces only a partial order over $\{\varphi(x) \mid x \in [A, B]\}$. (If $\tilde{\varphi}$ induces a complete linear order then the convex case will decompose into two monotone cases). We summarize this in the following simple proposition (whose proof is omitted).

Proposition 3.3. *Let $\varphi : [A, B] \to \mathbb{R}^+$, $L > 1$, and $\tilde{\varphi}$ be an L-approximation function of φ. If $\tilde{\varphi}(x_1) \geq L\tilde{\varphi}(x_2)$ then $\varphi(x_1) \geq \varphi(x_2)$. Else, if $\tilde{\varphi}(x_1) \leq \frac{\tilde{\varphi}(x_2)}{L}$ then $\varphi(x_1) \leq \varphi(x_2)$. Otherwise, ($L\tilde{\varphi}(x_2) > \tilde{\varphi}(x_1) > \frac{\tilde{\varphi}(x_2)}{L}$), the order between $\varphi(x_1)$ and $\varphi(x_2)$ cannot be deduced by querying $\tilde{\varphi}$.*

The algorithm for approximating a convex φ consists of 5 functions. IndirectApxConvex is the the outer-level function, SmartSearch is the high-level search procedure to find an element x' for which $\tilde{\varphi}(x')$ is lower than a given threshold, EquidistanceSearch is a low-level search procedure for finding such an x', and Consecutive and Shrink are two auxiliary functions.

EquidistanceSearch($\tilde{\varphi}, A, B, C, K, L, q$)) is the basic search procedure. Given oracle access to $\tilde{\varphi} : [A, B] \to \mathbb{R}$, an upper bound C of $\tilde{\varphi}^{\max}$, and real positive numbers K, L, it performs a number of evenly-spaced queries sufficient to find an element x' for which $\tilde{\varphi}(x') < KLY$, ($Y = C/K^q$). The idea behind this procedure is simple. If $\tilde{\varphi}^{\min} < Y$, then when the query points are close enough to each other, one is guaranteed to find at least one such element x'. When the ratio between C and Y is small, the number of sufficient such queries is polynomial in the input size. Otherwise, a more sophisticated search procedure needs to be called, namely SmartSearch.

Shrink($\tilde{\varphi}, A, B, L$) exploits the convexity of φ to reduce ("shrink") the interval $[A, B]$ from one of its endpoints. If the endpoints of the interval are unbalanced, i.e, the ratio between $\tilde{\varphi}(A)$ and $\tilde{\varphi}(B)$ is greater that L^2, then Shrink "cuts" away a piece from one of the sides of $[A, B]$ and returns a balanced interval $[A', B']$.

Consecutive($\tilde{\varphi}, A, B, M$) facilitates reducing the interval $[A, B]$ from the inside. Given a value M between $\tilde{\varphi}(A)$ and $\tilde{\varphi}(B)$, it returns two consecutive points a', A' such that $\tilde{\varphi}(a') \leq M$ and $\tilde{\varphi}(A') \geq M$.

Given oracle access to $\tilde{\varphi} : [A, B] \to \mathbb{R}$, and real positive numbers K, L, q, q^*, SmartSearch($\tilde{\varphi}, A, B, K, L, q, q^*$) returns an element x' with $\tilde{\varphi}(x') < KLY$, ($Y = C/K^q$ where $C = \max\{\tilde{\varphi}(A), \tilde{\varphi}(B)\}$). The parameter q^* bounds the number of queries Q^*, EquidistanceSearch is allowed to perform when called by SmartSearch. The algorithm starts with a call to Shrink in hope that the new maximal value of $\tilde{\varphi}$ on the reduced interval, C', will be small enough, so that the ratio between C' and Y will not be too large, and therefore the number of equidistance queries needed to find x' will not exceed Q^*. If unsuccessful, it calls EquidistanceSearch with a new value $y > Y$ (but still $y \ll C$), gets an element x'' for which $\tilde{\varphi}(x'') < y$, and then calls Consecutive twice. Let (a', A') =Consecutive($\tilde{\varphi}, A, x, y$) and (b', B') =Consecutive($\tilde{\varphi}, x, B, y$) be the results of these calls. Thus the interval $[a', b']$ is a smaller interval that contains x'' and $\tilde{\varphi}(a'), \tilde{\varphi}(b') \ll C'$. If needed, this process is repeated, until the requested x' is found.

Finally, we describe function IndirectApxConvex($\tilde{\varphi}, A, B, K, L$), which calculates a K^3L^2-approximation function of $\tilde{\varphi} : [A, B] \to \mathbb{Z}^+$. The function first checks whether $\min \tilde{\varphi} = 0$ by a call to SmartSearch. If the answer is in the positive, then a KL-approximation for φ is calculated by splitting the domain into two parts, over which φ is monotone. Otherwise, by performing a binary search (through calls to SmartSearch), the algorithm finds a value M and its argument x' for which $\varphi^{\min} > M$ and $\tilde{\varphi}(x') < K^2LM$. Using x' as an approximated argmin of φ, the algorithm splits the interval $[A, B]$ into two parts and tries to calculate monotone KL-approximations of φ over $[A, x']$ and $[x', B]$, acting *as if* φ *were* monotone over these two intervals. Since φ is not necessarily monotone over these intervals, a local correction may be requested, and the resulting approximation factor may deteriorate up to K^3L^2.

3.2.2 The algorithm

We start by stating function EquistableSearch. This function deals with the following question. Suppose we are told that the minimum value of φ is less than Y, i.e., there exists a point x' with $\varphi(x') < Y$. Since $\tilde{\varphi}$ is an L-approximation of φ, we have $\tilde{\varphi}(x') < LY$. Can we find such an x' efficiently? Since $\tilde{\varphi}$ is not necessarily convex, the answer to this question is in the negative (e.g., $\varphi(x) = Y - 1 + |x|$ and is defined over $[-Y, Y]$, $\tilde{\varphi}(x) = 2Y, \forall x \neq 0$ and $\tilde{\varphi}(0) = Y - 1$. Then $\tilde{\varphi}$ is a 2-approximation function of vp, whose minimizer cannot be found efficiently). But as it turns out, we can find $\tilde{x}$ such that $\tilde{\varphi}(\tilde{x}) < KLY$, where $K = 1 + \epsilon$ for infinitely small $\epsilon > 0$.

```
1: Function EquidistanceSearch(φ̃, A, B, C, K, L, q)
2: Perform ⌈q⌉K^(⌈q⌉−1) equidistance queries of φ̃ in [A, B].
3: if there exists a query point x with φ̃ < KLC/K^q then
4:     return x
5: else
6:     return ∞
7: end if
```

Algorithm 1: Returning a point on which the value of $\tilde{\varphi}$ is smaller than KLC/K^q, where $\tilde{\varphi}(x) \leq C$, $\forall x$.

