Finding monotone patterns in sublinear time

Omri Ben-Eliezer
Tel-Aviv University
Tel-Aviv, Israel
omrib@mail.tau.ac.il

Clément L. Canonne
Stanford University
Stanford, USA
ccanonne@cs.stanford.edu

Shoham Letzter
ETH Institute for Theoretical Studies
Zurich, Switzerland
shoham.letzter@eth-its.ethz.ch

Erik Waingarten
Columbia University
New York, USA
eaw@cs.columbia.edu

***Abstract*—We study the problem of finding monotone subsequences in an array from the viewpoint of sublinear algorithms. For fixed $k \in \mathbb{N}$ and $\varepsilon > 0$, we show that the non-adaptive query complexity of finding a length-k monotone subsequence of $f\colon [n] \to \mathbb{R}$, assuming that f is ε-far from free of such subsequences, is $\Theta((\log n)^{\lfloor \log_2 k \rfloor})$. Prior to our work, the best algorithm for this problem, due to Newman, Rabinovich, Rajendraprasad, and Sohler (2017), made $(\log n)^{O(k^2)}$ non-adaptive queries; and the only lower bound known, of $\Omega(\log n)$ queries for the case $k = 2$, followed from that on testing monotonicity due to Ergün, Kannan, Kumar, Rubinfeld, and Viswanathan (2000) and Fischer (2004).**

***Keywords*-property testing; algorithms; lower bounds; one-sided testing; monotone patterns; forbidden patterns**

I. Introduction

For a fixed integer $k \in \mathbb{N}$ and a function (or sequence) $f\colon [n] \to \mathbb{R}$, a *length-$k$ monotone subsequence of f* is a tuple of k indices, $(i_1, \ldots, i_k) \in [n]^k$, such that $i_1 < \cdots < i_k$ and $f(i_1) < \cdots < f(i_k)$. More generally, for a permutation $\pi\colon [k] \to [k]$, a *π-pattern of f* is given by a tuple of k indices $i_1 < \cdots < i_k$ such that $f(i_{j_1}) < f(i_{j_2})$ whenever $j_1, j_2 \in [k]$ satisfy $\pi(j_1) < \pi(j_2)$. A sequence f is π-free if there are no subsequences of f with order pattern π. Recently, Newman, Rabinovich, Rajendraprasad, and Sohler [1] initiated the study of property testing for forbidden order patterns in a sequence. Their paper was the first to analyze algorithms for finding π-patterns in sublinear time (for various classes of the permutation π); additional algorithms and lower bounds for several classes of permutations have later been obtained by Ben-Eliezer and Canonne [2].

Of particular interest of π-freeness testing is the case where $\pi = (12\ldots k)$, i.e., π is a monotone permutation. In this case, avoiding length-k monotone subsequence may be equivalently rephrased as being decomposable into $k-1$ monotone non-increasing subsequences. Specifically, a function $f\colon [n] \to \mathbb{R}$ is $(12\ldots k)$-free if and only if $[n]$ can be partitioned into $k-1$ disjoint sets $A_1, \ldots, A_{k-1}$ such that, for each $i \in [k-1]$, the restriction $f|_{A_i}$ is non-increasing. When interested in algorithms for testing $(12\ldots k)$-freeness that have a *one-sided error*,[1] the algorithmic task becomes the following:

> *For $k \in \mathbb{N}$ and $\varepsilon > 0$, design a randomized algorithm that, given query access to a function $f\colon [n] \to \mathbb{R}$ guaranteed to be ε-far from being $(12\ldots k)$-free,[2] outputs a length-k monotone subsequence of f with probability at least $9/10$.*

The task above is a natural generalization of monotonicity testing of a function $f\colon [n] \to \mathbb{R}$ with algorithms that make a one-sided error, a question which dates back to the early works in property testing, and has received significant attention since in various settings (see, e.g., [3]–[10], and the recent textbook [11]). For the problem of testing monotonicity, Ergün, Kannan, Kumar, Rubinfeld, and Viswanathan [12] were the first to give a non-adaptive algorithm which tests monotonicity of functions $f\colon [n] \to \mathbb{R}$ with one-sided error making $O(\log(n)/\varepsilon)$ queries. (Recall that an algorithm is *non-adaptive* if its queries do not depend on the answers to previous queries, or, equivalently, if all queries to the function can be made in parallel.) Furthermore, they showed that $\Omega(\log n)$ queries are necessary for non-adaptive algorithms. Subsequently, Fischer [13] showed that $\Omega(\log n)$ queries are necessary even for adaptive algorithms. Generalizing from monotonicity testing (when $k = 2$), Newman et al. gave in [1] the first sublinear-time algorithm for $(12\ldots k)$-freeness testing, whose query complexity is $(\log(n)/\varepsilon)^{O(k^2)}$. Their algorithm is non-adaptive and has one-sided error; as such, it outputs a length-k monotone subsequence with probability at least $9/10$ assuming the function f is ε-far from $(12\ldots k)$-free. However, other than the aforementioned lower bound of $\Omega(\log n)$ which follows from the case $k = 2$, no lower bounds were known for larger k.

The main contribution of this work is to settle the dependence on n in the query complexity of testing for $(12\ldots k)$-freeness with non-adaptive algorithms making one-sided error. Equivalently, we settle the complexity of non-adaptively finding a length-k monotone subsequence under the promise that the function $f\colon [n] \to \mathbb{R}$ is ε-far from $(12\ldots k)$-free.

Theorem I.1. *Let $k \in \mathbb{N}$ be a fixed parameter. For any $\varepsilon > 0$, there exists an algorithm that, given query access to a function $f\colon [n] \to \mathbb{R}$ which is ε-far from $(12\ldots k)$-*

[1] An algorithm for testing property $\mathcal{P}$ is said to have *one-sided error* if the algorithm always outputs "yes" if $f \in \mathcal{P}$ (perfect completeness).

[2] A function $f\colon [n] \to \mathbb{R}$ is *ε-far* from π-free if any π-free function $g\colon [n] \to \mathbb{R}$ satisfies $\mathbf{Pr}_{\boldsymbol{i} \sim [n]}[f(\boldsymbol{i}) \neq g(\boldsymbol{i})] \geq \varepsilon$.

2575-8454/19/$31.00 ©2019 IEEE
DOI 10.1109/FOCS.2019.00089

free, outputs a length-k monotone subsequence of f with probability at least $9/10$. The algorithm is non-adaptive and makes $(\log n)^{\lfloor \log_2 k \rfloor} \cdot \mathrm{poly}(1/\varepsilon)$ queries to f.

Our algorithm thus significantly improves on the $(\log(n)/\varepsilon)^{O(k^2)}$-query non-adaptive algorithm of [1]. Furthermore, its dependence on n is optimal; indeed, in the next theorem we prove a matching lower bound for all fixed $k \in \mathbb{N}$.

Theorem I.2. *Let $k \in \mathbb{N}$ be a fixed parameter. There exists a constant $\varepsilon_0 > 0$ such that any non-adaptive algorithm which, given query access to a function $f\colon [n] \to \mathbb{R}$ that is ε_0-far from $(12\dots k)$-free, outputs a length-k monotone subsequence with probability $9/10$, must make $\Omega((\log n)^{\lfloor \log_2 k \rfloor})$ queries. Moreover, one can take $\varepsilon_0 = 1/(4k)$.*

We further note that the lower bound holds even for the more restricted case where $f\colon [n] \to [n]$ is a permutation.

A. Related work

Testing monotonicity of a function over a partially ordered set $\mathcal{X}$ is a well-studied and fruitful question, with works spanning the past two decades. Particular cases include when $\mathcal{X}$ is the line $[n]$ [8]–[10], [12], [13], the Boolean hypercube $\{0,1\}^d$ [3], [14]–[23], and the hypergrid $[n]^d$ [24]–[26]. We refer the reader to [11, Chapter 4] for more on monotonicity testing, or for an overview of the field of property testing (as introduced in [27], [28]) in general.

This paper is concerned with the related line of work on finding order patterns in sequences and permutations. For the exact case, Guillemot and Marx [29] showed that an order pattern π of length k can be found in a sequence f of length n in time $2^{O(k^2 \log k)}n$; in particular, the problem of finding order patterns is fixed-parameter tractable (in the parameter k). Fox [30] later improved the running time to $2^{O(k^2)}n$. A very recent work of Kozma [31] provides the state-of-the-art for the case where $k = \Omega(\log n)$. In the sublinear regime, the most relevant works are the aforementioned papers of Newman et al. [1] and Ben-Eliezer and Canonne [2]. In particular, [1] shows an interesting dichotomy for testing π-freeness: when π is monotone, the non-adaptive query complexity is polylogarithmic in n for fixed k and ε, whereas for non-monotone π, the query complexity is $\Omega(\sqrt{n})$. Two related questions are that of estimating the *distance to monotonicity* and the length of the *longest increasing subsequence* (LIS), which have also received significant attention from both the sublinear algorithms perspective [32]–[34], as well as the streaming perspective [35]–[39]. In particular, Saks and Seshadhri gave in [34] a randomized algorithm which, on input $f\colon [n] \to \mathbb{R}$, makes $\mathrm{poly}(\log n, 1/\delta)$ queries and outputs $\widehat{m}$ approximating up to additive error δn the length of the longest increasing subsequence of f. This paper also studies monotone subsequences of the input function, albeit from a different (and incomparable) end of the problem. Loosely speaking, in [34] the main object of interest is a very *long* monotone subsequence (of length linear in n), and the task at hand is to get an estimate for its total length, whereas in our setting, there are $\Omega(n)$ disjoint copies of *short* monotone subsequences (of length k, which is a constant parameter), and these short subsequences may not necessarily combine to give one long monotone subsequence.

B. Our techniques: Upper bound

We now give a detailed overview of the techniques underlying our upper bound, Theorem I.1, and provide some intuition behind the algorithms and notions we introduce. The starting point of our discussion will be the algorithm of Newman et al. [1], which we re-interpret in terms of the language used throughout this paper; this will set up some of the main ideas behind our structural result (stated in Section II), which will be crucial in the analysis of the algorithm.

For simplicity, let $\varepsilon > 0$ be a small constant and let $k \in \mathbb{N}$ be fixed. Consider a function $f\colon [n] \to \mathbb{R}$ which is ε-far from $(12\dots k)$-free. This implies that there is a set $T \subseteq [n]^k$ of $\varepsilon n/k$ *disjoint* $(12\dots k)$-patterns. Specifically, the set T is comprised of k-tuples $(i_1,\dots,i_k) \in [n]$ where $i_1 < \dots < i_k$ and $f(i_1) < \dots < f(i_k)$ and each $i \in [n]$ appears in at most one k-tuple in T.[3] A key observation made in [1] is that if, for some $c \in [k-1]$, $(i_1,\dots,i_c,i_{c+1},\dots,i_k)$ and $(j_1,\dots,j_c,j_{c+1},\dots,j_k)$ are two k-tuples in T which satisfy $i_c < j_{c+1}$ and $f(i_c) < f(j_{c+1})$, then their combination

$$(i_1,\dots,i_c,j_{c+1},\dots,j_k)$$

is itself a length-k monotone subsequence of f. Therefore, in order to design efficient sampling algorithms, one should analyze to what extent parts of different $(12\dots k)$-tuples from T may be combined to form length-k monotone subsequences of f.

Towards this goal, assign to each k-tuple $(i_1,\dots,i_k)$ in T a *distance profile* $\textsf{dist-prof}(i_1,\dots,i_k) = (d_1,\dots,d_{k-1}) \in [\eta]^{k-1}$, where $\eta = O(\log n)$.[4] This distance profile is a $(k-1)$-tuple of non-negative integers satisfying

$$2^{d_j} \le i_{j+1} - i_j < 2^{d_j+1} \qquad j \in [k-1]\,;$$

and let $\textsf{gap}(i_1,\dots,i_k) = c \in [k-1]$ be the smallest integer where $d_c \ge d_j$ for all $j \in [k-1]$ (i.e., d_c denotes an (approximately) maximum length between two adjacent indices in the k-tuple). Suppose, furthermore, that for a particular $c \in [k-1]$, the subset $T_c \subseteq T$ of k-tuples whose gap is at c satisfies $|T_c| \ge \varepsilon n/k^2$ (such a $c \in [k-1]$

[3] To see why such T exists, take T to be a maximal set of disjoint $(12\dots k)$-patterns. Suppose $|T| < \varepsilon n/k$ and consider the function g given by greedily eliminating all $(12\dots k)$-patterns in f, and note that g is $(12\dots k)$-free and differs on f in less than εn indices.

[4] We remark that the notion of a distance profile is solely used for the introduction and for explaining [1], and thus does not explicitly appear in subsequent sections.

must exist since the T_c's partition T). If $(i_1, \ldots, i_k) \in T_c$ and $\mathsf{dist\text{-}prof}(i_1, \ldots, i_k) = (d_1, \ldots, d_k)$, then the probability that a uniformly random element $\boldsymbol{\ell}$ of $[n]$ "falls" into that gap is

$$\Pr_{\boldsymbol{\ell} \sim [n]}[i_c \leq \boldsymbol{\ell} \leq i_{c+1}] \geq \frac{2^{d_c}}{n}. \tag{1}$$

Whenever this occurs for a particular k-tuple $(i_1, \ldots, i_k)$ and $\boldsymbol{\ell} \in [n]$, we say that $\boldsymbol{\ell}$ *cuts* the tuple $(i_1, \ldots, i_k)$. Note that the indices $i_{c+1}, \ldots, i_k$ are contained within the interval $[\boldsymbol{\ell}, \boldsymbol{\ell} + k \cdot 2^{d_c+1}]$ and the indices $i_1, \ldots, i_c$ are contained within the interval $[\boldsymbol{\ell} - k \cdot 2^{d_c+1}, \boldsymbol{\ell}]$. As a result, if we denote by $\delta_d(\boldsymbol{\ell}) \in [0, 1]$, for each $d \in [\eta]$, the *density* of k-tuples from T_c lying inside $[\boldsymbol{\ell} - k \cdot 2^{d+1}, \boldsymbol{\ell} + k \cdot 2^{d+1}]$ (i.e., the fraction of this interval comprised of elements of T_c), we have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\boldsymbol{\ell} \sim [n]}\Big[\sum_{d \in [\eta]} \delta_d(\boldsymbol{\ell})\Big] \\
&\quad = \sum_{d \in [\eta]} \sum_{\substack{(i_1, \ldots, i_k) \in T_c \\ \mathsf{dist\text{-}prof}(i_1, \ldots, i_k)_c = d}} \frac{\Pr_{\boldsymbol{\ell} \sim [n]}[i_c \leq \boldsymbol{\ell} \leq i_{c+1}]}{2 \cdot k \cdot 2^{d+1}} \\
&\quad \gtrsim \frac{|T_c|}{n} \gtrsim \varepsilon.
\end{aligned} \tag{2}$$

For any ℓ achieving the above inequality, since $\eta = O(\log n)$, there exists some $d^* \in [\eta]$ such that $\delta_{d^*}(\ell) \gtrsim \varepsilon / \log n$. Consider now the set of k-tuples $T_{c,d^*}(\ell) \subseteq T_c$ contributing to $\delta_{d^*}(\ell)$, i.e., those k-tuples in T_c which are cut by ℓ and lie in $[\ell - k \cdot 2^{d^*+1}, \ell + k \cdot 2^{d^*+1}]$. Let $r_{\mathrm{med}} = \mathsf{median}\{f(i_c) : (i_1, \ldots, i_k) \in T_{c,d^*}(\ell)\}$, and

$$\begin{aligned}
T_L &= \{(i_1, \ldots, i_c) : (i_1, \ldots, i_k) \in T_{c,d^*}(\ell),\, f(i_c) \leq r_{\mathrm{med}}\}, \\
T_R &= \{(i_{c+1}, \ldots, i_k) : (i_1, \ldots, i_k) \in T_{c,d^*}(\ell),\, f(i_c) \geq r_{\mathrm{med}}\},
\end{aligned}$$

where we note that T_L and T_R both have size at least $|T_{c,d^*}(\ell)|/2$. If the algorithm finds a c-tuple in T_L and a $(k-c)$-tuple in T_R, by the observation made in [1] that was mentioned above, the algorithm could combine the tuples to form a length-k monotone subsequence of f. At a high level, one may then recursively apply these considerations on $[\ell - k \cdot 2^{d^*+1}, \ell]$ with T_L and $[\ell, \ell + k \cdot 2^{d^*+1}]$ with T_R. A natural algorithm then mimics the above reasoning algorithmically, i.e., samples a parameter $\boldsymbol{\ell} \sim [n]$, and tries to find the unknown parameter $d^* \in [\eta]$ in order to recurse on both the left and right sides; once the tuples have length 1, the algorithm samples within the interval to find an element of T_L or T_R. This is, in essence, what the algorithm from [1] does, and this approach leads to a query complexity of $(\log n)^{O(k^2)}$. In particular, suppose that at each (recursive) iteration, the parameter c, corresponding to the gap of tuples in T, always equals 1. Note that this occurs when all $(12 \ldots k)$-patterns $(i_1, \ldots, i_k)$ in T have $\mathsf{dist\text{-}prof}(i_1, \ldots, i_k) = (d_1, \ldots, d_{k-1})$ with

$$d_1 \geq d_2 \geq \cdots \geq d_{k-1}. \tag{3}$$

Then, if k is at k_0, a recursive call leads to a set T_L containing 1-tuples, and T_R containing $(k_0 - 1)$-tuples. This only decreases the length of the subsequences needed to be found by 1 (so there will be $k - 1$ recursive calls), while the algorithm pays for guessing the correct value of d^* out of $\Omega(\log n)$ choices, which may decrease the density of monotone k_0-subsequences within the interval of the recursive call by a factor as big as $\Omega(\log n)$.[5] As a result, the density of the length-k_0 monotone subsequence in the relevant interval could be as low as $\varepsilon / (\log n)^{k_0}$, which means that $(\log n)^{\Omega(k_0)}$ samples will be needed for the k_0-th round according to the above analysis, giving a total of $(\log n)^{\Omega(k^2)}$ samples (as opposed to $O((\log n)^{\lfloor \log_2 k \rfloor})$, which is the correct number, as we prove).

In order to overcome the above difficulty, we consider a particular choice of a family T of length-k monotone subsequences given by the "greedy" procedure (see Figure 4). Loosely speaking, this procedure begins with $T = \emptyset$ and iterates through each index $i_1 \in [n] \setminus T$. Each time, if (i_1) can be extended to a length-k monotone subsequence (otherwise it continues to the next available index), the procedure sets i_2 to be the first index, after i_1 and not already in T, such that (i_1, i_2) can be extended to a length-k monotone subsequence; then, it finds an index i_3 which is the next first index after i_2 and not in T such that (i_1, i_2, i_3) can be extended; and so on, until it has obtained a length-k monotone subsequence starting at i_1. It then adds the subsequence as a tuple to T, and repeats. This procedure eventually outputs a set T of disjoint, length-k monotone subsequences of f which has size at least $\varepsilon n / k^2$, and satisfies another crucial "interleaving" property (see Lemma II.1):

> ($\star$) If $(i_1, \ldots, i_k)$ and $(j_1, \ldots, j_k)$ are k-patterns from T and $c \in [k-1]$ satisfy $j_1 < i_1$, $j_c < i_c$, and $i_{c+1} < j_{c+1}$, then $f(i_{c+1}) < f(j_{c+1})$.

Moreover, a slight variant of (1) guarantees that for any $(i_1, \ldots, i_k) \in T_c$ with $\mathsf{dist\text{-}prof}(i_1, \ldots, i_k) = (d_1, \ldots, d_{k-1})$,

$$\Pr_{\boldsymbol{\ell} \sim [n]}\left[i_c + 2^{d_c}/3 \leq \boldsymbol{\ell} \leq i_{c+1} - 2^{d_c}/3\right] \gtrsim \frac{2^{d_c}}{n}.$$

Whenever the above event occurs, we say $\boldsymbol{\ell} \sim [n]$ *cuts* $(i_1, \ldots, i_k)$ at c *with slack*, and note that $i_1, \ldots, i_c$ lie in $[\boldsymbol{\ell} - k \cdot 2^{d_c+1}, \boldsymbol{\ell}]$ and $i_{c+1}, \ldots, i_k$ in $[\boldsymbol{\ell}, \boldsymbol{\ell} + k \cdot 2^{d_c+1}]$. We denote, similarly to the above, $\delta_d(\boldsymbol{\ell}) \in [0, 1]$ to be the density of k-tuples from T_c which are cut with slack by $\boldsymbol{\ell}$, and conclude (2). We then utilize ($\star$) to make the following claim: suppose two k-tuples $(i_1, \ldots, i_k), (j_1, \ldots, j_k) \in T_c$ satisfy $\mathsf{dist\text{-}prof}(i_1, \ldots, i_k) = (d_1, \ldots, d_{k-1})$, and $\mathsf{dist\text{-}prof}(j_1, \ldots, j_k) = (d'_1, \ldots, d'_{k-1})$, where $d_c \leq d'_c - a \log k$, for some constant a which is not too small. If $(i_1, \ldots, i_k)$ and $(j_1, \ldots, j_k)$ are cut at c with slack, this

[5] Initially, the density of T within $[n]$ is ε, and the density of T_L or T_R in $[\ell - k \cdot 2^{d^*+1}, \ell]$ and $[\ell, \ell + k \cdot 2^{d^*+1}]$ is $\varepsilon / \log n$.

means that ℓ lies roughly in the middle of i_c and i_{c+1} and of j_c and j_{c+1}, and since the distance between i_c and i_{c+1} is much smaller than that between j_c and j_{c+1}, the index j_1 will come before i_1, the index j_c will come before i_c, but the index i_{c+1} will come before j_{c+1}. By ($\star$), $f(i_{c+1}) < f(j_{c+1})$ (cf. Lemma II.11). In other words, the value, under the function f, of $(c+1)$-th indices from tuples in $T_{c,d}(\ell)$ increases as d increases.

As a result, if $\ell \in [n]$ satisfies $\sum_{d\in[\eta]} \delta_d(\ell) \gtrsim \varepsilon$, and $\delta_d(\ell) \ll \varepsilon$ for all $d \in [\eta]$, that is, if the summands in (2) are *spread out*, an algorithm could find a length-k monotone subsequence by sampling, for many values of $d \in [\eta]$, indices which appear as the $(c+1)$-th index of tuples in $T_{c,d}(\ell)$. We call such values of ℓ the starts of *growing suffixes* (as illustrated in Figure 5). In Section III-B, we describe an algorithm that makes $\tilde{O}(\log n/\varepsilon)$ queries and finds, with high probability, a length-k monotone subsequence if there are many such growing suffixes (see Lemma III.1). The algorithm works by randomly sampling $\boldsymbol{\ell} \sim [n]$ and hoping that $\boldsymbol{\ell}$ is the start of a growing suffix; if it is, the algorithm samples enough indices from the segments $[\ell + 2^d, \ell + 2^{d+1}]$ to find a $(c+1)$-th index of some tuple in $T_{c,d}(\ell)$, which gives a length-k monotone subsequence.

The other case corresponds to the scenario where $\ell \in [s]$ satisfies $\sum_{d\in[\eta]} \delta_d(\ell) \gtrsim \varepsilon$, but the summands are *concentrated* on few values of $d \in [\eta]$. In this case, we may consider a value of $d^* \in [\eta]$ which has $\delta_{d^*}(\ell) \gtrsim \varepsilon$, and then look at the intervals $[\ell - k \cdot 2^{d^*+1}, \ell]$ and $[\ell, \ell + k \cdot 2^{d^*+1}]$. We can still define T_L and T_R, both of which have size at least $|T_{c,d^*}|/2$ and have the property that any c-tuple from T_L can be combined with any $(k-c)$-tuple from T_R. Additionally, since $\delta_{d^*}(\ell) \gtrsim \varepsilon$, we crucially do *not* suffer a loss in the density of T_L and T_R in their corresponding intervals – a key improvement over the $\Omega(\log n)$ loss in density incurred by the original approach we first discussed. We refer to these intervals as *splittable intervals* (cf. Figure 6), and observe that they lead to a natural recursive application of these insights to the intervals $[\ell - k \cdot 2^{d^*+1}, \ell]$ and $[\ell, \ell + k \cdot 2^{d^*+1}]$. The main structural result, given in Theorem II.3, does exactly this, and encodes the outcomes of the splittable intervals in an object we term a *k-tree descriptor* (see Section II-C) whenever there are not too many growing suffixes. Intuitively, a k-tree descriptor consists of a rooted binary tree G on k leaves, as well as some additional information, which corresponds to a function $f\colon [n] \to \mathbb{R}$ without many growing suffixes. Each internal node v in G corresponds to a recursive application of the above insights, i.e., v has k_0 leaves in its subtree, a parameter $c_v \in [k_0 - 1]$ encoding the gap of sufficiently many k_0-tuples, and a collection of disjoint intervals of the form $[\ell - k \cdot 2^{d^*}, \ell + k \cdot 2^{d^*}]$ where ℓ cuts $(12\ldots k_0)$-patterns with slack at c_v and satisfies (2); the left child of v has c leaves and contains the $(12\ldots c)$-patterns in T_L and intervals $[\ell - k \cdot 2^{d^*}, \ell]$; the right child of v has $k_0 - c$ leaves and contains the $(12\ldots(k_0-c))$-patterns in T_R and intervals $[\ell, \ell + k \cdot 2^{d^*}]$ (see Figure 7).

The algorithm for this case is more involved than the previous, and leads to the $O((\log n)^{\lfloor \log_2 k \rfloor})$-query complexity stated in Theorem I.1. The algorithm proceeds in $r_0 = 1 + \lfloor \log_2 k \rfloor$ rounds, maintaining a set $\mathbf{A} \subseteq [n]$, initially empty:

- *Round 1*: For each $i \in [n]$, include i in $\mathbf{A}$ independently with probability $\Theta(1/(\varepsilon n))$.
- *Round r*, $2 \le r \le r_0$: For each $i \in \mathbf{A}$ from the previous round, and each $j = 1, \ldots, O(\log n)$, consider the interval $B_{i,j} = [i - 2^j, i + 2^j]$. For each $i' \in B_{i,j}$, include i' in $\mathbf{A}$ independently with probability $\Theta(1/(\varepsilon 2^j))$.

At the end of all rounds, the algorithm queries f at all indices in $\mathbf{A}$, and outputs a $(12\ldots k)$-pattern from $\mathbf{A}$, if one exists.

Recall the case considered in the sketch of the algorithm of [1], when the function f has all $(12\ldots k)$-patterns $(i_1, \ldots, i_k)$ in T satisfying $\textsf{dist-prof}(i_1, \ldots, i_k) = (d_1, \ldots, d_{k-1})$ with $d_1 \ge d_2 \ge \ldots \ge d_{k-1}$. In this case, the k-tree descriptor G consists of a rooted binary tree of depth k. The root has a left child which is a leaf (corresponding to 1-tuples of first indices of some tuples in T, stored in T_L) and a right child (corresponding to suffixes of length $(k-1)$ of some tuples in T, stored in T_R) is an internal node. The root node corresponds to one application of the structural result, and the right child corresponds to a $(k-1)$-tree descriptor for the tuples in T_R. Loosely speaking, as $d_2 \ge \ldots \ge d_{k-1}$ the same reasoning repeats $k-1$ times, and leads to a path of length $k-1$ down the right children of the tree, the right child of the $(k-1)$-th internal node corresponding to a 1-tuple (i.e., a leaf).[6]

To gain some intuition, we analyze how the algorithm behaves on these instances. Suppose that in round 1, the algorithm samples an element $i \in [n]$ which is the k-th index of a 1-tuple stored in the right-most leaf of G. In particular, this index belongs to the set T_R of the $(k-1)$-th internal node, as a second index of a cut (12)-pattern in the $(k-1)$-th recursive call of the structural result. Similarly, i also belongs to that set T_R of the $(k-2)$-th internal node, as a part the third index of a cut (123)-pattern in the $(k-2)$-th recursive call. We may continue with all these inclusions to the root, i.e., i is the k-th element of some $(12\ldots k)$-pattern in T, which is cut in the first call to the structural result. Round 2 of the algorithm will consider the $k-1$ intervals $B_{i,d'_{k-1}}, B_{i,d'_{k-2}}, \ldots, B_{i,d'_1}$, where $d'_j = d_j + \Theta(\log k)$, since it iterates through all $O(\log n)$ intervals of geometrically increasing lengths.[7] One can check that for each $j \in [k-1]$, the interval B_{i,d'_j} contains $[\ell_j - k \cdot 2^{d_j}, \ell_j]$, where ℓ_j is some index which cut the $(k-j+1)$-tuple $(i_j, \ldots, i_k)$ with slack in

[6] This is somewhat inaccurate, as in each step, after forming T_L and T_R, we apply the greedy algorithm again and obtain new sets T'_L and T'_R, which may violate the assumption $d_1 \ge d_2 \ge \ldots \ge d_k$. We ignore this detail at the moment to simplify the explanation.

[7] Note that the intervals B_{i,d'_j} and $B_{i,d'_{j+1}}$ may be the same, for instance when $d_j = d_{j+1}$.

the j-th recursive call of the structural result. Recall that the set T_L of 1-tuples has density $\Omega(\varepsilon)$ inside $[\ell_j - k\cdot 2^{d_j}, \ell_j]$ and may be combined with any $(k-j)$-tuple from T_R. Following this argument, in the *second* round of the algorithm, $\mathbf{A}$ will include some index of T_L (for each $j \in [k-1]$), and these indices combine to form a $(12\dots k)$-pattern – that is, with high probability, after two rounds, the algorithm succeeds in finding a monotone subsequence of length k.

Generalizing the above intuition for all possible distance profiles necessitates the use of $1 + \lfloor \log_2 k \rfloor$ rounds, and requires extra care. At a high level, consider an arbitrary k-tree descriptor G for $\Omega(\varepsilon n)$ many $(12\dots k)$-patterns in f. Denote the root u, and consider the unique leaf w of G where the root-to-w path $(u_1, \dots, u_h)$ with $u_1 = u$ and $u_h = w$, satisfies that at each internal node u_l, the next node u_{l+1} is the child with larger number of leaves in its subtree.[8] We call such a leaf a *primary index* of G. The crucial property of the primary index is that the root-to-leaf path of w, $(u_1, u_2, \dots u_h)$, is such that the siblings of the nodes on this path[9] have strictly fewer than $k/2$ leaves in their subtrees.

The relevant event in the first round of the algorithm is that of sampling an index $i \in [n]$ which belongs to a 1-tuple of the primary index w of G. This occurs with probability at least $1 - 1/(100k)$, since we sample each element of $[n]$ with probability $\Theta(1/(\varepsilon n))$ while there are at least $\Omega(\varepsilon n)$ many $(12\dots k)$-patterns. Now, roughly speaking, letting $(u_1, \dots, u_h)$ be the root-to-w path in G, and $(u'_2, \dots, u'_h)$ be the sibling nodes, the subtrees of G rooted at $u'_2, \dots, u'_h$ will be tree descriptors for the function f restricted to $B_{i,j}$'s and within these interval, the density of tuples is at least $\Omega(\varepsilon)$. As a result, the second round of the algorithm, recursively handles each subtree rooted at $u'_2, \dots, u'_h$ with one fewer round. Since the subtrees have strictly fewer than $k/2$ leaves, $\lfloor \log_2 k \rfloor - 1$ rounds are enough for an inductive argument. Moreover, since the total number of nodes is at most $2k$ and each recursive call succeeds with probability at least $1 - 1/(100k)$, by a union bound we may assume that all recursive calls succeed.