Lemma 3.4. *Let $\varphi : [A, B] \to \mathbb{R}^+$ be a convex function, $K, L > 1$ be arbitrary real numbers, and $\tilde{\varphi}$ be an L-approximation function of φ. For every upper bound $C \geq \max\{\tilde{\varphi}(A), \tilde{\varphi}(B)\}$ and $q \in \mathbb{R}$, let $Y = \frac{C}{K^q}$. If there exists x^* such that $\varphi(x^*) < Y$ then function* EquidistanceSearch *(Algorithm 1) finds in $O((1 + t_{\tilde{\varphi}})qK^{q-1})$ time a set of points $\tilde{X} \subset [A, B]$ such that $\tilde{\varphi}(x) < KLY$ for all $x \in \tilde{X}$ and there exists $\tilde{x} \in \tilde{X}$ such that $\varphi(\tilde{x}) < KY$.*

Proof. Let

$$Y = C/K^q, \qquad x^0 = \arg\min\{\varphi(x) < Y\}, \qquad x' = \arg\max\{\varphi(x) < Y\}, \qquad x'' = \arg\max\{\varphi(x) < KY\},$$

and

$$U_2 = [x', B], \qquad U_2' = [x', x''], \qquad U_{12} = U_{12}' = [x^0, x'],$$

see Figure 1. We will show below that the smaller the ratio $\frac{|U_2|}{|U_2'|}$ is, the easier it becomes to find a point $\tilde{x}$ with $\varphi(\tilde{x}) < KY$. Therefore the worst is when $\frac{|U_2|}{|U_2'|}$ is the largest possible. Since φ is convex over $[A, B]$, this ratio is maximized if φ is linear over U_2, see Figure 1. By triangle similarity we get

$$\frac{|U_2|}{|U_2'|} = \frac{\varphi(B) - Y}{KY - Y} \leq \frac{C - Y}{KY - Y} = \frac{K^q - 1}{K - 1} \leq \sum_{i=0}^{\lceil q \rceil - 1} K^i \leq \lceil q \rceil K^{\lceil q \rceil - 1}.$$

Repeating the same argument for $U_1 = \{x < x^0 \mid \varphi(x) \geq Y\}$, we get that $\frac{|U_1|}{|U_1'|} \leq \lceil q \rceil K^{\lceil q \rceil - 1}$. Therefore, we get that $\frac{|U_1|+|U_{12}|+|U_2|}{|U_1'|+|U_{12}'|+|U_2'|} \leq \lceil q \rceil K^{\lceil q \rceil - 1}$ (recall that $\frac{U_{12}}{U_{12}'} = 1$). We conclude the proof by performing $\lceil q \rceil K^{\lceil q \rceil - 1}$ equally-spaced queries of $\tilde{\varphi}$ over $[A, B]$, which by our calculation must include a point $\tilde{x} \in U_1' \cup U_{12}' \cup U_2'$, i.e., a point $\tilde{x}$ with $\tilde{\varphi}(\tilde{x}) \leq L\varphi(\tilde{x}) < KLY$. ◻

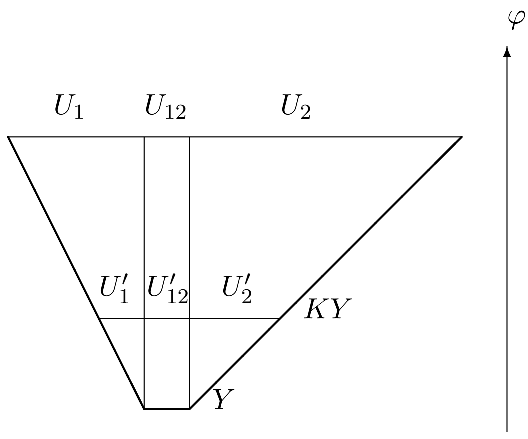

Figure 1: Finding values of φ smaller than KLY.

Remark. Note that if only one of the queries performed by Algorithm 1 satisfies $\tilde{\varphi}(x') \leq KLY$, then we must have $\varphi(x') \leq KY$. If more than one such point exists, say x', x'', by Proposition 3.3 we may not be able to deduce whether $\varphi(x') \leq KY$, or $\varphi(x'') \leq KY$, or both holds.

When the ratio between C and Y is "large", e.g., $q \approx \log_K C$, the number of queries performed by EquidistanceSearch may not be polynomially bounded by the input size. We next aim to give a more efficient algorithm (that we call SmartSearch) for finding values of φ smaller than KLC/K^q for large q's. This is done at the cost of bounding the ratio between K and L.

Before doing so, we state two auxiliary functions, Consecutive and Shrink

Function Consecutive$(\tilde{\varphi}, A, B, C)$
$amin_1 \leftarrow \arg\min\{\tilde{\varphi}(A), \tilde{\varphi}(B)\},\ amax_1 \leftarrow \arg\max\{\tilde{\varphi}(A), \tilde{\varphi}(B)\},\ j \leftarrow 0$
while $|amin_{j+1} - amax_{j+1}| > 1$ **do**
    $j \leftarrow j+1,\ mid \leftarrow \lfloor (amin_j + amax_j)/2 \rfloor$
    if $\tilde{\varphi}(mid) < C$ **then**
        $amin_{j+1} \leftarrow mid,\ amax_{j+1} \leftarrow amax_j$
    else
        $amin_{j+1} \leftarrow amin_j,\ amax_{j+1} \leftarrow mid$
    end if
end while
return $(amin_{j+1}, amax_{j+1})$

Algorithm 2: Returning 2 consecutive points $amin, amax \in [A, B]$ with $\tilde{\varphi}(amin) \leq C$ and $\tilde{\varphi}(amax) \geq C$.

Proposition 3.5. *Let $\tilde{\varphi} : [A, B] \to \mathbb{R}^+$ be an arbitrary function and $C \in \mathbb{R}^+$ be a constant. If $\max\{\tilde{\varphi}(A), \tilde{\varphi}(B)\} \geq C$ and $\min\{\tilde{\varphi}(A), \tilde{\varphi}(B)\} \leq C$ then function* Consecutive *(Algorithm 2) finds in $O((1 + t_{\tilde{\varphi}}) \log(B - A))$ time two consecutive points $amin, amax \in [A, B]$ with $\tilde{\varphi}(amin) \leq C$ and $\tilde{\varphi}(amax) \geq C$.*

Proof. At the beginning of the while loop the following inequalities hold for $j^* = 1$:

$$\tilde{\varphi}(amin_{j^*}) \leq C, \qquad \tilde{\varphi}(amax_{j^*}) \geq C. \tag{2}$$

Note that by the updates done in lines 6 and 8, the invariant (2) continues to hold for larger values of j^*. The algorithm exits the while loop when $|amin_{j^*} - amax_{j^*}| = 1$ so (2) holds for two consecutive elements. The running time of the algorithm follows from the fact that at each iteration of the while loop the size of the domain, i.e., $|amax_j - amin_j|$, is cut by half. □

We next state function Shrink that finds a subset $[A', B'] \subseteq [A, B]$ that contains an argmin of a convex function φ, where the ratio between $\tilde{\varphi}(A')$ and $\tilde{\varphi}(B')$ is bounded by L^2.

1: **Function** Shrink$(\tilde{\varphi}, A, B, L)$
2: $A_1 \leftarrow A,\ B_1 \leftarrow B,\ i \leftarrow 0$
3: **while** $\max\{\frac{\tilde{\varphi}(A_{i+1})}{\tilde{\varphi}(B_{i+1})}, \frac{\tilde{\varphi}(B_{i+1})}{\tilde{\varphi}(A_{i+1})}\} > L^2$ and $\min\{\tilde{\varphi}(A_{i+1}), \tilde{\varphi}(B_{i+1})\} > 0$ **do**
4: $\quad i \leftarrow i+1,\ C_i \leftarrow \max\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}$
5: $\quad (amin, amax) \leftarrow$ Consecutive$(\tilde{\varphi}, A_i, B_i, C_i/L)$
6: $\quad A_{i+1} \leftarrow \min\{amin, \arg\min\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}\},\ B_{i+1} \leftarrow \max\{amin, \arg\min\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}\}$
7: **end while**
8: **return** $[A_{i+1}, B_{i+1}]$

Algorithm 3: Returning a sub domain of $[A, B]$ that contains an argmin of a convex function φ, where the ratio between the values of $\tilde{\varphi}$ on its endpoints is bounded by L^2.

We prove in the Appendix the following lemma:

Lemma 3.6. *Let $\varphi : [A, B] \to \mathbb{R}^+$ be a convex function, $L > 1$ be a constant, and $\tilde{\varphi}$ be an L-approximation function of φ. Function* Shrink *(Algorithm 3) returns in $O((1+t_{\tilde{\varphi}})(1+\log_L(\frac{\varphi^{\max}}{\max\{\tilde{\varphi}(A'),\tilde{\varphi}(B')\}}))\log(B-A))$ time a sub domain $[A', B']$ that contains an argmin of φ over $[A, B]$, where the ratio between $\tilde{\varphi}(A')$ and $\tilde{\varphi}(B')$ is at most L^2.*

Remark: The maximal running time of Shrink is $O((1 + t_{\tilde{\varphi}})\log_L(\frac{\varphi^{\max}}{\varphi^{\min}})\log(B - A))$, and is realized whenever $\tilde{\varphi}(A') = \tilde{\varphi}(B') \approx \varphi^{\min}$.

We are ready to state function SmartSearch$(\tilde{\varphi}, A, B, K, L, q, q^*)$, which returns a point x' and an interval $[A', B'] \ni$, on which the value of $\tilde{\varphi} : [A, B]\mathbb{R}^+$ is smaller than KLY, $\quad (Y = C/K^q, \quad C = \max\{\tilde{\varphi}(A), \tilde{\varphi}(B)\})$, when the number of equidistance queries done at a single call to EquidistanceSearch is never more than $Q^* = q^* K^{q^*-1}$. The idea behind SmartSearch is as follows - if $q \leq q^*$, then simply call EquidistanceSearch. Otherwise, shrink the domain in which an argmin of φ lies in the following two ways. First, if the ratio between the values of $\tilde{\varphi}$ on the endpoints of the domain is greater than L^2, then call function Shrink to get a smaller interval $[A', B']$. Now the ratio between $\max\{\tilde{\varphi}(A'), \tilde{\varphi}(B')\}$ and Y decreases, so Q^* equidistance queries over $[A', B']$ may be sufficient in order to find x'. If this is the case - call EquidistanceSearch. Otherwise, shrink the domain in the second way, i.e., perform Q^* equidistance queries over $[A', B']$, get a point x'' on which $\tilde{\varphi}(x'') \leq KLY'$ $(Y' > Y)$, and reduce the domain to $[A'', B'']$ such that $\max\{\tilde{\varphi}(A'), \tilde{\varphi}(B')\}/\max\{\tilde{\varphi}(A''), \tilde{\varphi}(B'')\} > L^2 K$, via two calls to Consecutive. Now the ratio between $\max\{\tilde{\varphi}(A''), \tilde{\varphi}(B'')\}$ and Y decreases even further, so Q^* equidistance queries over $[A'', B'']$ may be sufficient in order to find x'. If so, simply call EquidistanceSearch. Otherwise - repeat the process of shrinking the domain in the two ways described above. The process is guaranteed to stop after at most $O(\log LC)$ iterations.

The proof of the following lemma is provided in the Appendix:

Lemma 3.7. *Let $\varphi : [A, B] \to \mathbb{R}^+$ be a convex function, $K > 1$, $q^* \in \mathbb{N}$ be arbitrary numbers and $q \in \mathbb{R}^+$ be a real number satisfying $q \geq q^*$. Let $L \in \mathbb{R}^+$ be an arbitrary number satisfying $K^{\frac{q^*}{4}-\frac{1}{2}} \geq L \geq 1$. Let $\tilde{\varphi}$ be an L-approximation function of φ, $C = \max\{\tilde{\varphi}(A), \tilde{\varphi}(B)\}$ and $Y = \frac{C}{K^q}$. If there exists a point x such*

Function SmartSearch($\tilde{\varphi}, A, B, K, L, q, q^*$)
$C_0 \leftarrow \max\{\tilde{\varphi}(A), \tilde{\varphi}(B)\},\ A_0 \leftarrow A,\ A_1 \leftarrow A,\ B_0 \leftarrow B,\ B_1 \leftarrow B,\ q_0 \leftarrow q,\ i \leftarrow 1$
while $\min\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\} > LC/K^{q-1}$ **do**
  if $\frac{\max\{\tilde{\varphi}(A_i),\tilde{\varphi}(B_i)\}}{\min\{\tilde{\varphi}(A_i),\tilde{\varphi}(B_i)\}} > L^2$ **then**
    $[A_i, B_i] \leftarrow$ Shrink($\tilde{\varphi}, A_i, B_i, L$)
  end if
  $C_i \leftarrow \max\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\},\ R_i \leftarrow C_{i-1}/C_i,\ r_i \leftarrow \log_K R_i,\ q_i \leftarrow q_{i-1} - r_i$
  if $q_i \leq q^*$ **then**
    return (EquidistanceSearch($\tilde{\varphi}, A_i, B_i, C_i, K, L, q_i$)$, A_i, B_i$)
  end if
  $\tilde{x} \leftarrow$ EquidistanceSearch($\tilde{\varphi}, A_i, B_i, C_i, K, L, q^*$)
  if $\tilde{x} = \infty$ **then**
    return (∞, A_i, B_i)
  end if
  $(A_{i+1}, amaxA) \leftarrow$ Consecutive($\tilde{\varphi}, A_i, \tilde{x}, \frac{C_i}{L^2 K}$)
  $(B_{i+1}, amaxB) \leftarrow$ Consecutive($\tilde{\varphi}, \tilde{x}, B_i, \frac{C_i}{L^2 K}$)
  $i \leftarrow i + 1$
end while
return($\arg\min\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}, A_i, B_i$)

Algorithm 4: Finding a point on which the value of $\tilde{\varphi}$ is smaller than $KL\max\{\tilde{\varphi}(A), \tilde{\varphi}(B)\}/K^q$ and an interval on which a minimum of φ is attained.

that $\varphi(x) < Y$ then function SmartSearch *(Algorithm 4) finds in*

$$\mathrm{O}\left((1+t_{\tilde{\varphi}})\left[\frac{q-q^*}{1+\log_K L}\ \ q^* K^{q^*-1} + \max\left\{\frac{q-q^*}{1+\log_K L}, \log_L \frac{\varphi^{\max}}{\varphi^{\min}}\right\}\log(B-A)\right]\right)$$

time a point $\tilde{x}$ such that $\tilde{\varphi}(\tilde{x}) < KLY$. Moreover, φ is decreasing over $[A, A_{i^*+1}]$ and increasing over $[B_{i^*+1}, B]$, where i^* is the number of times the while loop was executed.

We now state function IndirectApxConvex which constructs a convex $(1+\epsilon)L^2$-approximation of φ.

Overview of function IndirectApxConvex. We would like to split the domain of φ into two intervals, according to where φ is decreasing or increasing. In line 3, the algorithm checks whether $\min \varphi(x) = 0$ by performing a call to SmartSearch with a positive query value of less than 1. Since φ is integer-valued, the only such possible value is 0. We first consider the case where SmartSearch returns x' such that $\tilde{\varphi}(x') = 0$. Note that in this case, because $\tilde{\varphi}$ is a relative approximation of φ, we also have $\varphi(x') = 0$. This means the algorithm was successful in splitting the domain $[A, B]$ into two parts in where φ is monotone: φ is decreasing over $[A, x']$ and increasing over $[x', B]$. In this case the condition in line 4 is not satisfied, and the algorithm goes to line 7. In this line, the algorithm builds on each one of the intervals $[A, x']$ and $[x^*, B]$ a piecewise-linear approximations of φ (using IndirectApxDec, IndirectApxInc), and stores the corresponding breakpoints in W_D, W_I, respectively. Note that since φ is monotone in each one of these intervals, Proposition 3.1 assures us that we get a KL-approximation in both intervals. The algorithm then jumps to line 15, sets ψ to be the concatenation of ψ_D and ψ_I, and returns the greatest convex function that lies below ψ. (We note in passing that since $\tilde{\varphi}(x') = 0$, we get that $\psi_D(y_D) = \psi_I(y_I) = 0$, so neither one of the conditions in lines 9 and 12 is satisfied, so indeed the algorithm jumps to line 15. We note also that the concatenation operation is well defined since $W_D \cap W_I = \{x'\}$ and $\psi_D(x') = \psi_I(x')$.)