Unrolling the recursion, the query complexity $\Theta((\log n)^{\lfloor \log_2 k \rfloor})$ can be explained with a simple combinatorial game. We start with a rooted binary tree G on k leaves. In one round, whenever G is not simply a leaf, we pick the leaf w which is the primary index of G, and replace G with a collection of subtrees obtained by cutting out the root-to-w path in G. These rounds "pay" a factor of $\Theta(\log n)$, since the algorithm must find intervals on which the collection of subtrees form tree descriptors of f (restricted to these intervals). In the subsequent rounds, we recurse on each subtree simultaneously, picking the leaf of the primary index in each, and so on. After $\lfloor \log_2 k \rfloor$ many rounds, the trees are merely leaves, and the algorithm does not need to pay the factor $\Theta(\log n)$ to find good intervals, as it may simply sample from these intervals.

The execution of the above high-level plan is done in Section III-C, where Lemma III.2 is the main inductive lemma containing the analysis of the main algorithm (shown in Figure 11 and Figure 12).

C. Our techniques: Lower bound

In order to highlight the main ideas behind the proof of Theorem I.2 (the lower bound on the query complexity), we first cover the simpler case of $k = 2$. This case corresponds to a lower bound of $\Omega(\log n)$ on the number of queries needed for non-adaptive and one-sided algorithms for monotonicity testing. Such a lower bound is known, even for adaptive algorithms with two-sided error [12], [13]. We rederive and present the well-known non-adaptive one-sided lower bound in our language; after that, we generalize it to the significantly more involved case $k > 2$. For the purpose of this introduction, we give an overview assuming that both n and k are powers of 2; as described in Section IV, a simple "padding" argument generalizes the result to all n and k.

For any $n \in \mathbb{N}$ which is a power of 2 and $t \in [n]$, consider the *binary representation* $B_n(t) = (b_1^t, b_2^t, \dots, b_{\log_2 n}^t) \in \{0,1\}^{\log_2 n}$ of t, where $t = b_1^t \cdot 2^0 + b_2^t \cdot 2^1 + \cdots + b_{\log_2 n}^t \cdot 2^{\log_2 n - 1}$. For $i \in [\log_2 n]$, the *bit-flip operator*, $F_i \colon [n] \to [n]$, takes an input $t \in [n]$ with binary representation $B_n(t)$ and outputs the number $F_i(t) = t' \in [n]$ with binary representation obtained by flipping the i-th bit of $B_n(t)$. Finally, for any two distinct elements $x, y \in [n]$, let $M(x,y) \in [\log_2 n]$ be the index of the most significant bit in which they differ, i.e., the largest i where $b_i^x \neq b_i^y$.

As usual for lower bounds on randomized algorithms, we rely on Yao's minimax principle [40]. In particular, our lower bounds proceed by defining, for each n and k (which are powers of 2), a distribution $\mathcal{D}_{n,k}$ supported on functions $f \colon [n] \to \mathbb{R}$ which are all ε-far from $(12\dots k)$-free. We show that any deterministic and non-adaptive algorithm which makes fewer than q queries, where $q = c_k(\log_2 n)^{\log_2 k}$ and $c_k > 0$ depends only on k, fails to find a $(12\dots k)$-pattern in a random $\boldsymbol{f} \sim \mathcal{D}_{n,k}$, with probability at least $1/10$. Note that any deterministic, non-adaptive algorithm which makes fewer than q queries is equivalently specified by a set $Q \subseteq [n]$ with $|Q| < q$. Thus, the task of the lower bound is to design a distribution $\mathcal{D}_{n,k}$ supported on functions $f \colon [n] \to \mathbb{R}$, each of which is ε-far from $(12\dots k)$-free, such that for any $Q \subseteq [n]$ with $|Q| < c_k(\log_2 n)^{\log_2 k}$ the following holds

$$\Pr_{\boldsymbol{f} \sim \mathcal{D}_{n,k}}\left[\exists i_1, \dots, i_k \in Q : {}^{i_1 < \cdots < i_k \text{ and}}_{\boldsymbol{f}(i_1) < \cdots < \boldsymbol{f}(i_k)}\right] \leq \frac{9}{10}. \tag{4}$$

[8] Ties are broken by picking the left child.

[9] For example, if $(u_1, \dots, u_h)$ is the root-to-w path where u_1 is the root and $u_h = w$, the sibling nodes along the path are given by $u'_2, u'_3, \dots, u'_h$, where u'_l is the sibling of u_l. Namely, if the l-th node on the root-to-w path is a left child of the $(l-1)$th node, then u'_l is the right child of the $(l-1)$-th node. Analogously, if the l-th node is a right child of the $(l-1)$-th node, then u'_l is the left child of the $(l-1)$-th node.

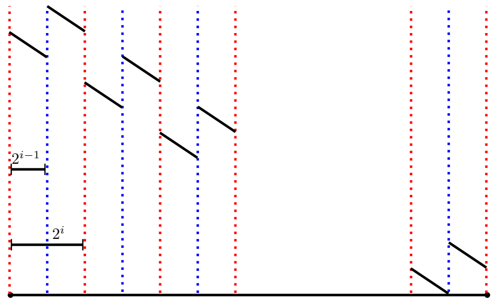

Figure 1. Example of a function f_i lying in the support of $\mathcal{D}_{n,2}$. One may view the domain as being divided into intervals of length 2^i (displayed as intervals lying between dotted red lines) and the permutation $f^{\downarrow}$ flipped across adjacent intervals of length 2^{i-1} inside a segment of length 2^i (displayed as intervals lying between dotted blue lines). Note that all (12)-patterns in f_i above have the ith bit flipped.

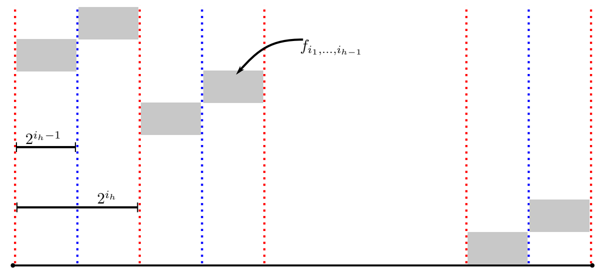

Figure 2. Example of a function $f_{i_1,\ldots,i_h}$ lying in the support of $\mathcal{D}_{n,k}$, where $k = 2^h$. Similarly to the case of $k = 2$ (shown in Figure 1), the domain is divided into intervals of length 2^{i_h} (shown between red dotted lines), and functions are flipped across adjacent intervals of length 2^{i_h-1} within an interval of length 2^{i_h} (shown between blue dotted lines). Inside each grey region is a recursive application of the construction, $f_{i_1,\ldots,i_{h-1}}$, after shifting the range.

LOWER BOUND FOR $k = 2$ (MONOTONICITY). The case of $k = 2$ relies on the following idea: for any $i \in [\log_2 n]$, one can construct a function (in fact, a permutation) $f_i \colon [n] \to [n]$ which is 1/2-far from (12)-free, and furthermore, all pairs of distinct elements $(x, y) \in [n]^2$ where $x < y$ and $f_i(x) < f_i(y)$ satisfy $M(x, y) = i$. One can construct such a function $f_i \colon [n] \to [n]$, for any $i \in [\log_2 n]$ in the following way. First, let $f^{\downarrow} \colon [n] \to [n]$ be the decreasing permutation, $f^{\downarrow}(x) = n+1-x$ for any $x \in [n]$. Now take f_i to be $f^{\downarrow} \circ F_i$, where $\circ$ denotes function composition, that is, $f_i(x) = f^{\downarrow}(F_i(x))$ for any $x \in [n]$. Finally, set $\mathcal{D}_{n,2}$ to be the uniform distribution over the functions $f_1, f_2, \ldots, f_{\log n}$ (see Figure 1).

Towards proving (4) for the distribution $\mathcal{D}_{n,2}$, we introduce the notion of *binary profiles* captured by a set of queries. For any fixed $Q \subseteq [n]$, the binary profiles captured in Q are given by the set

$$\textsf{bin-prof}(Q) = \{i \in [\log_2 n] \,:\, \exists x, y \in Q \text{ s.t. } M(x, y) = i\}.$$

Since all (12)-patterns (x, y) of f_i have $M(x, y) = i$, the probability over $\boldsymbol{f} \sim \mathcal{D}_{n,2}$ that an algorithm whose set of queries is Q, finds a (12)-pattern in $\boldsymbol{f}$ is at most $|\textsf{bin-prof}(Q)|/\log_2 n$. We show that for any set $Q \subseteq [n]$, $|\textsf{bin-prof}(Q)| \leq |Q| - 1$. This completes (4), and proves the lower bound of $\frac{9}{10}\log_2 n$ for $k = 2$.

The proof that $|\textsf{bin-prof}(Q)| \leq |Q| - 1$ for any set $Q \subset [n]$, i.e., the number of *captured profiles* is bounded by the number of queries, follows by induction on $|Q|$. The base case $|Q| \leq 2$ is trivial. When $|Q| > 2$, let $i_{\max} = \max \textsf{bin-prof}(Q)$. Consider the partition of Q into Q_0 and Q_1, where

$$Q_0 = \{x \in Q : b^x_{i_{\max}} = 0\} \text{ and } Q_1 = \{y \in Q : b^y_{i_{\max}} = 1\}.$$

Since $\textsf{bin-prof}(Q) = \textsf{bin-prof}(Q_0) \cup \textsf{bin-prof}(Q_1) \cup \{i_{\max}\}$, we conclude that $|\textsf{bin-prof}(Q)| \leq |\textsf{bin-prof}(Q_0)| + |\textsf{bin-prof}(Q_1)| + 1 \leq |Q_0| - 1 + |Q_1| - 1 + 1 = |Q| - 1$, where the second inequality follows from the inductive hypothesis.

GENERALIZATION TO $k > 2$: PROOF OF THEOREM I.2. We now provide a detailed sketch of the proof of Theorem I.2. The main objects and notions used are defined, while leaving technical details to Section IV. Let $k = 2^h$ for $h \in \mathbb{N}$; the case $h = 1$ corresponds to the previous discussion.

We first define the distributions $\mathcal{D}_{n,k}$ supported on permutations $f \colon [n] \to [n]$ which are $\Omega(1/k)$-far from $(12\ldots k)$-free. Recall that the function f_i in the case $k = 2$ was constructed by "flipping" bit i in the representation of $f^{\downarrow}$, that is, $f_i = f^{\downarrow} \circ F_i$. Generalizing this construction, for any $i_1 < i_2 < \cdots < i_h \in [\log_2 n]$ we let $f_{i_1,\ldots,i_h} \colon [n] \to [n]$ denote the result of flipping bits $i_1, i_2, \ldots, i_h$ in the representation of $f^{\downarrow}$:

$$f_{i_1,\ldots,i_h} := f^{\downarrow} \circ F_{i_h} \circ \ldots \circ F_{i_1}.$$

It can be shown that $f_{i_1,\ldots,i_h}$ is $(1/k)$-far from $(12\ldots k)$-free (see Figure 2). We take $\mathcal{D}_{n,k}$ as the uniform distribution over all functions of the form $f_{i_1,\ldots,i_h}$, where $i_1 < \cdots < i_h \in [\log_2 n]$.

Towards the proof of (4) for the distribution $\mathcal{D}_{n,k}$, we generalize the notion of a binary profile. Consider any k-tuple of indices $(x_1, \ldots, x_k) \in [n]^k$ satisfying $x_1 < \cdots < x_k$. We say that $(x_1, x_2, \ldots, x_k)$ has *h-profile of type* $(i_1, \ldots, i_h)$ if, for every $j \in [k-1]$, $M(x_j, x_{j+1}) = i_{M(j-1,j)}$.

For instance, when $h = 3$ (i.e. $k = 8$) the tuple $(x_1, \ldots, x_k)$ has h-profile of type (i_1, i_2, i_3) if the sequence $(M(x_j, x_{j+1}))_{j=1}^7$ is $(i_1, i_2, i_1, i_3, i_1, i_2, i_1)$. See Figure 3 for a visual demonstration of a 3-profile.[10]

It can be shown that for any $i_1 < \cdots < i_h \in [\log_2 n]$, the function $f = f_{i_1,\ldots,i_h}$ has the following property. If $x_1 < \cdots < x_k \in [n]$ satisfy $f(x_1) < \cdots < f(x_k)$, i.e., the k-tuple $(x_1, \ldots, x_k)$ is a $(12\ldots k)$-pattern of $f_{i_1,\ldots,i_k}$, then $(x_1, \ldots, x_k)$ has an h-profile of type $(i_1, \ldots, i_h)$. We thus proceed similarly to the case $k = 2$. For any $Q \subseteq [n]$, we

[10] Unlike the case $k = 2$, not all tuples $(x_1, \ldots, x_k)$ with $x_1 < \ldots < x_k$ have an h-profile. For what follows we will only be interested in tuples that *do* have a profile.

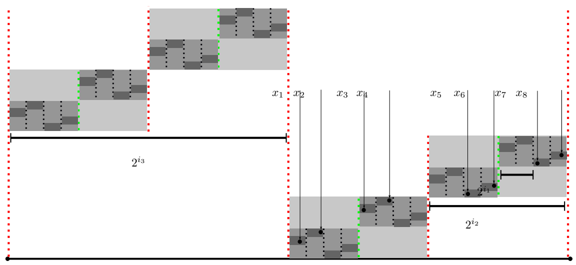

Figure 3. Example of a function f_{i_1,i_2,i_3} in the support of $\mathcal{D}_{n,8}$, with a k-tuple $(x_1,\ldots,x_8)$ whose h-profile has type (i_1,i_2,i_3).

define the set of all h-profiles captured by Q as follows

$$\textsf{bin-prof}_h(Q)=\left\{(i_1,\ldots,i_h):\ \substack{\exists x_1,\ldots,x_k\in Q \text{ s.t. } x_1<\cdots<x_k\\ \text{and } (x_1,\ldots,x_k) \text{ has } h\text{-profile}\\ \text{of type } (i_1,\ldots,i_h)}\right\}.$$

The proof that $|\textsf{bin-prof}_h(Q)|\le |Q|-1$ for any $Q\subseteq[n]$ follows by induction on h. The base case $h=1$ was covered in the discussion on $k=2$. For $h>1$, we define subsets

$$\emptyset=B_{\log_2 n+1}\subseteq B_{\log_2 n}\subseteq\ldots\subseteq B_1=Q,$$

where, given B_{i+1}, the set $B_i\supseteq B_{i+1}$ is an arbitrary maximal subset of Q containing B_{i+1}, so that no two elements $x\neq y\in B_i$ satisfy $M(x,y)<i$. Additionally, for each $j\in[\log_2 n]$ we let

$$N_j=\left\{(i_2,\ldots,i_h):\ \substack{1\le j<i_2\cdots<i_h\le\log_2 n \text{ and}\\ (j,i_2,\ldots,i_h)\in\textsf{bin-prof}_h(Q)}\right\}.$$

The key observation is that $N_j\subseteq\textsf{bin-prof}_{h-1}(B_j\setminus B_{j+1})$. To see this, note first that any $(j,i_2,\ldots,i_h)\in\textsf{bin-prof}_h(Q)$ also satisfies $(j,i_2,\ldots,i_h)\in\textsf{bin-prof}_h(B_j)$. Indeed, suppose that a tuple $(x_1,\ldots,x_k)$ with $x_1<\cdots<x_k\in Q$ has h-profile $(j,i_2,\ldots,i_h)$. By the maximality of B_j, we know that for every $1\le\ell\le k$ there exists $y_\ell\in B_j$ such that either $x_\ell=y_\ell$ or $M(x_\ell,y_\ell)<j$. This implies that $\{y_1,\ldots,y_k\}\subseteq B_j$ has h-profile $(j,i_2,\ldots,i_h)$.

Now, suppose that $y_1,\ldots,y_k\in B_j$ satisfies $y_1<\ldots<y_k$ and has h-profile of type $(j,i_2,\ldots,i_h)$ in B_j. For any $1\le t\le k/2$ we have $M(y_{2t-1},y_{2t})=j$. Therefore, at most one of y_{2t-1},y_{2t} is in B_{j+1}, and hence, for any such t there exists $z_t\in\{y_{2t-1},y_{2t}\}\setminus B_{j+1}\subseteq B_j\setminus B_{j+1}$. It follows that $(z_1,\ldots,z_{k/2})\in B_j\setminus B_{j+1}$ has $(h-1)$-profile $(i_2,\ldots,i_h)$. This concludes the proof that $N_j\subseteq\textsf{bin-prof}_{h-1}(B_j\setminus B_{j+1})$.