We next consider the case where the condition in line 4 is met, i.e., the minimal value of φ is strictly positive. Since $\tilde{\varphi}$ is not necessarily convex, it may be too costly to find an exact realizer of the minimum value of $\tilde{\varphi}$, so instead, the algorithm calculates an approximated argmin x' which is returned by a call to

1: **Function** IndirectApxConvex($\tilde{\varphi}, A, B, K, L$)
2: $q^* \leftarrow \lceil 2 + 4\log_K L \rceil,\ C \leftarrow \max\{\tilde{\varphi}(A), \tilde{\varphi}(B)\}$,
3: $(x', A', B') \leftarrow$ SmartSearch($\tilde{\varphi}, A, B, K, L, 1 + \lceil \log_K LC \rceil, q^*$)
4: **if** $x' = \infty$ (i.e., $\min \tilde{\varphi} \geq 1$) **then**
5: by binary search, find the maximal integer $q' \in [1, \lceil \log_K C \rceil]$ for which $(x', A', B') \leftarrow$ SmartSearch($\tilde{\varphi}, A, B, K, L, q', q^*$) satisfies $\tilde{\varphi}(x') < \frac{KLC}{K^{q'}}$
6: **end if**
7: $(\psi_D, W_D) \leftarrow$ IndirectApxDec($\tilde{\varphi}, A, x', K$), $(\psi_I, W_I) \leftarrow$ IndirectApxInc($\tilde{\varphi}, x', B, K$)
8: $y_D \leftarrow \min_{w \in W_D} \psi_D(w) = \psi_D(x'),\ \ y_I \leftarrow \max_{w \in W_I} \psi_I(w) = \psi_I(x')$
9: **if** $\psi_D(x') < \psi_I(x')$ or ($\psi_D(x') = \psi_I(x')$ and $\psi_I(x') < \tilde{\varphi}(x')$) **then**
10: $W_D \leftarrow W_D \cap [A, y_D],\ W_I \leftarrow W_I \setminus \{x'\} \cup \{y_D + 1\},\ \psi_I(x) \leftarrow \tilde{\varphi}(y_I),\ \forall x \in [y_D + 1, y_I]$
11: **end if**
12: **if** $\psi_D(x') > \psi_I(x')$ **then**
13: $W_D \leftarrow W_D \setminus \{x'\} \cup \{y_I - 1\},\ W_I \leftarrow W_I \cap [y_I, B],\ \psi_D(x) \leftarrow \tilde{\varphi}(y_D),\ \forall x \in [y_D, y_I - 1]$
14: **end if**
15: Let $\psi : [A, B] \to \mathbb{R}^+$ be the concatenation of $\psi_D : W_D \to \mathbb{R}^+$ and $\psi_I : W_I \to \mathbb{R}^+$
16: **return** the convex extension of ψ induced by $W_D \cup W_I$

Algorithm 5: Approximating a convex function that is accessed via an L-approximation function $\tilde{\varphi}$.

SmartSearch in line 5. (Note that due to Lemma 3.7, φ is guaranteed to be decreasing over $[A, A']$ and increasing over $[B', B]$. Also note that neither φ nor $\tilde{\varphi}$ is necessarily monotone over either $[A, x']$ or $[x', B]$.) The algorithm then enters line 7 and constructs the two functions ψ_D, ψ_I by calling IndirectApxDec and IndirectApxInc with the approximated argmin x'. Therefore, due to Proposition 3.2, ψ_D is a decreasing step functions with breakpoints in W_D, and ψ_I is an increasing step function with breakpoints in W_I. Let y_D be the least minimizer of ψ_D, and y_I be the greatest minimizer of ψ_I, see line 8.

If the algorithm is lucky, no smaller minimum was found, i.e., $\psi_D(x') = \psi_I(x') = \tilde{\varphi}(x')$, and the algorithm jumps to line 15. By the construction of the ψ functions (e.g., see line 23 in function IndirectApxInc which implies that $\psi(x) \leq \tilde{\varphi}(x),\ \forall x \in W_D \cup W_I$), we get that ψ is a unimodal step function over $[A, B]$ with $O(|W_D \cup W_I|)$ steps which is minimized at x', so:

$$\psi(x') = \tilde{\varphi}(x'),\ \ \psi \text{ is decreasing in } [A, x'],\ \ \psi \text{ is increasing in } [x', B],\ \text{and } \psi(x) \leq \tilde{\varphi}(x),\ \forall x \in W_D \cup W_I. \tag{3}$$

Otherwise ($\psi_D(x') \neq \psi_I(x')$ or $\psi_I(x') < \tilde{\varphi}(x')$), the algorithm performs a local correction of ψ_D and ψ_I in lines 9-14, so that a concatenation of ψ_D and ψ_I is possible, and where (3) still holds.

Using (3), it is possible to prove that ψ is indeed a KL^2-approximation of φ, as we do in the proof of lemma 3.8 below. We prove the following lemma in the Appendix:

Lemma 3.8. *Let $\varphi : [A, B] \to \mathbb{Z}^+$ be a convex function, $L > 1$ be a constant, $K = 1 + \epsilon > 1$ be a constant, and $\tilde{\varphi}$ be an L-approximation function of φ. Function* IndirectApxConvex *(Algorithm 5), when called with parameters $\tilde{\varphi}, A, B, \sqrt[3]{K}, L$, constructs in*

$$\mathrm{O}\left(\frac{1 + t_{\tilde{\varphi}}}{\epsilon} \log \tilde{\varphi}^{\max} \left[\log\log \tilde{\varphi}^{\max} + \log(1/\epsilon)\right] \left[L^4 + \frac{logK}{\log \min\{K, L\}} \log(B - A)\right]\right)$$

time a convex piecewise-linear KL^2-approximation function for φ that has $O(\frac{\log \tilde{\varphi}^{\max}}{\epsilon})$ breakpoints.

Remark. If φ is not known to be integer-valued but $\varphi^{\min}$ is provided, then we can approximate φ as before. All we need to do is to multiply $\tilde{\varphi}$ by $\frac{1}{\varphi^{\min}}$ and proceed as before. Of course, the running time will increase, as $\varphi^{\max}$ increases to $\frac{\varphi^{\max}}{\varphi^{\min}}$.

4 Conclusions and future research

Let $\tilde{\varphi}$ be an L approximation function of a convex function φ. In this paper, given an oracle access to $\tilde{\varphi}$, we construct an efficient convex KL^2-approximation of φ. We consider first the computation time in Theorem 1.1. Note that the dependency of the running time on ϵ is $\frac{1}{\epsilon}\log\frac{1}{\epsilon}$, and on $\frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}}$ is $\frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}}\log\frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}}$. Is it possible to improve the running time? Taking advantage of the convexity of φ, and using the slope trick in [HNO13], it may be possible to reduce the term $\frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}}$ to $\min\{\frac{\sigma_{\varphi}^{\max}}{\sigma_{\tilde{\varphi}}^{\min}}, \frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}}\}$.

We next consider the approximation ratio in Theorem 1.1. If L is not constant, but we have an FPTAS for φ instead (so $L = 1+\delta > 1$ for arbitrary small δ), then we have the following easy corollary.

Theorem 4.1 (Succinct approximation of a convex function via an FPTAS general oracle)**.** *A convex function $\varphi : [A, B]\to\mathbb{Z}^+$ that cannot be evaluated direcly, but only via an FPTAS admits a convex FPTAS.*

An interesting open question arises when L is a constant. The approximation ratio in Theorem 1.1 is L^2K. We would like to reduce it to KL. A possible idea of how to do so is if we could have assured that the point x' function IndirectApxConvex gets from function SmartSearch is the $\tilde{x}$ that is defined in the statement of Lemma 3.4. Then (17) in the proof of Lemma 3.8 would have changed to

$$\psi(x) \leq \psi(b) \leq K\psi(a) \leq K\tilde{\varphi}(a) \leq K^2LY \leq K^3LY.$$

The explanation of the forth inequality is as follows. Since φ is decreasing over $[\tilde{x}, a]$ we get $\varphi(a) \leq \varphi(\tilde{x}) \leq KY$. Since $\tilde{\varphi}$ is an L-approximation of φ we get then $\tilde{\varphi}(a) \leq L\varphi(a) \leq KLY$. The last inequlity is due to the second inequality in (13).

We note that even though the oracle function $\tilde{\varphi}$ is unstructured (i.e., it is neither monotone nor convex), the *knowledge* that it approximates a convex function is instrumental to our algorithm design. This raises the following question. Suppose φ has no structure. From the discussion in the Introduction, we know that it does not necessarily admit an efficient succinct approximation. Therefore, if we still want to have such an approximation, we must impose an additional constraint, e.g., that φ is *close* to being structured. By close we mean that φ admits an L-approximation $\tilde{\varphi}$ that is structured. Can one then effectively approximate φ? It turns out that the answer is in the positive. And the fact that we have access to a structured function facilitates the proof of the following two results considerably (a proof is given in the Appendix).

Proposition 4.2 (Succinct approximation of a general function via an L-approximation monotone oracle)**.** *Let $\varphi : [A, B]\to\mathbb{R}^+$ be a nonnegative real-valued function, $L > 1$ be a constant, and $\tilde{\varphi}$ be a monotone L-approximation function of φ. Then for every $K = 1+\epsilon > 1$, it is possible to construct in $O(\frac{1+t_{\tilde{\varphi}}}{\epsilon}\log\frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}}\log(B-A))$ time a monotone step KL-approximation function of φ with $O(1+\frac{1}{\epsilon}\log\frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}})$ steps.*

Using the slope trick in [HNO13], we get an improved space and running time in the convex case.

Proposition 4.3 (Succinct approximation of a general function via an L-approximation convex oracle)**.** *Let $\varphi : [A, B]\to\mathbb{Z}^+$ be a nonnegative integer-valued function, $L > 1$ be a constant, and $\tilde{\varphi}$ be a convex L-approximation function of φ. Then for every $K = 1+\epsilon > 1$, it is possible to construct in $O(\frac{1+t_{\tilde{\varphi}}}{\epsilon}\log\min\{\frac{\sigma_{\varphi}^{\max}}{\sigma_{\tilde{\varphi}}^{\min}}, \frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}}\}\log(B-A))$ time a piecewise-linear convex KL-approximation function of φ with $O(1+\frac{1}{\epsilon}\log\min\{\frac{\sigma_{\varphi}^{\max}}{\sigma_{\tilde{\varphi}}^{\min}}, \frac{\tilde{\varphi}^{\max}}{\tilde{\varphi}^{\min}}\})$ breakpoints.*

We conclude this section by considering the impossibility result in Theorem 1.2. This result deals with a specific, although quite general, class of discrete convex functions. But there are other classes of discrete convex functions, see Figure 1.15 in [Mur03] which depicts the inclusion relationship among these classes.

It is easy to show that Theorem 1.1 can be extended to hold also for the class of multivariate *separable* convex functions, which is the most restrictive class in the figure. It is interesting to distinguish between the classes of multivariate discrete convex functions that admit efficient succinct approximations, and those who don't. Moreover, for the former case it is desirable to design such efficient succinct approximations.

Recently, [CDJ14] studied fixed-dimensional stochastic dynamic programs in a discrete setting over a finite horizon, under the primary assumption that the cost-to-go functions are discrete $L^\natural$-convex. They proposed a pseudo-polynomial time approximation scheme that solves multi-dimensional dynamic programs to within an arbitrary pre-specified *additive* error of $\epsilon > 0$. The proposed approximation algorithm is a generalization of the explicit-enumeration algorithm, offers a full control in the tradeoff between accuracy and running time, but runs in time pseudo-polynomial in the input size. If the class of discrete $L^\natural$-convex functions turns out not to admit efficient succinct approximations, then their result is in a way best possible. It is interesting to give a result of this type for all classes of discrete convex functions that do not admit efficient succinct approximations.

Acknowledments: The author thanks Jim Orlin for fruitful discussions on an earlier version of this paper, and Chung-Lun Li for improving the presentation of the proof of Theorem 1.2.

References

[ACCL08] N. Ailon, B. Chazelle, S. Comandur, and D. Liu. Property-preserving data reconstruction. *Algorithmica*, 51:160–182, 2008.

[BLR90] M. Blum, M. Luby, and R. Rubinfeld. Self-testing/correcting with application to numerical problems. In *Proceedings of the 22th Annual ACM-SIAM Symposium on Theory of Computation*, pages 73–83, 1990.

[CDJ14] W. Chen, M. Dawande, and G. Janakiraman. Fixed-dimensional stochastic dynamic programs: An approximation scheme and an inventory application. *Operations Research*, to appear, 2014.

[CNC10] M.Y. Kovalyov C.T. Ng and T.C.E. Cheng. A simple FPTAS for a single-item capacitated economic lot-sizing problem with a monotone cost structure. *European Journal of Operational Research*, 200:621–624, 2010.

[FT90] P. Favati and F. Tardella. Convexity in nonlinear integer programming. *Ricerca Operativa*, 53:3–44, 1990.

[HKL+08] N. Halman, D. Klabjan, C.-L. Li, J. Orlin, and D. Simchi-Levi. Fully polynomial time approximation schemes for stochastic dynamic programs. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 700–709, 2008.

[HKL+13] N. Halman, D. Klabjan, C.-L. Li, J. Orlin, and D. Simchi-Levi. Fully polynomial time approximation schemes for stochastic dynamic programs. Technical report 3918, Optimization Online, June 2013.

[HKM+09] N. Halman, D. Klabjan, M. Mostagir, J. Orlin, and D. Simchi-Levi. A fully polynomial time approximation scheme for single-item stochastic inventory control with discrete demand. *Mathematics of Operations Research*, 34:674–685, 2009.

[HNO13] N. Halman, G. Nannicini, and J. Orlin. A computationally efficient fptas for convex stochastic dynamic programs. In *Proceedings of ESA 2013, volume 8125 of Lecture Notes in Computer Science*, pages 577–588. Springer, 2013.

[HR18] G.H. Hardy and S. Ramanujan. Asymptotic formulae in combinatory analysis. *Proceedings of the London Mathematical Society*, 17:75–115, 1918.

[Mil71] B.L. Miller. On minimizing nonseparable functions defined on the integers with an inventory application. *SIAM Journal on Applied Mathematics*, 21:166–185, 1971.

[Mur03] K. Murota. *Discrete Convex Analysis*. SIAM, Philadelphia, 2003.

[RS96] R. Rubinfeld and M. Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing*, 25:647–668, 1996.

[SS10] M. Saks and C. Seshadhri. Local monotonicity reconstruction. *SIAM Journal on Computing*, 39(7):2897–2926, 2010.