We now use the last observation to prove that $|\textsf{bin-prof}_h(Q)|\le|Q|-1$. Note that

$$\textsf{bin-prof}_h(Q)=\bigcup_{j=1}^{\log_2 n}\{(j,i_2,\ldots,i_h):(i_2,\ldots,i_h)\in N_j\}$$

and $Q=\bigcup_{j=1}^{\log_2 n}(B_j\setminus B_{j+1})$, where both unions are disjoint unions. By the induction assumption, $|N_j|\le|\textsf{bin-prof}_{h-1}(B_j\setminus B_{j+1})|<|B_j\setminus B_{j+1}|$ for any j if N_j is non-empty; If N_j is empty, then $|N_j|\le|\textsf{bin-prof}_{h-1}(B_j\setminus B_{j+1})|\le|B_j\setminus B_{j+1}|$ trivially holds. Hence

$$|Q|=\sum_{j=1}^{\log_2 n}|B_j\setminus B_{j+1}|>\sum_{j=1}^{\log_2 n}|N_j|=|\textsf{bin-prof}_h(Q)|,$$

where the strict inequality follows because if $\textsf{bin-prof}_h(Q)$ is non-empty then N_j is non-empty for some j. This completes the proof.

D. Organization

We start by introducing the notation that we shall use throughout the paper in Section I-E. In Section II we prove our main structural result, and formally define the notions that underlie it: namely, Theorem II.3, along with the definitions of growing suffixes and representation by tree descriptors (Definitions II.4 and II.7). Section III then leverages this dichotomy to describe and analyze our testing algorithm, thus establishing the upper bound of Theorem I.1 (see Theorem III.1 for a formal statement). Finally, we complement this algorithm with a matching lower bound in Section IV, where we prove Theorem I.2.

While Section III crucially relies on Section II, these two sections are independent of Section IV, which is mostly self-contained.

E. Notation and Preliminaries

We write $a\lesssim b$ if there exists a universal positive constant $C>0$ such that $a\le Cb$, and $a\asymp b$ if we have both $a\lesssim b$ and $b\lesssim a$. At times, we write $\mathrm{poly}(k)$ to stand for $O(k^C)$, where $C>0$ is a large enough universal constant. Unless otherwise stated, all logarithms will be in base 2. We frequently denote $\mathcal{I}$ as a collection of disjoint intervals, $I_1,\ldots,I_s$, and then write $\mathcal{S}(\mathcal{I})$ for the set of all sub-intervals which lie within some interval in $\mathcal{I}$. For two collections of disjoint intervals $\mathcal{I}_0$ and $\mathcal{I}_1$, we say that $\mathcal{I}_1$ is a *refinement* of $\mathcal{I}_0$ if every interval in $\mathcal{I}_1$ is contained within an interval in $\mathcal{I}_0$. (We remark that it is not the case that intervals in $\mathcal{I}_1$ must form a partition of intervals in $\mathcal{I}_0$.) For a particular set $A\subseteq[n]$ and an interval $I\subseteq[n]$, we define the *density* of A in I as the ratio $|A\cap I|/|I|$. Given a set S, we write $\boldsymbol{x}\sim S$ to indicate that $\boldsymbol{x}$ is a random variable given by a sample drawn uniformly at random from S, and $\mathcal{P}(S)$ for the power set of S. Given a sequence f of length n, we shall interchangeably use the notions $(12\ldots k)$-*copy*, $(12\ldots k)$-*pattern*, and *length-k increasing subsequence*, to refer to a tuple $(x_1,\ldots,x_k)\in[n]^k$ such that $x_1<\ldots<x_k$ and $f(x_1)<\ldots<f(x_k)$.

II. Structural Result

A. Rematching procedure

Let $f\colon[n]\to\mathbb{R}$ be a function which is ε-far from $(12\ldots k)$-free. Let T be a set of k-tuples representing monotone subsequences of length k within f, i.e.,

$$T\subseteq\left\{(i_1,\ldots,i_k)\in[n]^k:\ \substack{i_1<\cdots<i_k \text{ and}\\ f(i_1)<\cdots<f(i_k)}\right\},$$

and for such T let $E(T)$ be the set of indices of subsequences in T, so

$$E(T) = \bigcup_{(i_1,\ldots,i_k)\in T} \{i_1,\ldots,i_k\}.$$

Observation II.1. If $f\colon [n] \to \mathbb{R}$ is ε-far from $(12\ldots k)$-free, then there exists a set $T \subseteq [n]^k$ of disjoint length-k increasing subsequences of f such that $|T| \geq \varepsilon n/k$.

To see why the observation holds, take T to be a maximal disjoint set of such k-tuples. Then we can obtain a $(12\ldots k)$-free sequence from f by changing only the entries of $E(T)$ (e.g. for every $i \in E(T)$ define $f(i) = f(j)$ where j is the largest $[n] \setminus E(T)$ which is smaller than i. If there is no $j \in [n]\setminus E(T)$ where $j < i$, let $f(i) = \max_{\ell\in[n]} f(\ell)$). Since f is ε-far from being $(12\ldots k)$-free, we have $|E(T)| \geq \varepsilon n$, thus $|T| \geq \varepsilon n/k$.

In this section, we show that from a function $f\colon [n] \to \mathbb{R}$ which is ε-far from $(12\ldots k)$-free and a set T_0 of disjoint, length-k monotone subsequences of f, a greedy rematching algorithm finds a set T of disjoint, length-k monotone subsequences of f where $E(T) \subseteq E(T_0)$ with some additional structure, which will later be exploited in the structural lemma and the algorithm. The greedy rematching algorithm, `GreedyDisjointTuples`, is specified in Figure 4; for convenience, in view of its later use in the algorithm, we phrase it in terms of an arbitrary parameter k_0, not necessarily the (fixed) parameter k itself.

Lemma II.1. *Let $k_0 \in \mathbb{N}$, $f\colon [n] \to \mathbb{R}$, and let $T_0 \subseteq [n]^{k_0}$ be a set of disjoint monotone subsequences of f of length k_0. Then there exists a set $T \subseteq [n]^{k_0}$ of disjoint k_0-tuples with $E(T) \subseteq E(T_0)$ such that the following holds.*

1) *The set T holds disjoint monotone subsequences of length k_0.*
2) *The size of T satisfies $|T| \geq |T_0|/k_0$.*
3) *For any two $(i_1,\ldots,i_{k_0}),(j_1,\ldots,j_{k_0}) \in T$ and any $\ell \in [k_0-1]$, if $i_1 < j_1$, $i_\ell < j_\ell$ and $i_{\ell+1} > j_{\ell+1}$ then $f(i_{\ell+1}) > f(j_{\ell+1})$.*

Proof of Lemma II.1: We show that the subroutine `GreedyDisjointTuples`(f, k_0, T_0), described in Figure 4, finds a set T with $E(T) \subseteq E(T_0)$ satisfying properties 1, 2, and 3. Property 1 is clear from the description of `GreedyDisjointTuples`(f, k_0, T_0). For 2, suppose $|T| < |T_0|/k_0$, then, there exists a tuple $(i_1,\ldots,i_{k_0}) \in T_0$ with $\{i_1,\ldots,i_{k_0}\} \cap E(T) = \emptyset$. Since `GreedyDisjointTuples`$(f, k_0, T_0)$ increases the size of T throughout the execution, $\{i_1,\ldots,i_{k_0}\} \cap T = \emptyset$ at every point in the execution of the algorithm. This is a contradiction; when $i = i_1$, a monotone subsequence disjoint from T would have been found, and i_1 included in T. Finally, for 3, consider the iteration when $i = i_1$, and note that at this moment, $T \cap \{i_1,\ldots,i_{k_0}, j_1,\ldots,j_{k_0}\} = \emptyset$. Suppose that $i_\ell < j_\ell$, $j_{\ell+1} < i_{\ell+1}$; if $f(j_{\ell+1}) \geq f(i_{\ell+1})$, then $(i_1,\ldots,i_\ell, j_{\ell+1},\ldots,j_{k_0})$ is an increasing subsequence in $E(T_0) \setminus E(T)$, which means that $j_{\ell+1}$ would have been preferred over $i_{\ell+1}$, a contradiction. ∎

Subroutine `GreedyDisjointTuples`(f, k_0, T_0)

Input: A function $f\colon [n] \to \mathbb{R}$, integer $k_0 \in \mathbb{N}$, and a set T_0 of disjoint monotone subsequences of f of length k_0.
Output: a set $T \subseteq [n]^{k_0}$ of disjoint monotone subsequences of f of length k_0.

1) Let $T = \emptyset$ and i be the minimum element in $E(T_0)$. Repeat the following.
   i. Let $i_1 \leftarrow i$. If there exists $j_2,\ldots,j_{k_0} \in E(T_0) \setminus E(T)$ such that $(i_1, j_2,\ldots,j_{k_0})$ is an increasing subsequence of f, pick $i_2,\ldots,i_{k_0} \in E(T_0) \setminus E(T)$ recursively as follows: for $\ell = 2,\ldots,k_0$, let i_ℓ be the smallest element in $E(T_0) \setminus E(T)$ for which there exist $j_{\ell+1},\ldots,j_{k_0} \in E(T_0) \setminus E(T)$ such that $(i_1,\ldots,i_\ell, j_{\ell+1},\ldots,j_{k_0})$ is an increasing subsequence of f.
   ii. If $(i_1,\ldots,i_{k_0})$ is a monotone subsequence found by (i), set $T \leftarrow T \cup \{(i_1,\ldots,i_{k_0})\}$.
   iii. Let i be the next element of $E(T_0) \setminus E(T)$, if such an element exists; otherwise, proceed to 2.
2) Output T.

Figure 4. Description of the `GreedyDisjointTuples` subroutine.

Definition II.2 (c-gap)**.** *Let $(i_1,\ldots,i_{k_0})$ be a monotone subsequence of f and let $c \in [k_0-1]$. We say that $(i_1,\ldots,i_{k_0})$ is a c-*gap subsequence *if c is the smallest integer such that $i_{c+1} - i_c \geq i_{b+1} - i_b$ for all $b \in [k_0-1]$.*

Note that for a set T of disjoint length-k_0 monotone subsequences of f, we may partition the k_0-tuples of T into $(T_1,\ldots,T_{k_0-1})$ where for each $c \in [k_0-1]$, T_c holds the c-gap monotone subsequences of T. As these sets form a partition of T, the following lemma is immediate from Lemma II.1.

Lemma II.3. *Let $f\colon [n] \to \mathbb{R}$, and let T_0 be a set of disjoint length-k_0 monotone subsequences of f. Then there exist $c \in [k_0-1]$ and a family $T \subseteq [n]^{k_0}$ of disjoint monotone subsequences of f, with $E(T) \subseteq E(T_0)$ such that the following holds.*

1) *The subsequences in T are all c-gap subsequences.*
2) $|T| \geq |T_0|/k_0^2$.
3) *For any two $(i_1,\ldots,i_{k_0}),(j_1,\ldots,j_{k_0}) \in T$ and any $\ell \in [k_0-1]$, if $i_1 < j_1$, $i_\ell < j_\ell$ and $i_{\ell+1} > j_{\ell+1}$ then $f(i_{\ell+1}) > f(j_{\ell+1})$.*

B. Growing suffixes and splittable intervals

We now proceed to set up notation and prepare for the main structural theorem for sequences $f\colon [n] \to \mathbb{R}$ which are ε-far from $(12\ldots k)$-free. In order to simplify the presentation of the subsequent discussion, consider fixed $k \in \mathbb{N}$ and $\varepsilon \in (0,1)$, as well as a fixed sequence $f\colon [n] \to \mathbb{R}$ which is ε-far from $(12\ldots k)$-free. We will, at times, suppress polynomial factors in k by writing $\mathrm{poly}(k)$ to refer to a large enough polynomial in k, whose degree is a large enough universal constant. By Observation II.1 and Lemma II.3, there exists an integer $c \in [k-1]$ and a set T of disjoint monotone subsequences of f which have a c-gap, satisfying $|T| \geq \varepsilon n/\mathrm{poly}(k)$ and property 3 from Lemma II.3. For the rest of the subsection, we consider a fixed setting of such $c \in [k-1]$ and set T.

We will show (in Theorem II.2) that one of the following two possibilities holds. Either there is a large set of what we call *growing suffixes* (see Definition II.4 for a formal definition), or there are disjoint intervals which we call *splittable* (see Definition II.5 for a formal definition). Intuitively, a growing suffix will be given by the suffix $(a, n]$ and will have the property that by dividing $(a, n]$ into $\Theta(\log_2(n-a))$ segments of geometrically increasing lengths, there are many monotone subsequences $(i_1, \ldots, i_k)$ of f lying inside $(a, n]$ where each i_t belongs to a different segment. In the other case, an interval $[a, b]$ is called splittable if it can be divided into three sub-intervals of roughly equal size, which we refer to as the left, middle, and right intervals, with the following property: the left interval contains a large set T_L of $(12\ldots c)$-patterns, the right interval contains a large set T_R of $(12\ldots(k-c))$-patterns, and combining any $(12\ldots c)$-pattern in T_L with any $(12\ldots(k-c))$-pattern in T_R yields a $(12\ldots k)$-pattern.

For each index $a \in [n]$, let $\eta_a = \lceil \log_2(n-a) \rceil$. Let $S_1(a), \ldots, S_{\eta_a}(a) \subseteq [n]$ be disjoint intervals given by $S_t(a) = [a + 2^{t-1}, a + 2^t) \cap [n]$. The collection of intervals $S(a) = (S_t(a) : t \in [\eta_a])$ partitions the suffix $(a, n]$ into intervals of geometrically increasing lengths (except possibly the last interval, which may be shorter), and we refer to the collection $S(a)$ as the *growing suffix* at a.

Definition II.4. *Let $\alpha, \beta \in [0, 1]$. We say that an index $a \in [n]$ starts an (α, β)*-growing suffix *if, when considering the collection of intervals $S(a) = \{S_t(a) : t \in [\eta_a]\}$, for each $t \in [\eta_a]$ there is a subset $D_t(a) \subseteq S_t(a)$ of indices such that the following properties hold.*

1) *We have $|D_t(a)|/|S_t(a)| \leq \alpha$ for all $t \in [\eta_a]$, and $\sum_{t=1}^{\eta_a} |D_t(a)|/|S_t(a)| \geq \beta$.*
2) *For every $t, t' \in [\eta_a]$ where $t < t'$, if $b \in D_t(a)$ and $b' \in D_{t'}(a)$, then $f(b) < f(b')$.*

Intuitively, our parameter regime will correspond to the case when α is much smaller than β, specifically, $\alpha \leq \beta/\mathrm{poly}(k)$, for a sufficiently large-degree polynomial in k. If $a \in [n]$ starts an (α, β)-growing suffix with these parameters, then the η_a segments, $S_1(a), \ldots, S_{\eta_a}(a)$, contain many monotone subsequences of length k which are algorithmically easy to find (given access to the start a). Indeed, by (2), it suffices to find a k-tuple $(i_1, \ldots, i_k)$ such that $i_1 \in D_{t_1}, \ldots, i_k \in D_{t_k}$, for some $t_1, \ldots, t_k \in [\eta_a]$ with $t_1 < \ldots < t_k$ (see Figure 5). By (1), the sum of densities is at least β, yet each density is less than $\alpha \leq \beta/\mathrm{poly}(k)$.

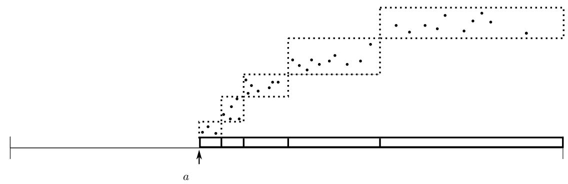

Figure 5. Depiction of an (α, β)-growing suffix at index $a \in [n]$ (see Definition II.4). The labeled segments $S_t(a)$ are shown, as well as the subsets $D_t(a)$. Notice that for all j, all the elements in $D_t(a)$ lie below those in $D_{t+1}(a)$. In Section III-B, we show that if an algorithm knows that a starts an (α, β)-growing suffix, for $\alpha \leq \beta/\mathrm{poly}(k)$, then sampling $\mathrm{poly}(k)/\beta$ many random indices from each $S_t(a)$ finds a monotone pattern with probability at least $9/10$.

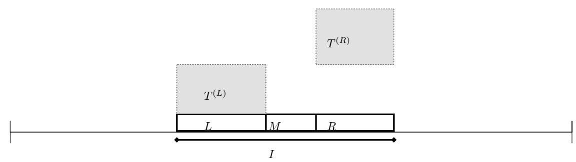

Figure 6. Depiction of a (c, α, β)-splittable interval, as defined in Definition II.5. The interval I is divided into three adjacent intervals, L, M, and R, and the disjoint monotone sequences are divided so that $T^{(L)}$ contains the indices $(i_1, \ldots, i_c)$ and $T^{(R)}$ contains the indices $(i_{c+1}, \ldots, i_k)$. Furthermore, we have that every $(i_1, \ldots, i_c) \in T^{(L)}$ and $(j_{c+1}, \ldots, j_k) \in T^{(R)}$ have $f(i_c) < f(j_{c+1})$, so that any monotone pattern of length c in $E(T^{(L)})$ may be combined with any monotone pattern of length $k-c$ in $E(T^{(R)})$ to obtain a monotone pattern of length k within I.

In other words, the densities of $D_1(a), \ldots, D_{\eta_a}(a)$ within $S_1(a), \ldots, S_{\eta_a}(a)$, respectively, must be spread out, which implies, intuitively, that there are many ways to pick suitable $i_1, \ldots, i_k$.

Definition II.5. *Let $\alpha, \beta \in (0, 1]$ and $c \in [k_0 - 1]$. Let $I \subseteq [n]$ be an interval, let $T \subseteq I^{k_0}$ be a set of disjoint, length-k_0 monotone subsequences of f lying in I, and define*

$$T^{(L)} = \{(i_1, \ldots, i_c) \in I^c : (i_1, \ldots, i_c) \text{ is a prefix of a } k_0\text{-tuple in } T\}, \text{ and}$$
$$T^{(R)} = \{(j_1, \ldots, j_{k_0-c}) \in I^{k_0-c} : (j_1, \ldots, j_{k_0-c}) \text{ is a suffix of a } k_0\text{-tuple in } T\}.$$

We say that the pair (I, T) is (c, α, β)-splittable *if $|T|/|I| \geq \beta$; $f(i_c) < f(j_1)$ for every $(i_1, \ldots, i_c) \in T^{(L)}$ and $(j_1, \ldots, j_{k_0-c}) \in T^{(R)}$; and there is a partition of I into three adjacent intervals $L, M, R \subseteq I$ (that appear in this order, from left to right) of size at least $\alpha|I|$, satisfying $T^{(L)} \subseteq L^c$ and $T^{(R)} \subseteq R^{k_0-c}$.*

A collection of disjoint interval-tuple pairs $(I_1, T_1), \ldots, (I_s, T_s)$ is called a (c, α, β)-splittable collection *of T if each (I_j, T_j) is (c, α, β)-splittable and the sets $(T_j : j \in [s])$ partition T.*

We now state the main theorem of this section, whose proof will be given in Section II-E.

Theorem II.2. *Let $k, k_0 \in \mathbb{N}$ be positive integers satisfying $1 \le k_0 \le k$, and let $\delta \in (0,1)$ and let $C > 0$. Let $f\colon [n] \to \mathbb{R}$ be a function and let $T_0 \subseteq [n]^{k_0}$ be a set of δn disjoint monotone subsequences of f of length k_0. Then there exists an $\alpha \ge \Omega(\delta/k^5)$ such that at least one of the following conditions holds.*

1) *Either there exists a set $H \subseteq [n]$, of indices that start an $(\alpha, Ck\alpha)$-growing suffix, satisfying $\alpha|H| \ge \delta n/\mathrm{poly}(k, \log(1/\delta))$; or*
2) *There exists an integer c with $1 \le c < k_0$, a set T, with $E(T) \subseteq E(T_0)$, of disjoint length-k_0 monotone subsequences, and a $(c, 1/(6k), \alpha)$-splittable collection of T, of disjoint interval-tuple pairs $(I_1, T_1), \dots, (I_s, T_s)$, such that*

$$\alpha \sum_{h=1}^{s} |I_h| \ge \frac{|T_0|}{\mathrm{poly}(k, \log(1/\delta))}.$$

We remark that the above theorem is stated with respect to the two parameters, k_0 and k, for ease of applicability. In particular, in the next section, we will apply Theorem II.2 multiple times, and it will be convenient to have k be fixed and k_0 be a varying parameter. In that sense, even though the monotone subsequences in question have length k_0, the relevant parameters that Theorem II.2 lower bounds only depend on k.

Consider the following scenario: $f\colon [n] \to \mathbb{R}$ is a sequence which is ε-far from $(12\dots k)$-free, so by Observation II.1, there exists a set T_0 of disjoint, length-k monotone subsequences of f of size at least $\varepsilon n/k$. Suppose that upon applying Theorem II.2 with $k_0 = k$ and $\delta = \varepsilon/k$, (2) holds. Then, there exists a $(c, 1/(6k), \alpha)$-splittable collection of a large subset of disjoint, length-k monotone subsequences T into disjoint interval-tuple pairs $(I_1, T_1), \dots, (I_s, T_s)$. For each $h \in [s]$, the pair (I_h, T_h) is $(c, 1/(6k), \alpha)$-splittable, so let $I_h = L_h \cup M_h \cup R_h$ be the left, middle, and right intervals of I_h; furthermore, let $T_h^{(L)}$ be the $(12\dots c)$-patterns in L_h which appear as prefixes of T_h, and $T_h^{(R)}$ be the $(12\dots(k-c))$-patterns in R_h which appear as suffixes of T_h in R_h. Thus, the restricted function $f_{|L_h}\colon L_h \to \mathbb{R}$ contains $|T_h|$ disjoint $(12\dots c)$-patterns, and $f_{|R_h}\colon R_h \to \mathbb{R}$ contains $|T_h|$ disjoint $(12\dots(k-c))$-patterns. This naturally leads to a recursive application of Theorem II.2 to the function $f_{|L_h}$ with $k_0 = c$, and to the function $f_{|R_h}$ with $k_0 = k - c$, for all $h \in [s]$.

C. Tree descriptors

We now introduce the notion of *tree descriptors*, which will summarize information about a function f after applying Theorem II.2 recursively. Then, we state the main structural result for functions that are ε-far from $(12\dots k)$-free. The goal is to say that every function which is ε-far from $(12\dots k)$-free either has many growing suffixes, or there exists a tree descriptor which describes the behavior of many disjoint, length-k monotone subsequences in the function. The following two definitions make up the notion of a tree descriptor representing a function. Figure 7 shows an example of Definitions II.6 and II.7.

Definition II.6. *Let $k_0 \in \mathbb{N}$ and $\delta \in (0,1)$. A (k_0, δ)-*weighted-tree *is a pair (G, ϱ), where*

- *$G = (V, E, w)$ is a rooted binary tree with edges labeled by a function $w\colon E \to \{0,1\}$. Every non-leaf node has two outgoing edges, e_0, e_1 with $w(e_0) = 0$ and $w(e_1) = 1$. The set of leaves $V_\ell \subseteq V$ satisfies $|V_\ell| = k_0$, and $\le_G$ is the total order defined on the leaves by the values of w on a root-to-leaf path.*[11]
- *$\varrho\colon V \to [\lceil \log(1/\delta) \rceil]$ is a function that assigns a positive integer to each node of G.*

In the next definition, we show how we use weighted trees to represent a function f and a set of disjoint, length-k_0 monotone subsequences.

Definition II.7. *Let $k, k_0 \in \mathbb{N}$ be such that $1 \le k_0 \le k$, let $\alpha \in (0,1)$, let $I \subseteq \mathbb{N}$ be an interval, and let $f\colon I \to \mathbb{R}$ be a function. Let $T \subseteq I^{k_0}$ be a set of disjoint monotone subsequences of f. A triple (G, ϱ, I) is called a (k, k_0, δ)-*tree descriptor[12] *of (f, T, I), if (G, ϱ) is a (k_0, δ)-weighted tree, I is a function $\mathsf{I}\colon V \to \mathcal{P}(\mathcal{I})$ (where $V = V(G)$), and the following recursive definition holds.*

1) *If $k_0 = 1$ (so $T \subseteq I$),*
   - *The graph $G = (V, E, w)$ is the rooted tree with one node, r, and no edges.*
   - *The function $\varrho\colon V \to [\lceil \log(1/\delta) \rceil]$ (simply mapping one node) satisfies $2^{-\varrho(r)} \le |T|/|I| \le 2^{-\varrho(r)+1}$.*
   - *The map $\mathsf{I}\colon V \to \mathcal{S}(I)$ is given by $\mathsf{I}(r) = \{\{t\} : t \in T\}$.*
2) *If $k_0 > 1$,*
   - *The graph $G = (V, E, w)$ is a rooted binary tree with k_0 leaves. We refer to the root by r, the left child of the root (namely, the child incident with the edge given 0 by w) by v_{left}, and the right child of the root (the child incident with the edge given 1) by v_{right}. Let c be the number of leaves in the subtree of v_{left}, so v_{right} has $k_0 - c$ leaves in its subtree.*
   - *Write $\mathsf{I}(r) = \{I_1, \dots, I_s\}$. Then $I_1, \dots, I_s$ are disjoint sub-intervals of I, and, setting $T_i = (I_i)^{k_0} \cap T$, the pairs $(I_1, T_1), \dots, (I_s, T_s)$ form a*

[11] Specifically, for $l_1, l_2 \in V_\ell$ at depths d_1 and d_2, with root to leaf paths $(r, u^{(1)}, \dots, u^{(d_1-1)}, l_1)$ and $(r, v^{(1)}, \dots, v^{(d_2-1)}, l_2)$, then $l_1 \le_G l_2$ if, and only if, $(w(r, u^{(1)}), w(u^{(1)}, u^{(2)}), \dots, w(u^{(d_1-1)}, l_1)) \le (w(r, v^{(1)}), w(v^{(1)}, v^{(2)}), \dots, w(v^{(d_2-1)}, l_2))$ in the natural partial order on $\{0,1\}^*$.

[12] We shall sometimes refer to this as a k_0-tree descriptor, in particular when k, δ are not crucial to the discussion.

$(c, 1/(6k), 2^{-\varrho(r)})$*-splittable collection of* T*, and*

$$2^{-\varrho(r)} \sum_{h=1}^{s} |I_h| \geq \frac{|T|}{\mathrm{poly}(k, \log(1/\delta))^k}.$$

- *For each* $h \in [s]$ *there exists a partition* (L_h, M_h, R_h) *of* I_h *that satisfies Definition II.5, such that the sets* $T_h^{(L)}$*, of prefixes of length* c *of subsequences in* T_h*, and* $T_h^{(R)}$*, of suffixes of length* $k_0 - c$ *of subsequences in* T_h*, satisfy* $T_h^{(L)} \subseteq (L_h)^c$ *and* $T_h^{(R)} \subseteq (R_h)^{k_0-c}$*. Moreover, the following holds.*
  The tuple $(G_{\mathsf{left}}, \varrho_{\mathsf{left}}, \mathsf{I}_{h,\mathsf{left}})$ *is a* (k, c, δ)*-tree descriptor of* f*,* $T_h^{(L)}$*, and* L_h*, where* G_{left} *is the subtree rooted at* v_{left}*,* ϱ_{left} *is the restriction of* ϱ *to the subtree* G_{left}*, and* $\mathsf{I}_{h,\mathsf{left}}$ *is defined by* $\mathsf{I}_{h,\mathsf{left}}(v) := \{J \in \mathsf{I}(v) \colon J \subseteq L_h\}$ *for all* $v \in G_{\mathsf{left}}$*. Analogously, the tuple* $(G_{\mathsf{right}}, \varrho_{\mathsf{right}}, \mathsf{I}_{h,\mathsf{right}})$ *is a* $(k, k_0 - c, \delta)$*-tree descriptor of* f*,* $T_h^{(R)}$*, and* R_h*, where* $G_{\mathsf{right}}, \varrho_{\mathsf{right}}, \mathsf{I}_{h,\mathsf{right}}$ *are defined analogously.*

We remark that it is *not* the case that for every function $f \colon I \to \mathbb{R}$ defined on an interval I, and for every $T \subseteq I^{k_0}$ which is a set of disjoint, length-k_0 monotone subsequences of f, there must exist a k_0-tree descriptor which represents (f, T, I). The goal will be to apply Theorem II.2 recursively whenever we are in (2), and to find a sufficiently large set T of disjoint length-k monotone subsequences, as well as a k-tree descriptor which represents (f, T, I).

D. The structural dichotomy theorem

We are now in a position to state the main structural theorem of far-from-$(12\ldots k)$-free sequences, which guarantees that every far-from-$(12\ldots k)$-free sequence either has many growing suffixes, or can be represented by a tree descriptor. The algorithm for finding a $(12\ldots k)$-pattern will proceed by considering the two cases independently. The first case, when a sequence has many growing suffixes, is easy for algorithms; we will give a straight-forward sampling algorithm making roughly $O_k(\log n/\varepsilon)$ queries. The second case, when a sequence is represented by a tree descriptor is the "hard" case for the algorithm.

Theorem II.3 (Main structural result)**.** *Let* $k \in \mathbb{N}$*,* $\varepsilon > 0$*, and let* $f \colon [n] \to \mathbb{R}$ *be a function which is* ε*-far from* $(12\ldots k)$*-free. Then one of the following holds, where* $C > 0$ *is a large constant.*

- *There exists a parameter* $\alpha \geq \varepsilon/\mathrm{poly}(k, \log(1/\varepsilon))^k$*, and a set* $H \subseteq [n]$ *of indices which start an* $(\alpha, Ck\alpha)$*-growing suffix, with*

$$\alpha|H| \geq \frac{\varepsilon n}{\mathrm{poly}(k, \log(1/\varepsilon))^k},$$

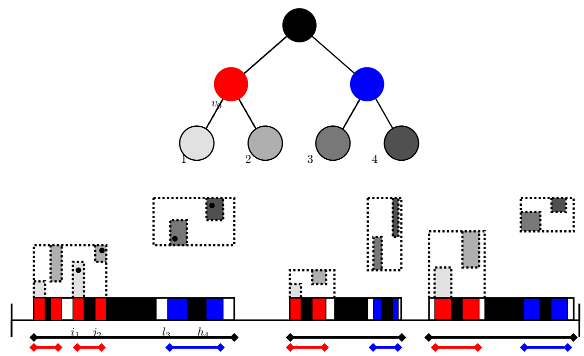

Figure 7. Depiction of a tree descriptor (G, ϱ, I) representing (f, T, I), as defined in Definitions II.6 and II.7. The graph G displayed above is a rooted tree with four leaves, which are ordered and labeled left-to-right. The root node r, filled in black, has its corresponding intervals from $\mathsf{I}(r)$ shown below the sequence as three black intervals. Each of the black intervals in $\mathsf{I}(r)$ is a $(2, \alpha, \beta)$-splittable interval, for $\alpha \approx 1/3$ and $\beta \geq 1/6$. Then, the root has the left child v_0, filled in red, and the right child v_1, filled in blue. The red intervals are those belonging to $\mathsf{I}(v_0)$, and the blue intervals are those belonging to $\mathsf{I}(v_1)$. Each black interval in $\mathsf{I}(r)$ has a left part, which contains intervals in $\mathsf{I}(v_0)$, and a right part, which contains intervals in $\mathsf{I}(v_1)$. The red and blue intervals in $\mathsf{I}(v_0)$ and $\mathsf{I}(v_1)$ are also $(1, \alpha, \beta)$-splittable, and the left part of the red intervals contains indices which will form the 1 in the monotone pattern of length 4, and the right part of the red intervals contains indices which will form the 2. Likewise, the left part of blue intervals will contain the indices corresponding to 3, and the right part of the blue intervals will contain indices corresponding to 4. The regions where the indices from T lie are shown above the sequence, where the indices 1–4 of some monotone pattern in T lie in regions which are progressively darker. In order to see how a monotone subsequence may be sampled given that (G, ℓ, I) is a tree descriptor for (f, T, I) with sufficiently large T, consider indices i_1 and j_2 that belong to some subsequences from T, and lie in different shaded regions of the same red interval, within a black interval; and furthermore, l_3 and h_4 belong to some subsequence from T, and lie in different shaded regions of the same blue interval, within the same black interval as i_1 and j_2; then, the subsequence (i_1, j_2, l_3, h_4) is a monotone subsequence even though $(i_1, j_2, l_3, h_4) \notin T$.