Appendix

A Function IndirectApxInc

1: **Function** IndirectApxInc($\tilde{\varphi}, A, B, K$)
2: $x \leftarrow B$ and $W \leftarrow \{A, B\}$
3: **while** $x > A + 1$ and $K\tilde{\varphi}(A) < \tilde{\varphi}(x)$ **do**
4: $\quad j \leftarrow 0,\ A_1 \leftarrow A,\ B_1 \leftarrow B,\ mid \leftarrow \lfloor (A+B)/2 \rfloor,\ s \leftarrow mid$
5: $\quad$ **while** $K\tilde{\varphi}(\min\{mid, s\}) \geq \tilde{\varphi}(x)$ or $K\tilde{\varphi}(\max\{mid, s\}) < \tilde{\varphi}(x)$ **do**
6: $\quad\quad mid \leftarrow \lfloor (A_j + B_j)/2 \rfloor$
7: $\quad\quad$ **if** $K\tilde{\varphi}(mid) < \tilde{\varphi}(x)$ **then**
8: $\quad\quad\quad j \leftarrow j + 1,\ s \leftarrow mid + 1$
9: $\quad\quad\quad$ **if** $K\tilde{\varphi}(s) < \tilde{\varphi}(x)$ **then**
10: $\quad\quad\quad\quad A_{j+1} \leftarrow s,\ B_{j+1} \leftarrow B_j$
11: $\quad\quad\quad$ **end if**
12: $\quad\quad$ **else**
13: $\quad\quad\quad s \leftarrow mid - 1$
14: $\quad\quad\quad$ **if** $K\tilde{\varphi}(s) \geq \tilde{\varphi}(x)$ **then**
15: $\quad\quad\quad\quad A_{j+1} \leftarrow A_j,\ B_{j+1} \leftarrow s$
16: $\quad\quad\quad$ **end if**
17: $\quad\quad$ **end if**
18: $\quad$ **end while**
19: $\quad x \leftarrow \min\{mid, s\},\ W \leftarrow W \cup \{x, x+1\}$
20: **end while**
21: Define a function $\psi : [A, B] \rightarrow \mathbb{R}^+$ as follows: $\psi(B) \leftarrow \tilde{\varphi}(B),\ x \leftarrow B$
22: **while** $x \neq A$ **do**
23: $\quad \psi(\mathrm{prev}(x, W)) \leftarrow \min\{\tilde{\varphi}(\mathrm{prev}(x, W)), \tilde{\varphi}(x)\},\ x \leftarrow \mathrm{prev}(x, W)$
24: **end while**
25: Extend the definition of ψ to $[A, B]$ by setting $\psi(z) \leftarrow \psi(\mathrm{next}(z, W))$ for every $z \notin W$
26: **return** $(\psi(\cdot), W)$

Algorithm 6: Approximating an increasing function that is accessed via an approximation function $\tilde{\varphi}$.

B Proof of Proposition 3.2

Proof. We start by considering the cardinality of W and the running time of the algorithm. We call the while loop that starts at line 3 the *outer loop* and the while loop that starts at line 5 the *inner loop*. We first consider the inner. Note that for each iteration of the outer loop, the inner loop is executed at least once because we have $mid = s$ at the first time the condition of the inner loop is checked. Note also that in the remaining times the condition of the inner loop is checked (in the same iteration of the outer loop) we have $|s - mid| = 1$. In the first iteration of the inner loop the interval is $[A_1, B_1]$ and the condition of the outer loop implies that $K\tilde{\varphi}(A_1) < \tilde{\varphi}(x)$. In addition, $K\tilde{\varphi}(B_1) \geq \tilde{\varphi}(x)$ (since $B_1 = x$). The algorithm chooses a middle element $mid \in [A_1, B_1]$. The algorithm considers two different cases. Case 1: $K\tilde{\varphi}(mid) < \tilde{\varphi}(x)$. In this case, the algorithm chooses a second element $s \leftarrow mid + 1$. If $K\tilde{\varphi}(s) \geq \tilde{\varphi}(x)$, then the inner loop is completed by assigning $x \leftarrow mid$. Otherwise, the algorithm sets a new (reduced) interval $[A_2, B_2] \leftarrow [s, B_1]$. Case 2: $K\tilde{\varphi}(mid) \geq \tilde{\varphi}(x)$. In this case, the algorithm sets $s \leftarrow mid - 1$. If

$K\tilde{\varphi}(s) < \tilde{\varphi}(x)$, then the inner loop is completed by assigning $x \leftarrow s$. Otherwise, the algorithm sets a new (reduced) interval $[A_2, B_2] = [A_1, s]$. In both cases, if the inner loop is not completed, then we get a new interval $[A_2, B_2]$ of size at most half of the one of $[A_1, B_1]$, and it satisfies $K\tilde{\varphi}(A_2) < \tilde{\varphi}(x)$ and $K\tilde{\varphi}(B_2) \geq \tilde{\varphi}(x)$. The inner loop continues this way. Clearly, the inner loop is exhausted in $O(\log(B-A))$ steps. Note that for every consecutive elements $x, y \in W$ with $y > x+1$ we have $K\tilde{\varphi}(x) < \tilde{\varphi}(y)$. Thus, the outer loop repeats at most $O(1 + \log_K \frac{\varphi^{\max}}{\varphi^{\min}})$ times. Clearly, the cardinality of W is $O(1 + \log_K \frac{\varphi^{\max}}{\varphi^{\min}})$. The computational time required in each iteration of the outer loop is $O(t_{\tilde{\varphi}} \log(B - A))$, and since the loop that starts at line 22 runs only $|W|$ times, the claimed running time of the algorithm follows. We note in passing that indeed

$$\psi(y) \leq K\psi(x), \quad \forall \text{ consecutive } x < y \in W \text{ with } y - x \geq 2. \tag{4}$$

We next prove that if φ is increasing over $[A, B]$ then ψ is an increasing KL-approximation step function of φ. By the construction of ψ in line 23, we have $\psi(x) \leq \tilde{\varphi}(x)$ for any $x \in W$. This, together with the fact that $\tilde{\varphi}$ is an L-approximation of φ, implies that

$$\psi(x) \leq L\varphi(x), \quad \forall x \in W.$$

On the other hand, for any $x \in W$, there exists $y \in W$ such that $y \geq x$ and $\psi(x) = \tilde{\varphi}(y)$. Because $\tilde{\varphi}$ is an L-approximation of φ, we have $\tilde{\varphi}(y) \geq \varphi(y)$. Thus,

$$\psi(x) = \tilde{\varphi}(y) \geq \varphi(y) \geq \varphi(x), \quad \forall x \in W, \tag{5}$$

where the second inequality is due to the monotonicity of φ. Hence, ψ is an increasing L-approximation step function of the restriction of φ over W.

We conclude the proof by considering the approximation ratio of ψ over $[A, B] \setminus W$. Let $x \in [A, B] \setminus W, y = \mathrm{next}(x, W), z = \mathrm{prev}(x, W)$. Due to line 25 we have $\psi(x) = \psi(y)$, thus

$$\psi(x) = \psi(y) \leq K\psi(z) \leq K\tilde{\varphi}(z) \leq KL\varphi(z) \leq KL\varphi(x).$$

(The first inequality is due to (4), the second one is due to line 23, the third one is because $\tilde{\varphi}$ is an L-approximation function of φ, and the last one is due to the monotonicity of φ).