- *or there exists a set* $T \subseteq [n]^k$ *of disjoint monotone subsequences of* f *satisfying*

$$|T| \geq \frac{\varepsilon n}{\mathrm{poly}(k, \log(1/\varepsilon))^{k^2}}$$

  and a (k, k, β)*-tree descriptor* (G, ϱ, I) *which represents* $(f, T, [n])$*, where* $\beta \geq \varepsilon/\mathrm{poly}(k, \log(1/\varepsilon))^{k^2}$.

Proof: We shall prove the following claim, by induction, for all $k_0 \in [k]$. Here $C > 0$ is a large constant, and $C' > 0$ is a large enough constant such that $\alpha \geq \delta/(C'k^5)$ in the statement of Theorem II.2, applied with the constant C.

Claim II.8. *Let* $K = C'k^5$ *and let* $P(\cdot, \cdot)$ *be the function from the statement of Theorem II.2; so* $P(x, y) = \mathrm{poly}(x, \log y)$*, and we may assume that* P *is increasing in both variables. Let* $A(\cdot, \cdot)$ *and* $B(\cdot, \cdot)$ *be increasing functions,* *such that*

$$A(k_0,1/\delta) \geq 12k\lceil\log(K^{k_0}/\delta)\rceil \cdot P(k,1/\delta)\cdot A(k_0-1,K/\delta)$$
$$A(1,1/\delta)=1/\delta$$
$$B(k_0,1/\delta)\geq 2\cdot P(k,K/\delta)\cdot (2k\lceil\log(KB(k_0-1,K/\delta)/\delta)\rceil)^{2k_0} \cdot B(k_0-1,K/\delta)$$
$$B(1,1/\delta)=1/\delta \tag{5}$$

Note that there exists such $A(\cdot,\cdot)$ and $B(\cdot,\cdot)$ with $A(k,1/\delta) = (\mathrm{poly}(k,\log(1/\delta)))^k$ and $B(k,1/\delta) = (\mathrm{poly}(k,\log(1/\delta)))^{k^2}$.

Let $I\subseteq\mathbb{N}$ be an interval, let g be a sequence $g\colon I\to\mathbb{R}$, let $T_0\subseteq I^{k_0}$ be a set of disjoint length-k_0 monotone subsequences, and define $\delta := |T_0|/|I|$. Then

1) *Either there exists $\alpha\geq\delta/K^{k_0}$, which is an integer power of $1/2$, along with a set $H\subseteq I$ of $(\alpha, Ck\alpha)$-growing suffix start points such that*
$$\alpha|H|\geq\frac{\delta|I|}{A(k_0,1/\delta)},$$
2) *Or there exists a set $T\subseteq I^{k_0}$ of disjoint k_0-tuples satisfying $E(T)\subseteq E(T_0)$ and*
$$|T|\geq\frac{|T_0|}{B(k_0,1/\delta)}$$
and a (k,k_0,α)-tree descriptor (G,ϱ,I) for (g,T,I), where $\alpha\geq\delta/B(k_0,1/\delta)$.

Note that since f is ε-far from $(12\dots k)$-free, there is a set $T_0\subseteq[n]^k$ of at least $\varepsilon n/k$ disjoint length-k monotone subsequences. By applying the above claim for $k_0=k$, T_0, $[n]$ and f, the theorem follows. Thus, it remains to prove the claim; we proceed by induction.

- if $k_0=1$: Note that here T_0 is a subset of I. We define the $(k,1,\delta)$-tree descriptor (G,ϱ,I) which represents $f, T=T_0, I$ in the natural way:
  - $G=(V,E)$ is a rooted tree with one node: $V=\{r\}$ and $E=\emptyset$.
  - $\varrho\colon V\to\mathbb{N}$ is given by $\varrho(r)=\lceil\log(1/\delta)\rceil$, so $2^{-\varrho(r)}\leq|I\cap T|/|I|\leq 2^{-\varrho(r)+1}$.
  - $\mathsf{I}\colon V\to\mathcal{S}(I)$ is given by $\mathsf{I}(r)=\{\{t\}: t\in T\}$.
- if $2\leq k_0\leq k$: By Theorem II.2, there exists $\alpha\geq\delta/K$ such that one of (1) and (2), from the statement of the theorem, holds.
  - If (1) holds, there is a set $H\subseteq I$ of $(\alpha,Ck\alpha)$-growing suffix start points with
  $$\alpha|H|\geq\frac{\delta|I|}{P(k,1/\delta)};$$
  note that we may assume that α is an integer power of $1/2$.[13]
  - Otherwise, (2) holds, and we are given an integer $c\in[k_0-1]$, a set T of disjoint length-k_0 monotone subsequences, with $E(T)\subseteq E(T_0)$, and a $(c,1/(6k),\alpha)$-splittable collection of T into disjoint interval-tuple pairs $(I_1,T_1),\dots,(I_s,T_s)$, such that
  $$\alpha\sum_{h=1}^{s}|I_h|\geq\frac{|T_0|}{P(k,1/\delta)}=\frac{\delta|I|}{P(k,1/\delta)}.$$
  Recall that by definition of splittability, $|T_h|/|I_h|\geq\alpha$ for every $h\in[s]$.

If (1) holds, we are done; so we assume that (2) holds. For each $h\in[s]$, since (I_h,T_h) is a $(c,1/(6k),\alpha)$-splittable pair, there exists a partition (L_h,M_h,R_h) that satisfies the conditions stated in Definition II.5. Let $T_h^{(L)}$ be the collection of prefixes of length c of subsequences in T_h, and let $T_h^{(R)}$ be the collection of suffixes of length k_0-c of subsequences in T_h.
We apply the induction hypothesis to each of the pairs $(L_h,T_h^{(L)})$ and $(R_h,T_h^{(R)})$. We consider two cases for each $h\in[s]$.

1) (1) holds for either $(L_h,T_h^{(L)})$ or $(R_h,T_h^{(R)})$. This means that there exists β_h, which is an integer power of $1/2$, and which satisfies $\beta_h\geq\alpha/K^{\max\{c,k_0-c\}}\geq\alpha/K^{k_0-1}\geq\delta/K^{k_0}$, and a set $H_h\subseteq I_h$ of start points of $(\beta_h,Ck\beta_h)$-growing subsequences, such that (using $|R_h|,|L_h|\geq|I_h|/(6k)$)
$$\beta_h|H_h|\geq\frac{\alpha|I_h|}{6k\cdot A(k_0-1,1/\alpha)}$$
2) Otherwise, (2) holds for both $(L_h,T_h^{(L)})$ and $(R_h,T_h^{(R)})$. Setting $\beta=\alpha/B(k_0-1,1/\alpha)$, this means that there exists a (k,c,β)-tree descriptor $(G_h^{(L)},\varrho_h^{(L)},\mathsf{I}_h^{(L)})$, for $(g,\mathcal{L}_h,L_h)$ where $\mathcal{L}_h\subseteq(L_h)^c$ is a set of length-c monotone subsequences, such that $E(\mathcal{L}_h)\subseteq E(T_h^{(L)})$ and
$$|\mathcal{L}_h|\geq\frac{|T_h^{(L)}|}{B(k_0-1,1/\alpha)}, \tag{6}$$
and, similarly, there exists a (k,k_0-c,β)-tree descriptor $(G_h^{(R)},\varrho_h^{(R)},\mathsf{I}_h^{(R)})$ for $(g,\mathcal{R}_h,L_h)$, where $\mathcal{R}_h\subseteq(R_h)^{k_0-c}$ is a set of length-(k_0-c) monotone subsequences, such that $E(\mathcal{R}_h)\subseteq E(T_h^{(R)})$ and
$$|\mathcal{R}_h|\geq\frac{|T_h^{(R)}|}{B(k_0-1,1/\alpha)}. \tag{7}$$
For convenience, we shall assume that $|\mathcal{L}_h|=|\mathcal{R}_h|$, by possibly removing some elements of the largest of the two (and reflecting this in the corresponding tree descriptor).

[13] to be precise and to ensure that we can take α to be an integer power of 2, it might be better to apply Theorem II.2 with constant $2C$, to allow for some slack; this does not change the argument.

Suppose first that

$$\sum_{h:\text{ first case holds for } h} |I_h| \geq \frac{1}{2} \cdot \sum_{h=1}^{s} |I_h|.$$

Since each β_h is an integer power of $1/2$, there are at most $\lceil \log(K^{k_0}/\delta) \rceil$ possible values for β_h. Hence, there exists some β (with $\beta \geq \delta/K^{k_0}$) such that the collection S, of indices $h \in [s]$ for which the first case holds for h and $\beta_h = \beta$, satisfies

$$\sum_{h \in S} |I_h| \geq \frac{1}{2\lceil \log(K^{k_0}/\delta) \rceil} \cdot \sum_{h=1}^{s} |I_h|.$$

Let $H = \bigcup_{h \in S} H_h$. Then H is a set of start points of $(\beta, Ck\beta)$-growing suffixes, with

$$\begin{aligned}
\beta|H| &\geq \frac{\alpha}{6k \cdot A(k_0 - 1, 1/\alpha)} \cdot \sum_{h \in S} |I_h| \\
&\geq \frac{\alpha}{12k\lceil \log(K^{k_0}/\delta) \rceil \cdot A(k_0 - 1, 1/\alpha)} \cdot \sum_{h=1}^{s} |I_h| \\
&\geq \frac{\delta|I|}{12k\lceil \log(K^{k_0}/\delta) \rceil \cdot P(k, 1/\delta) \cdot A(k_0 - 1, 1/\alpha)} \\
&\geq \frac{\delta|I|}{A(k_0, 1/\delta)},
\end{aligned}$$

where the last inequality follows from (5). This proves the claim in this case.

Next, we may assume that

$$\sum_{h:\text{ second case holds for } h} |I_h| \geq \frac{1}{2} \cdot \sum_{h=1}^{s} |I_h|.$$

Note that the number of quadruples $(G_h^{(L)}, \varrho_h^{(L)}, G_h^{(R)}, \varrho_h^{(R)})$ (whose elements are as above) is at most $(2c)^{2c}(2(k_0 - c))^{2(k_0 - c)}(\lceil \log(1/\beta) \rceil)^{2k_0} \leq (2k\lceil \log(1/\beta) \rceil)^{2k_0}$, since the number of trees on l vertices is at most l^l, and we have at most $\lceil \log(1/\beta) \rceil$ possible weights to assign to each of the vertices. It follows that there exists such a quadruple $(G_L^*, \varrho_L^*, G_R^*, \varrho_R^*)$ such that if S is the set of indices h that were assigned this quadruple, then

$$\begin{aligned}
\alpha \cdot \sum_{h \in S} |I_h| &\geq \frac{\alpha}{(2k\lceil \log(1/\beta) \rceil)^{2k_0}} \sum_{2^{\text{nd}} \text{ case holds for } h} |I_h| \\
&\geq \frac{\alpha}{2 \cdot (2k\lceil \log(1/\beta) \rceil)^{2k_0}} \sum_{h=1}^{s} |I_h| \\
&\geq \frac{|T_0|}{2 \cdot P(k, 1/\delta) \cdot (2k\lceil \log(1/\beta) \rceil)^{2k_0}}.
\end{aligned} \tag{8}$$

We form a set $\mathcal{T}_h$ of monotone length-k_0 subsequences by matching elements from $\mathcal{L}_h$ with elements from $\mathcal{R}_h$ for each $h \in S$; that they can be matched follows from the assumption that $|\mathcal{L}_h| = |\mathcal{R}_h|$, and that these form monotone subsequences follows from the assumptions on $\mathcal{L}_h, \mathcal{R}_h$. Set $\mathcal{T} := \cup_{h \in S} \mathcal{T}_h$. Note that $(I_h, \mathcal{T}_h)$ is (k_0, c, β)-splittable by (6) and (7) (using $\beta = \alpha/B(k_0 - 1, 1/\alpha)$). Let (G, ϱ) be the (k, k_0, β)-weighted-tree obtained by taking a root r, with weight $\varrho(r) = \lceil \log(1/\beta) \rceil$, adding the tree (G_L^*, ϱ^*) as a subtree to its left (i.e. the root of this tree is joined to r by an edge with value 0) and adding the tree (G_R^*, ϱ^*) as a subtree to its right. Now, we form a (G, ϱ, I)-tree descriptor by setting

$$\mathsf{I}(v) = \begin{cases} \{I_h : h \in S\} & v = r \\ \bigcup_{h \in S} \mathsf{I}_h^{(L)}(v) & v \in G_L^* \\ \bigcup_{h \in S} \mathsf{I}_h^{(R)}(v) & v \in G_R^*. \end{cases}$$

We claim that (G, ϱ, I) is a (k, k_0, β)-tree descriptor for $(g, \mathcal{T}, I)$. Indeed, $((I_h, \mathcal{T}_h))_{h \in S}$ is a $(c, 1/(6k), 2^{-\varrho(r)})$-splittable collection of $\mathcal{T}$, and, by (8) and because $|T_0| \geq |\mathcal{T}|$

$$\begin{aligned}
2^{-\varrho(r)} \sum_{h \in S} |I_h| &\geq \frac{\alpha}{2} \cdot \sum_{h \in S} |I_h| \\
&\geq \frac{|\mathcal{T}|}{4 \cdot P(k, 1/\delta) \cdot (2k\lceil \log(1/\beta) \rceil)^{2k_0}} \\
&= \frac{|\mathcal{T}|}{\text{poly}(k, \log(1/\delta))^k}.
\end{aligned}$$

The remaining requirements in the recursive defnition of a tree descriptor (see Definition II.7) follow as $(G_L^*, \varrho^*, \mathsf{I}_h^{(L)})$ is a (k, c, β)-tree descriptor for $(g, \mathcal{L}_h, L_h)$ and $(G_L^*, \varrho^*, \mathsf{I}_h^{(R)})$ is a $(k, k_0 - c, \beta)$-tree descriptor for $(g, \mathcal{R}_h, R_h)$ for every $h \in S$. Since $\beta = \alpha/B(k_0 - 1, 1/\alpha) \geq \delta/B(k_0, 1/\delta)$, it follows that (G, ϱ, I) is a $(k, k_0, \delta/B(k_0, 1/\delta))$-tree descriptor for $(g, \mathcal{T}, I)$.

It remains to lower-bound the size of $\mathcal{T}$. Using (7) and (8), we have

$$\begin{aligned}
|\mathcal{T}| = \sum_{h \in S} |\mathcal{R}_h| &\geq \frac{1}{B(k_0 - 1, 1/\alpha)} \cdot \sum_{h \in S} |T_h| \\
&\geq \frac{\alpha}{B(k_0 - 1, 1/\alpha)} \cdot \sum_{h \in S} |I_h| \\
&\geq \frac{|T_0|}{2 \cdot P(k, 1/\delta)(2k\lceil \log(1/\beta) \rceil)^{2k_0} B(k_0 - 1, 1/\alpha)} \\
&\geq \frac{|T_0|}{B(k_0, 1/\delta)}.
\end{aligned}$$

This completes the proof of the inductive claim in this case. ∎

E. Proof of Theorem II.2

We now prove Theorem II.2. For the rest of this section, let $k, k_0 \in \mathbb{N}$, with $1 \leq k_0 \leq k$, be fixed, and let $f : [n] \to \mathbb{R}$ be a fixed function. Let T_0 be a set of δn disjoint monotone subsequences of f of length k_0. We apply Lemma II.3 to the set T_0; this specifies an integer $c \in [k_0 - 1]$ and a subset T

of at least $\delta n/k^2$ disjoint monotone subsequences of length k_0 satisfying the conclusion of Lemma II.3.

Definition II.9. *Let $(i_1,\ldots,i_{k_0}) \in [n]^{k_0}$ be a monotone subsequence with a c-gap. We say that $(i_1,\ldots,i_{k_0})$ is* at scale t *if $2^t \le i_{c+1} - i_c \le 2^{t+1}$, where $t \in \{0,\ldots,\lfloor \log n \rfloor\}$.*

Definition II.10. *Let $(i_1,\ldots,i_{k_0}) \in [n]^{k_0}$ be a monotone subsequence with a c-gap. For $\gamma \in (0,1)$, we say that $\ell \in [n]$* γ-cuts $(i_1,\ldots,i_{k_0})$ at c with slack *if*

$$i_c + \gamma(i_{c+1} - i_c) \le \ell \le i_{c+1} - \gamma(i_{c+1} - i_c). \tag{9}$$

We hereafter consider the parameter setting of $\gamma := 1/3$. For $\ell \in [n]$, $t \in \{0,\ldots,\lfloor \log n \rfloor\}$, and any subset $U \subset T$ of disjoint $(12\ldots k_0)$-patterns in f let

$$A_t(\ell,U) = \{(i_1,\ldots,i_{k_0}) \in U : \substack{(i_1,\ldots,i_{k_0}) \text{ is at} \\ \text{scale } t \text{ and is } \gamma\text{-cut} \\ \text{at } c \text{ with slack by } \ell}\} \tag{10}$$

We note that for each $(i_1,\ldots,i_{k_0}) \in A_t(\ell,U)$, the index i_{c+1} is in $[\ell, \ell + 2^{t+1}]$, and since $A_t(\ell,U)$ is made of disjoint monotone sequences, $|A_t(\ell,U)| \le 2^{t+1}$.

Lemma II.11. *For every $\ell \in [n]$, $t \in \{0,\ldots,\lfloor \log n \rfloor\}$, and $U \subset T$,*

- *Every $(i_1,\ldots,i_{k_0}) \in A_t(\ell,U)$ satisfies*
$$\ell - (k-1)2^{t+1} \le i_1,\ldots,i_c \le \ell - \gamma 2^t$$
$$\ell + \gamma 2^t \le i_{c+1},\ldots,i_{k_0} \le \ell + (k-1)2^{t+1}.$$
- *Let $t_1 \ge t_2 + 1 + \log(1/\gamma) + \log(c+1)$, $(i_1,\ldots,i_{k_0}) \in A_{t_1}(\ell,U)$ and $(j_1,\ldots,j_{k_0}) \in A_{t_2}(\ell,U)$. Then $f(j_{c+1}) < f(i_{c+1})$.*

Proof: Fix any $\ell \in [n]$, $t \in \{0,\ldots,\lfloor \log n \rfloor\}$ and $U \subset T$. To establish the first bullet, consider any $(i_1,\ldots,i_{k_0}) \in A_t(\ell,U)$. By definition of a c-gap sequence, we have

$$i_1 \ge i_{c+1} - c(i_{c+1} - i_c) \ge \ell - (k-1)2^{t+1},$$

using $i_{c+1} - i_c \le 2^{t+1}$ and $i_{c+1} \ge \ell$. By (9), we have $i_c \le \ell - \gamma 2^t$ (using $i_{c+1} - i_c \ge 2^t$). The first inequality follows as $i_1 < \cdots < i_c$. The inequality for $i_{c+1},\ldots,i_{k_0}$ follows similarly.

For the second bullet, let $(i_1,\ldots,i_{k_0}) \in A_{t_1}(\ell,U)$ and $(j_1,\ldots,j_{k_0}) \in A_{t_2}(\ell,U)$ and suppose that $2^{t_1} \ge 2^{t_2+1} \cdot (c+1)/\gamma$. We have $i_c \le \ell - \gamma 2^{t_1}$ and $j_c \ge \ell - 2^{t_2+1}$ (using (9) and (10)), from which it follows that $j_c > i_c$. Similarly, $i_1 < i_c \le \ell - \gamma 2^{t_1}$ and $j_1 \ge \ell - (c-1)2^{t_2+1}$, implying that $j_1 > i_1$, and $i_{c+1} \ge \ell + \gamma 2^{t_1}$ and $j_{c+1} \le \ell + 2^{t_2+1}$, which implies that $i_{c+1} > j_{c+1}$. The inequality $f(j_{c+1}) < f(i_{c+1})$ follows from the assumption that T satisfies (3) from Lemma II.3. ∎

The proof of Theorem II.2 will follow by considering a random $\boldsymbol{\ell} \sim [n]$ and the sets $A_1(\boldsymbol{\ell},T),\ldots,A_{\lfloor \log n \rfloor}(\boldsymbol{\ell},T)$. By looking at how the sizes of the sets $A_1(\boldsymbol{\ell},T),\ldots,A_{\log n-1}(\boldsymbol{\ell},T)$ vary, we will be able to say that $\boldsymbol{\ell}$ is the start of a growing suffix, or identify a splittable interval. Towards this goal, we first establish a simple lemma; here $v(\ell,U)$ is defined to be $\sum_{t=0}^{\lfloor \log n \rfloor} |A_t(\ell,U)|/2^t$.

Lemma II.12. *Let $U \subset T$ be any subset and $\boldsymbol{\ell} \sim [n]$ be sampled u.a.r. Then $\mathbb{E}_{\boldsymbol{\ell}\sim[n]}[v(\boldsymbol{\ell},U)] \ge \frac{|U|}{3n}$.*

Proof: Fix a sequence $i = (i_1,\ldots,i_{k_0}) \in U$, and let $t(i) \in \{0,\ldots,\lfloor \log n \rfloor\}$ be its scale. Then, the probability (over a uniformly random $\boldsymbol{\ell}$ in $[n]$) that i belongs to $A_{t(i)}(\boldsymbol{\ell},U)$ is lower bounded as

$$\mathop{\mathbf{Pr}}_{\boldsymbol{\ell}\sim[n]}[i \in A_{t(i)}(\boldsymbol{\ell},U)] \ge \frac{(1-2\gamma)2^{t(i)}}{n} = \frac{2^{t(i)}}{3n}.$$

Therefore, $\sum_{t=0}^{\log n-1} \sum_{i\in U:\, t(i)=t} \mathbf{Pr}_{\boldsymbol{\ell}\sim[n]}[i \in A_t(\boldsymbol{\ell},U)]/2^t \ge |U|/(3n)$, or, equivalently, since we have $\mathbf{Pr}_{\boldsymbol{\ell}\sim[n]}[i \in A_t(\boldsymbol{\ell},U)] = 0$ for $t \ne t(i)$, $\mathbb{E}_{\boldsymbol{\ell}\sim[n]}\left[\sum_{t=0}^{\log n-1} \frac{|A_t(\boldsymbol{\ell},U)|}{2^t}\right] = \mathbb{E}_{\boldsymbol{\ell}\sim[n]}\left[\sum_{t=0}^{\log n-1} \sum_{i\in U} \frac{\mathbb{1}\{i\in A_t(\boldsymbol{\ell},U)\}}{2^t}\right] \ge \frac{|U|}{3n}$, establishing the lemma. ∎

We next establish an auxiliary lemma that we will use in order to find growing suffixes.

Lemma II.13. *Let $\ell \in [n]$ and $U \subset T$ be such that every $t \in \{0,\ldots,\lfloor \log n \rfloor\}$ satisfies $|A_t(\ell,U)|/2^t \le \beta$. Then, if $\ell' \in [n]$ is any index satisfying*

$$\max\{i_c : \substack{(i_1,\ldots,i_{k_0})\in A_t(\ell,U), \\ t\in\{0,\ldots,\lfloor \log n \rfloor\}}\} \le \ell' \le \ell, \tag{11}$$

then ℓ' is the start of an $(4\beta, v(\ell,U)/(12\log k))$-growing suffix.

Proof: Let $\Delta = 1 + \log(1/\gamma) + \log(c+1)$, and notice that $3 \le \Delta \le 3\log k$. Then, there exists a set $\mathcal{T} \subseteq \{0,\ldots,\lfloor \log n \rfloor\}$ such that

1) All distinct $t,t' \in \mathcal{T}$ satisfy $|t - t'| \ge \Delta$; and,
2) $\sum_{t\in\mathcal{T}} \frac{|A_t(\ell,U)|}{2^t} \ge \frac{1}{\Delta+1} \sum_{t=0}^{\log n-1} \frac{|A_t(\ell,U)|}{2^t} = \frac{v(\ell,U)}{\Delta+1}$.

(Such a set exists by an averaging argument.) Now, consider the sets, for $t \in \{0,\ldots,\lfloor \log n \rfloor\}$,

$$D_t(\ell) = \begin{cases} \{i_{c+1} : (i_1,\ldots,i_{k_0}) \in A_t(\ell,U)\} & \text{if } t \in \mathcal{T} \\ \emptyset & \text{o.w.} \end{cases}$$

Considering any $\ell' \in [n]$ satisfying (11), we have the following for all $t \in \{0,\ldots,\lfloor \log n \rfloor\}$ with $D_t(\ell) \ne \emptyset$: $\ell - 2^{t+1} \le \ell' \le \ell$; $\min D_t(\ell) \ge \ell + 2^t/3$; and $\max D_t(\ell) \le \ell' + 2^{t+1}$. Therefore, $D_t(\ell) \subset S_{t-1}(\ell') \cup S_t(\ell') \cup S_{t+1}(\ell')$. (Recall that $S_t(a) = [a + 2^{t-1}, a + 2^t)$.) For each $t \in \mathcal{T}$, let $n(t) \in \{t-1, t, t+1\}$ satisfying $|D_t(\ell) \cap S_{n(t)}(\ell')| \ge |D_t(\ell)|/3$, and notice that all $n(t) \in \{0,\ldots,\lfloor \log n \rfloor\}$ are distinct since $\Delta \ge 3$.

The first condition in Definition II.4 holds as the densities of $D_t(\ell) \cap S_{n(t)}(\ell')$ in the corresponding intervals $S_{n(t)}(\ell')$ are upper bounded by $|D_t(\ell)|/|S_{n(t)}(\ell')| \le$

$|A_t(\ell, U)|/2^{t-2} \leq 4\beta$, and the sum of these densities satisfies

$$\sum_{t\in\mathcal{T}} \frac{|D_t(\ell) \cap S_{n(t)}(\ell')|}{|S_{n(t)}(\ell')|} \geq \sum_{t\in\mathcal{T}} \frac{|D_t(\ell)|}{3\cdot 2^t} = \sum_{t\in\mathcal{T}} \frac{|A_t(\ell, U)|}{3\cdot 2^t} \geq \frac{v(\ell, U)}{3(\Delta+1)},$$

which is at least $v(\ell, U)/(12\log k)$. The second condition in Definition II.4 holds, because for any choice of $b \in D_t(\ell), b' \in D_{t'}(\ell)$ with $t < t'$, we have $t' \geq t + \Delta$ (by the choice of $\mathcal{T}$), and hence $f(b) < f(b')$ by the second item of Lemma II.11. ■

Lemma II.14. *For every $\eta > 0$, there exists a subset $U \subset T$ such that every $(i_1, \ldots, i_{k_0}) \in U$ has i_c as the start of an $(1, \eta)$-growing suffix, and every $\ell \in [n]$ satisfies $v(\ell, T\setminus U) \leq 12\eta\log(k)$.*

Proof: Define sets U_j, elements ℓ_j, and k_0-tuples $(i_{j,1}, \ldots, i_{j,k_0})$ recursively as follows. Set $U_0 := \emptyset$, and given a set U_{j-1}, if $v(\ell, T\setminus U_{j-1}) \leq 12\eta\log k$ for every $\ell \in [n]$, stop; otherwise, let $\ell_j \in [n]$ be such that $v(\ell_j, T \setminus U_{j-1}) > 12\eta\log k$ and define $U_j = U_{j-1} \cup \{(i_{j,1}, \ldots, i_{j,k_0})\}$, where

$$i_{j,c} = \max\{i_c : {\scriptstyle (i_1,\ldots,i_{k_0})\in T\setminus U_j \text{ and } \atop (i_1,\ldots,i_{k_0}) \text{ is } \gamma\text{-cut by } \ell}\}.$$

Let j^* be the maximum j for which U_j was defined, and set $U := U_{j^*}$. Every k_0-tuple in U is of the form $(i_{j,1}, \ldots, i_{j,k_0})$ for some $j \leq j^*$. By Lemma II.13, applied with $\ell = \ell_j$, $U = T \setminus U_{j-1}$, $i_{j,c}$, it follows that $i_{j,c}$ is the start of an $(1, \eta)$-growing suffix, for every j for which U_j was defined. Lemma II.14 follows. ■

We let $C > 0$ be a large enough constant. Let $U \subset T$ be the set obtained from Lemma II.14 with $\eta = Ck$, and suppose that $|U| \geq |T|/2$. Then, we may let $\alpha = 1$ and $H = \{i_c : (i_1, \ldots, i_{k_0}) \in U\}$. Notice that every index in H is the start of an $(\alpha, Ck\alpha)$-growing suffix, and since $|H| \geq |T|/2$, we obtain the first item in Theorem II.2. Suppose then, that $|U| < |T|/2$, and consider the set $V = T\setminus U$. By definition of V, we now have $v(\ell, V) \leq 12Ck\log k$ for every $\ell \in [n]$. Let b_0 be the largest integer which satisfies $2^{b_0} \leq 12Ck\log k$ and b_1 be the smallest integer which satisfies $2^{-b_1} \leq \delta/(12k^2)$, so $2^{b_0} \lesssim 2^{b_1} \asymp k^2/\delta$. For $-b_0 \leq j \leq b_1$, consider the pairwise-disjoint sets

$$B_j = \left\{\ell \in [n] : 2^{-j} \leq v(\ell, V) \leq 2^{-j+1}\right\}, \tag{12}$$

and note that by Lemma II.12, since $|V| \geq |T|/2 \geq \delta n/2k^2$,

$$\frac{1}{n}\sum_{j=-b_0}^{b_1} |B_j| \cdot 2^{-j+1} \geq \frac{1}{n}\sum_{\ell\in[n]} v(\ell, V) \geq \frac{\delta}{6k^2}.$$

Thus, denoting $\mu := \frac{\delta}{6k^2(b_1+b_0+1)} \asymp \frac{\delta}{k^2\log(k/\delta)}$, there is an integer $-b_0 \leq j^* \leq b_1$ that satisfies

$$|B_{j^*}| \cdot 2^{-j^*} \geq \mu n. \tag{13}$$

Lemma II.15. *There exists a deterministic algorithm,* `GreedyDisjointIntervals`(f, B, j), *which takes three inputs: a function $f\colon [n] \to \mathbb{R}$, a set $B \subseteq [n]$ of integers, and an integer $j \in [-b_0, b_1]$, and outputs a collection $\mathcal{I}$ of interval-tuple pairs or a subset $H \subseteq B$. An execution of the algorithm* `GreedyDisjointIntervals`(f, B_{j^*}, j^*) *where μ, B_{j^*} and j^* are defined in* (13)*, satisfies one of the following two conditions, where $C > 0$ is a large constant.*

- *The algorithm returns a set $H \subseteq B$ of indices that start a $(4\cdot 2^{-j^*}/(Ck\log k), 2^{-j^*}/(12\log k))$-growing suffix, and $|H| \geq 2^{j^*-1}\mu n$; or*
- *The algorithm returns a $(c, 1/(6k), 2^{-j^*}/(8Ck^2\log k))$-splittable collection $(I_1, T_1), \ldots, (I_s, T_s)$, where $\sum_{h=1}^{s} |I_h| \geq 2^{j^*-2}\mu n$.*

Subroutine `GreedyDisjointIntervals`(f, B, j)

Input: A function $f\colon [n] \to \mathbb{R}$, a set $B \subseteq [n]$ and an integer j, such that every $\ell \in B$ satisfies $2^{-j} \leq v(\ell, V) \leq 2^{-j+1}$.
Output: a set of disjoint intervals-tuple pairs $(I_1, T_1), \ldots, (I_s, T_s)$ or a subset $H \subseteq B$.