On the other hand we have

$$\psi(x) = \psi(y) \geq \varphi(y) \geq \varphi(x),$$

where the first inequality is due to (5) and the second one is due to the monotonicity of φ. Therefore, ψ is a KL-approximation of φ over $[A, B]$. □

C Proof of Lemma 3.6

Proof. The algorithm enters the while loop only if (recall that $C_i \leftarrow \max\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}$):

$$\tilde{\varphi}(\arg\min\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}) < \frac{C_i}{L^2}. \tag{6}$$

Therefore, the call to CONSECUTIVE with C_i/L_i is well defined. By Proposition 3.5, we get in line 5 two *consecutive* elements $amin, amax$ in $[A_i, B_i]$ with

$$\tilde{\varphi}(amin) \leq \frac{C_i}{L}, \qquad \tilde{\varphi}(amax) \geq \frac{C_i}{L}. \tag{7}$$

Due to (6) and (7)

$$\tilde{\varphi}(amax) \geq \frac{C_i}{L} > L\tilde{\varphi}(\arg\min\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}),$$

so by Proposition 3.3 we get

$$\varphi(amax) > \varphi(\arg\min\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}).$$

Considering the points $amax, A_i, B_i$, due to the convexity of φ we get that φ is increasing over

$$\Big[\min\{amax, \arg\max\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}\},\ \max\{amax, \arg\max\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}\}\Big].$$

Hence, φ attains a minimum in

$$[A_{i+1}, B_{i+1}] := \Big[\min\{amin, \arg\min\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}\},\ \max\{amin, \arg\min\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}\}\Big].$$

Note that the value of $\tilde{\varphi}$ over the endpoints of the new domain $[A_{i+1}, B_{i+1}]$ is upper bounded by $\frac{C_i}{L}$. Therefore, the number of iterations of the while loop is $O(\log_L \frac{\tilde{\varphi}^{\max}}{\max\{\tilde{\varphi}(A'), \tilde{\varphi}(B')\}})$, where in each iteration Consecutive is called once. The claimed overall running time follows. $\square$

D Proof of Lemma 3.7

Proof. Let $Y \leftarrow C_0/K^{q_0} = C/K^q$. We first consider the ratio between the values of $\tilde{\varphi}$ on the endpoints A_i, B_i. When the algorithm reaches line 7, we must have

$$\min\{\tilde{\varphi}(A_i),\ \tilde{\varphi}(B_i)\} \geq \frac{\max\{\tilde{\varphi}(A_i), \tilde{\varphi}(B_i)\}}{L^2} > \frac{C_i}{L^2 K}. \tag{8}$$

(if the condition in line 4 is met, then a Shrink operation is called.) We next consider R_i, i.e., the ratio between the maximal value of $\tilde{\varphi}$ on A_{i-1}, B_{i-1} and the maximal value of $\tilde{\varphi}$ on A_i, B_i. Note that $R_i > 1$ implies that either the condition in line 4 was met (i.e., Shrink was executed), or lines 15 and 16 were executed in the previous iteration of the while loop as we explain below. The algorithm updates the exponent q_i of the K in order to keep

$$\frac{C_i}{K^{q_i}} = \frac{\frac{C_{i-1}}{R_i}}{K^{q_{i-1}-r_i}} = \frac{C_{i-1}}{K^{q_{i-1}}} = \ldots = Y \tag{9}$$

invariant throughout the execution of the algorithm.

Regarding the condition in line 8, if q_i is relatively small (i.e., $q_i \leq q^*$) then Equidistance-Search is not too costly, and the algorithm exits by executing Equidistance-Search with the original bound $Y = C_i/K^{q_i}$ and returning its outcome. Otherwise, the algorithm enters line 11 and executes Equidistance-Search with a larger value of bound (i.e., C_i/K^{q^*}). If Equidistance-Search does not find a value of $\tilde{\varphi}$ smaller than KLC_i/K^{q^*}, clearly $\tilde{\varphi}$ does not have a value smaller than $Y = KLC_i/K^{q_i}$, so the algorithm exits by returning the negative answer (∞, A_i, B_i). Otherwise, Equidistance-Search finds a value of $\tilde{\varphi}$ smaller than KLC_i/K^{q^*} together with its argument $\tilde{x}$. I.e.,

$$\tilde{\varphi}(\tilde{x}) < \frac{KLC_i}{K^{q^*}} \leq \frac{C_i}{L^3 K}, \tag{10}$$

where the second inequality is due to the lemma's assertion that $L \leq K^{\frac{q^*}{4}-\frac{1}{2}}$. (Note that the calls to EquidistanceSearch are valid because C_i is indeed an upper bound on the maximal value of $\tilde{\varphi}$ over

$[A_i, B_i]$). We now turn to the calls to CONSECUTIVE. In line 15, the algorithm calls CONSECUTIVE and finds two consecutive elements $amaxB, A_{i+1} \in [A_i, \tilde{x}]$ with (note that $amaxB = A_{i+1} - 1$)

$$\tilde{\varphi}(amaxA) \geq \frac{C_i}{L^2 K}, \qquad \tilde{\varphi}(A_{i+1}) \leq \frac{C_i}{L^2 K}. \tag{11}$$

(Note that this call is valid due to (8) and (10)). The algorithm also calls CONSECUTIVE in line 16 and finds two consecutive elements $B_{i+1}, amaxB \in [\tilde{x}, B_i]$ satisfying (note that $amaxB = B_{i+1} + 1$)

$$\tilde{\varphi}(B_{i+1}) \leq \frac{C_i}{L^2 K}, \qquad \tilde{\varphi}(amaxB) \geq \frac{C_i}{L^2 K}. \tag{12}$$

(Note again that this call is valid due to (8) and (10)). Due to Proposition 3.3 and (10)-(12) we get

$$\varphi(A_{i+1} - 1) \geq \varphi(\tilde{x}) \leq \varphi(B_{i+1} + 1).$$

Due to the convexity of φ we get that it is decreasing over $[A, A_{i+1}]$ and increasing over $[B_{i+1}, B]$. Hence, φ can achieve a value of less than C_i/K^{q_i} only in $[A_{i+1}, B_{i+1}]$. Note that the value of $\tilde{\varphi}$ on the endpoints of this domain is upper bounded by $\frac{C_i}{L^2 K}$. Therefore, the updated ratio satisfies $R_i > L^2 K$. Therefore, in the next iteration of the while loop the new exponent of K, q_i, will be smaller than the old one by at least $1 + 2\log_K L$. Thus, the number of iterations is at most $(\lfloor q \rfloor - q^*)/(1 + 2\log_K L)$. The overall running time of the algorithm follows because each iteration of the algorithm performs one EQUIDISTANCESEARCH and at most two CONSECUTIVE operations, and the overall running time of the various calls to SHRINK is bounded by $(1 + t_{\tilde{\varphi}})(\log_L \frac{\varphi^{\max}}{\varphi^{\min}} \log(B - A)$.

∎

E Proof of Lemma 3.8

Proof. We analyze INDIRECTAPXCON when called with parameters $\tilde{\varphi}, A, B, K, L$. For the sake of brevity, we will only prove that ψ is an increasing $K^3 L^2$-approximation of φ over $[x', B]$. The proof that ψ is a decreasing $K^2 L$-approximation of φ over domain $[A, x']$ is similar. If the condition in line 4 is not met, then $\tilde{\varphi}(x') = 0$. We get then $0 = \tilde{\varphi}(x') \geq \varphi(x') \geq 0$, where the first inequality is due to $\tilde{\varphi}$ being an L-approximation of φ and the second one is due to the nonnegativity of φ. Thus, φ is increasing over $[x', B]$, and due to Proposition 3.1, our ψ is a KL-approximation of φ over $[x', B]$.

We next consider the case where the condition in line 4 is met. Suppose first that $\psi_D(x') = \psi_I(x') = \tilde{\varphi}(x')$, i.e., no lower value of $\tilde{\varphi}$ was discovered while performing line 7. Note that in this case (3) holds, as explained in the description of the algorithm. In the proof we will use 2 more equations, namely, (13) and (14). Let $Y' = C/K^{q'}$. Line 5 coupled with Lemmas 3.4 and 3.7 imply the correctness of the inequalities

$$\tilde{\varphi}(x') < KLY', \quad \varphi(x) \geq \frac{Y'}{K}, \ \forall x \in [A, B]. \tag{13}$$

(The second inequality is due to the maximality of q'.) Recall that Proposition 3.2 tells us that

$$\psi(y) \leq K\psi(x), \quad \forall \text{ consecutive } x < y \in W_I \text{ with } y - x \geq 2. \tag{14}$$

Let $x^* = \arg\min_{x \in [A,B]} \varphi(x)$ be a realizer of the optimal value of φ, and let $b = \min\{x \geq x^* \mid x \in W_I\}$. We distinguish between two cases. (i) If $x^* \leq x'$ we are done, since then φ is increasing over $[x', B]$, and again due to Proposition 3.1, our ψ is a KL-approximation of φ over $[x', B]$. (ii) If, on the other hand, $x^* > x'$ and we consider the approximation ratio over $[b, B]$ then because φ is increasing over this interval, Proposition 3.1 tells us that our ψ is a KL-approximation of φ over $[b, B]$. Otherwise, $x^* > x'$ and consider the approximation ratio on $x \in [x', b]$. We consider below each of the upper and lower bounds of the approximation ratio.

Lower bound: if $x \in [x', x^*]$ then we have

$$\psi(x) \geq \psi(x') = \tilde{\varphi}(x') \geq \varphi(x') \geq \varphi(x), \tag{15}$$

where the first inequality and the equality are due to (3), the second inequality is due to $\tilde{\varphi}$ being an L-approximation of φ, and the last one is due to φ being decreasing over $[x', x^*]$. If $x \in [x^*, b]$ then

$$\psi(x) = \psi(b) = \tilde{\varphi}(c) \geq \varphi(c) \geq \varphi(x),$$

where the first two equalities are by the construction of ψ (lines 25 and 23, respectively, in IndirectApx-Inc, where $c \geq b$), the first inequality is due to $\tilde{\varphi}$ being an L-approximation of φ, and the last one is due to φ being increasing over $[x^*, b]$. We summarize the above two equations:

$$\psi(x) \geq \varphi(x), \quad \forall x \in [x', b]. \tag{16}$$

Upper bound: If $x^* = b$ then

$$\psi(x) \leq \psi(b) = \psi(x^*) \leq \tilde{\varphi}(x^*) \leq L\varphi(x^*) \leq L\varphi(x),$$

where the first two inequalities are due to (3), the third one is due to $\tilde{\varphi}$ being an L-approximation of φ, and the last one is since x^* is a minimizer of φ. If, $x^* < b$ then $b - a \geq 2$ and we use (14):

$$\psi(x) \leq \psi(b) \leq K\psi(a) \leq K\tilde{\varphi}(a) \leq KL\tilde{\varphi}(x') \leq K^3L^2\varphi(x), \tag{17}$$

where the first and third inequalities are due to (3), and the second inequality is due to (14). Regarding the forth inequality - note that the monotonicity of φ over $[x', a]$ implies $\varphi(a) \leq \varphi(x')$. Proposition 3.3 then tells us that $\tilde{\varphi}(a) \leq L\tilde{\varphi}(x')$. The last inequality is due to both inequalities in (13). We summarize the above two equations:

$$\psi(x) \leq K^3L^2\varphi(x), \quad \forall x \in [x', b]. \tag{18}$$

We conclude from (16)-(18) that ψ is a K^3L^2-approximation of φ over $[x', b]$.

It remains to deal with the case where $\psi_D(x') \neq \psi_I(x')$ or $\psi_D(x') = \psi_I(x') < \tilde{\varphi}(x')$, i.e., the algorithm performs changes to the original domains W_D, W_I and functions ψ_D, ψ_I in either line 10 or 13. Note that in the case that $\psi_D(x') > \psi_I(x')$, ψ over $[x', B]$ is a restriction of the original ψ_I that was constructed by IndirectApxInc, so the same analysis above still holds. In the other two cases, the approximated argmin is moved left to y_D, and ψ over $[x' + 1, B]$ consists of the original ψ_I "glued" with the constant function over $[y_D + 1, y_I]$ with value $\tilde{\varphi}(y_I)$. (Note: the intuition for which this correction is successful is that the value of the constant function is the value of $\tilde{\varphi}$ on the *right* endpoint of the domain. Also in line 23 in IndirectApxSet such corrections were made.) We will show that this modified function keeps the claimed approximation error. Going over the proof for the case of $\psi_D(x') = \psi_I(x') < \tilde{\varphi}(x')$ (with $x' = y_D + 1$), only 2 inequalities need be to be proved, namely (15) and (17). Considering (15), we will prove it for $x \in [y_D, x^*]$ (instead of $x \in [y_D + 1, x^*]$). Indeed,

$$\psi(x) \geq \psi(y_D) = \tilde{\varphi}(y_D) \geq \varphi(y_D) \geq \varphi(x),$$

The first inequality hold because the corrected ψ is increasing in $[y_D, B]$, and the other inequalities still hold from the same reasons. Regarding (17), the proof remains unchanged, except for the case where $a = y_D + 1$

(recall that the first two elements in W_I are $y_D + 1$ and y_I, so $b = y_I$, and that $\psi(y_D + 1) = \psi(y_I)$). In this case we have an even smaller upper bound, i.e.,

$$\psi(x) \leq \psi(b) = \psi(y_I) \leq \tilde{\varphi}(x') \leq K^2 L\varphi(y),$$

where the first inequality is due to the definition of y_I. We note in passing that this is no surprise - the fact the the algorithm found a smaller realizer for $\tilde{\varphi}$ than x' should only improve the approximation ratio.

It remains to consider the running time of the algorithm. Clearly, in the worst case, the running time is bounded by the time it takes to perform the binary search in line 5. There are $O(\log(\log_K C - q^*))$ calls to SmartSearch. Taking the right value of q^* and transforming bases of the log to 2 we get the claimed running time. □

F Proof of Proposition 4.2

Proof. We prove the theorem for the case $\tilde{\varphi}$ is increasing. The proof for the case $\tilde{\varphi}$ is decreasing is similar and hence omitted. Let ψ =IndirectApxInc$(\tilde{\varphi}, A, B, K)$. By Proposition 3.2,

$$\psi(y) \leq K\psi(x), \quad \forall \text{ consecutive } x < y \in W \text{ with } y - x \geq 2. \tag{19}$$

Let $z \in [A, B]$. If $z \in W$ then $\psi(z) = \tilde{\varphi}(z)$, so the fact that $\tilde{\varphi}$ is an L-approximation of φ implies $\varphi(z) \leq \psi(z) \leq Lvp(z)$. Otherwise, let $x = \text{prev}(z, W)$ and $y = \text{next}(z, W)$. We get that

$$\varphi(z) \leq \tilde{\varphi}(z) \leq \tilde{\varphi}(y) = \psi(y) = \psi(z) \leq K\psi(x) = K\tilde{\varphi}(x) \leq K\tilde{\varphi}(z) \leq KL\varphi(z).$$

(The first and last inequalities are due to $\tilde{\varphi}$ being an L-approximation of φ. The equalities are due to the fact that the monotonicity of $\tilde{\varphi}$ implies $\psi(t) = \tilde{\varphi}(t)$, $\forall t \in W$. The second and forth inequalities are due to the monotonicity of $\tilde{\varphi}$. The third inequality is due to (19).) Therefore, ψ is a KL-approximation of φ. The cardinality of W and the time needed to build ψ derive both from Proposition 3.2. □

G Proof of Proposition 4.3

Proof. We use function ApxSetSlope from [HNO13]. Let W =ApxSetSlope$(\tilde{\varphi}, [A, B], K)$. Let ψ be the piecewise linear extension of $\tilde{\varphi}$ induced by W. Then, Definition 3.1 coupled with Theorem 3.2 in [HNO13] imply that ψ is a K-approximation of $\tilde{\varphi}$. Therefore, applying twice the definition of approximation functions we get that

$$\varphi(z) \leq \tilde{\varphi}(z) \leq \psi(z) \leq K\tilde{\varphi}(z) \leq KL\varphi(z), \ \forall z \in [A, B],$$

so ψ is a KL-approximation of φ. The cardinality of W and the time needed to build ψ derive both from Theorem 3.2 in [HNO13]. □