1) Let $\mathcal{I}$ be a collection of interval-tuple pairs, which is initially empty.
2) Consider the map $q\colon B \to \{0, \ldots, \lfloor\log n\rfloor\} \cup \{\perp\}$ defined by
$$q(\ell) = \begin{cases} \perp & \text{if } {\scriptstyle \forall t\in\{0,\ldots,\lfloor\log n\rfloor\},} \atop \frac{|A_t(\ell,V)|}{2^t} < \frac{2^{-j}}{Ck\log k} \\ \max\left\{t : \frac{|A_t(\ell,V)|}{2^t} \geq \frac{2^{-j}}{Ck\log k}\right\} & \text{o.w.} \end{cases}.$$
3) Let $H = \{\ell \in B : q(\ell) = \perp\}$, and **return** H if $|H| \geq |B|/2$.
4) Otherwise, let $D \leftarrow B \setminus H$ and repeat the following until $D = \emptyset$:
   - Pick any $\ell \in D$ where $q(\ell) = \max_{\ell'\in D} q(\ell')$, and let $t = q(\ell)$.
   - Let $I \leftarrow [\ell - k2^{t+1}, \ell + k2^{t+1}] \cap [n]$ and $T' \leftarrow A_t(\ell, V)$.
   - Obtain T'' from T' as follows: find a value ν such that at least $|T'|/2$ of tuples $(i_1, \ldots, i_{k_0}) \in T'$ satisfy $f(i_c) \leq \nu$, and at least $|T'|/2$ of tuples $(i_1, \ldots, i_{k_0}) \in T'$ satisfy $f(i_{c+1}) > \nu$ (ν could be taken to be the median of the multiset $\{f(i_c) : (i_1, \ldots, i_{k_0}) \in T'\}$). Recombine these prefixes and suffixes (matching them in one-to-one correspondence) to obtain a set of disjoint k_0-tuples T'' of size $|T''| \geq |T'|/2$.
   - Append (I, T'') to $\mathcal{I}$, and let $D \leftarrow D \setminus [\ell - 2\cdot k2^{t+1}, \ell + 2\cdot k2^{t+1}]$.
5) **return** $\mathcal{I}$.

Figure 8. Description of the `GreedyDisjointIntervals` subroutine.

Proof: It is clear that the algorithm always terminates, and outputs either a collection $\mathcal{I}$ of interval-tuple pairs or a subset $H \subseteq B$. Suppose that the input of the algorithm, (f, B_{j^*}, j^*), satisfies (13), and consider the two possible types of outputs.

If the algorithm returns a set $H \subseteq B_{j^*}$ (in step 3), then we have $|H| \geq \frac{|B|}{2} \geq \frac{1}{2} \cdot 2^{j^*}\mu n$ (the second inequality by (13)). (To see why the elements of H start $(4\cdot 2^{-j^*}/(Ck\log k), 2^{-j^*}/(12\log k))$-growing suffixes (Definition II.4), notice that we may apply Lemma II.13 with $\ell' = \ell$ and $\beta = 2^{-j^*}/(Ck\log k)$.)

If, instead, the algorithm returns a collection $\mathcal{I} = ((I_h, T_h) : h \in [s])$ in step 5, we have that, by construction, each T_h is obtained from a set $T'_h = A_t(\ell, V)$ for some ℓ with $q(\ell) \neq \perp$. Consequently, for all $h \in [s]$ we have

$$\frac{|T_h|}{|I_h|} \geq \frac{|T'_h|}{2|I_h|} \geq \frac{|A_{q(\ell)}(\ell, V)|}{4k \cdot 2^{q(\ell)+1}} \geq \frac{1}{8k} \cdot \frac{2^{-j^*}}{Ck\log k}. \tag{14}$$

(from the definition of $q(\ell)$). To argue that $\sum_{h=1}^{s}|I_h|$ is large, observe that, since we did not output the set H, we must have had $|D| > |B_{j^*}|/2$. Since, when adding (I_h, T_h) (corresponding to some ℓ_h) to $\mathcal{I}$ we remove at most $4k2^{q(\ell)+1} = 2|I_h|$ elements from D, in order to obtain an empty set D and reach step 5 we must have $\sum_{h=1}^{s}|I_h| \geq |B_{j^*}|/4$, which is at least $2^{j^*}\mu n/4$ by (13). Moreover, the sets I_h are disjoint: this is because of our choice of maximal $q(\ell)$ in step 4, which ensures that after removing $[\ell - 2k2^{q(\ell)+1}, \ell + 2k2^{q(\ell)+1}]$ in step 4 there cannot remain any $\ell' \in D$ with $[\ell' - k2^{q(\ell')+1}, \ell' + k2^{q(\ell')+1}] \cap I_h \neq \emptyset$.

Thus, it remains to prove that $\mathcal{I}$ is a $(c, 1/(6k), 2^{-j^*}/(8Ck^2\log k))$-splittable collection. To do so, consider any $(I_h, T_h) \in \mathcal{I}$. The first condition in Definition II.5 of splittable pairs, namely that $|T_h|/|I_h| \geq 2^{-j^*}/(8Ck^2\log k)$ holds due to (14). Recalling step 4, we have $I_h = [\ell - k2^{t+1}, \ell + k2^{t+1}]$ for some ℓ, where $t = q(\ell)$, and T_h obtained from $T'_h = A_t(\ell, V)$. Set

$$\begin{aligned} L_h &:= [\ell - k2^{t+1}, \ell - \gamma 2^t], \\ M_h &:= (\ell - \gamma 2^t, \ell + \gamma 2^t), \\ R_h &:= [\ell + \gamma 2^t, \ell + k2^{t+1}]. \end{aligned}$$

This is a partition of I_h into three adjacent intervals whose size is at least $|I_h|/(6k)$ (recall that $\gamma = 1/3$). Moreover, for every $(i_1, \ldots, i_{k_0}) \in T'_h$, the c-prefix $(i_1, \ldots, i_c)$ is in $(L_h)^c$ while the $(k_0 - c)$-suffix $(i_{c+1}, \ldots, i_{k_0})$ is in $(R_h)^{k_0 - c}$, by the first item of Lemma II.11. Since T_h is obtained from a subset of these very prefixes and suffices, the conclusion holds for T_h as well. Moreover, our construction of T_h from T'_h guarantees that the last requirement in Definition II.5 holds: for every prefix $(i_1, \ldots, i_c)$ of a tuple in T_h and suffix $(j_1, \ldots, j_{k_0 - c})$ of a tuple in T_h, we have $f(i_c) < f(j_1)$. This shows that (I_h, T_h) is $(c, 1/(6k), 2^{-j^*}/(8Ck^2\log k))$-splittable, and overall that $\mathcal{I}$ is a $(c, 1/(6k), 2^{-j^*}/(8Ck^2\log k))$-splittable collection as claimed. ∎

Theorem II.2 follows by executing `GreedyDisjointIntervals`(f, B_{j^*}, j^*). If the algorithm outputs a set $H \subseteq B_{j^*}$, set $\alpha = 4\cdot 2^{-j^*}/(Ck\log k)$, so we have identified a subset H of $(\alpha, C'\alpha k)$-growing suffixes (where $C' = C/48$) satisfying $\alpha|H| \geq \delta n/\mathrm{poly}(k, \log(1/\delta)) = |T_0|/\mathrm{poly}(k, \log(1/\delta))$ (using the definition of μ before (13)). Otherwise, set $\alpha = 2^{-j^*}/(8Ck^2\log k)$, and the algorithm outputs a $(c, 1/(6k), \alpha)$-splittable collection $\{(I_1, T_1), \ldots, (I_s, T_s)\}$ of the set $T' := \cup_{h\in[s]}T_h$. Clearly, $E(T') \subseteq E(T)$, and moreover, $\alpha\sum_{h=1}^{s}|I_h| \geq \delta n/\mathrm{poly}(k, \log(1/\delta)) = |T_0|/\mathrm{poly}(k, \log(1/\delta))$. In fact, $2^{-j^*} = \Omega(\delta/k^2)$ and so $\alpha \geq \Omega(\delta/(k^4\log k))$.

III. The Algorithm

A. High-level plan

We now present the algorithm for finding monotone subsequences of length k.

Theorem III.1. *Consider any fixed value of $k \in \mathbb{N}$. There exists a non-adaptive and randomized algorithm,* $\texttt{Sampler}_k(f, \varepsilon)$*, which takes two inputs: query access to a function $f\colon [n] \to \mathbb{R}$ and a parameter $\varepsilon > 0$. If f is ε-far from $(12\ldots k)$-free, then* $\texttt{Sampler}_k(f, \varepsilon)$ *finds a $(12\ldots k)$-pattern with probability at least $9/10$. The query complexity of* $\texttt{Sampler}_k(f, \varepsilon)$ *is at most*

$$\frac{1}{\varepsilon}\left(\frac{\log n}{\varepsilon}\right)^{\lfloor \log_2 k \rfloor} \cdot \mathrm{poly}(\log(1/\varepsilon)).$$

The particular dependence on k and $\log(1/\varepsilon)$ obtained from Theorem III.1 is on the order of $(k\log(1/\varepsilon))^{O(k^2)}$. The algorithm is divided into two cases, corresponding to the two outcomes from an application of Theorem II.3. Suppose $f\colon [n] \to \mathbb{R}$ is a function which is ε-far from being $(12\ldots k)$-free. By Theorem II.3 one of the followin holds, where $C > 0$ is a large constant.

- Case 1: there exist $\alpha \geq \varepsilon/\mathrm{polylog}(1/\varepsilon)$ and a set $H \subseteq [n]$ of $(\alpha, Ck\alpha)$-growing suffixes where $\alpha|H| \geq \varepsilon n/\mathrm{polylog}(1/\varepsilon)$, or
- Case 2: there exist a set $T \subseteq [n]^k$ of disjoint, length-k monotone sequences, that satisfies $|T| \geq \varepsilon n/(\mathrm{polylog}(1/\varepsilon))$, and a k-tree descriptor (G, ϱ, I) which represents $(f, T, [n])$.

Theorem III.1 follows from analyzing the two cases independently, and designing an algorithm for each.

Lemma III.1 (Case 1)**.** *Consider any fixed value of $k \in \mathbb{N}$, and let $C > 0$ be a large enough constant. There exists a non-adaptive and randomized algorithm,* $\texttt{Sample-Suffix}_k(f, \varepsilon)$ *which takes two inputs: query access to a function $f\colon [n] \to \mathbb{R}$ and a parameter $\varepsilon > 0$. Suppose there exist $\alpha \in (0,1)$ and a set $H \subseteq [n]$ of $(\alpha, Ck\alpha)$-growing suffixes satisfying $\alpha|H| \geq \varepsilon n/\mathrm{polylog}(1/\varepsilon)$,*[14] *then* $\texttt{Sample-Suffix}_k(f, \varepsilon)$ *finds a length-k monotone*

[14]Here we think of k as fixed, so $\mathrm{polylog}(1/\varepsilon)$ is allowed to depend on k. In this lemma, the expression stands for $(k\log(1/\varepsilon))^k$.

subsequence of f with probability at least $9/10$. The query complexity of $\texttt{Sample-Suffix}_k(f,\varepsilon)$ *is at most*

$$\frac{\log n}{\varepsilon} \cdot \mathrm{polylog}(1/\varepsilon).$$

Lemma III.1 above, which corresponds to the first case of Theorem II.3, is proved in Section III-B.

Lemma III.2 (Case 2)**.** *Consider any fixed value of $k \in \mathbb{N}$. There exists a non-adaptive, randomized algorithm,* $\texttt{Sample-Splittable}_k(f,\varepsilon)$ *which takes two inputs: query access to a sequence $f\colon [n] \to \mathbb{R}$ and a parameter $\varepsilon > 0$. Suppose there exists a set $T \subseteq [n]^k$ of disjoint, length-k monotone subsequences of f where $|T| \geq \varepsilon n/\mathrm{polylog}(1/\varepsilon)$,*[15] *as well as a (k,k,α)-tree descriptor (G,ϱ,I) that represents $(f,T,[n])$, where $\alpha \geq \varepsilon/\mathrm{polylog}(1/\varepsilon)$, then* $\texttt{Sample-Splittable}_k(f,\varepsilon)$ *finds a length-k monotone subsequence of f with probability at least $9/10$. The query complexity of* $\texttt{Sample-Splittable}_k(f,\varepsilon)$ *is at most*

$$\frac{1}{\varepsilon}\left(\frac{\log n}{\varepsilon}\right)^{\lfloor \log_2 k \rfloor} \cdot \mathrm{polylog}(1/\varepsilon).$$

Proof of Theorem III.1 assuming Lemmas III.1 and III.2: The algorithm $\texttt{Sampler}_k(f,\varepsilon)$ executes both $\texttt{Sample-Suffix}_k(f,\varepsilon)$ and $\texttt{Sample-Splittable}_k(f,\varepsilon)$; if either algorithm finds a length-k monotone subsequence of f, output such a subsequence. We note that by Theorem II.3, either case 1, or case 2 holds. If case 1 holds, then by Lemma III.1, $\texttt{Sample-Suffix}(f,\varepsilon)$ outputs a length-k monotone subsequence with probability at least $9/10$, and if case 2 holds, then by Lemma III.2, $\texttt{Sample-Splittable}_k(f,\varepsilon)$ outputs a length-k monotone subsequence with probability at least $9/10$. Thus, regardless of which case holds, a length-k monotone subsequence will be found with probability at least $9/10$. The query complexity then follows from the maximum of the two query complexities. ∎

B. Proof of Lemma III.1: an algorithm for growing suffixes

We now prove Lemma III.1. Let $C > 0$ be a large constant, and let $k \in \mathbb{N}$ be fixed. Let $\varepsilon > 0$ and $f\colon [n] \to \mathbb{R}$ be a function which is ε-far from $(12\ldots k)$-free. Furthermore, as per the assumption of case 1 of the algorithm, we assume that there exists a parameter $\alpha \in (0,1)$ as well as a set $H \subseteq [n]$ of $(\alpha, Ck\alpha)$-growing suffixes, where $\alpha|H| \geq \varepsilon n/\mathrm{polylog}(1/\varepsilon)$.

The algorithm, which underlies the result of Lemma III.1, proceeds by sampling uniformly at random an index $\boldsymbol{a} \sim [n]$, and running a sub-routine which we call $\texttt{Growing-Suffix}$, with $\boldsymbol{a}$ as input. The sub-routine is designed so that if $\boldsymbol{a}$ is the start of an $(\alpha, Ck\alpha)$-growing suffix then the algorithm will find a length-k monotone subsequence of f with probability at least $99/100$. The subroutine, $\texttt{Growing-Suffix}$, is presented in Figure 9.

[15]in this case the $\mathrm{polylog}(1/\varepsilon)$ term stands for $(k\log(1/\varepsilon))^{O(k^2)}$

> Subroutine $\texttt{Growing-Suffix}(f,\alpha_0,a)$
>
> **Input:** Query access to a function $f\colon [n] \to \mathbb{R}$, a parameter $\alpha_0 \in (0,1)$, and an index $a \in [n]$.
> **Output:** a subset of k indices $i_1 < \cdots < i_k$ where $f(i_1) < \cdots < f(i_k)$, or fail.
>
> 1) Let $\eta_a = \lceil \log(n-a) \rceil$ and consider the sets $S_j(a) = (a + \ell_{j-1}, a + \ell_j] \cap [n]$ for all $j \in [\eta_a]$ and $\ell_j = 2^j$.
> 2) For each $j \in [\eta_a]$, let $\mathbf{A}_j \subseteq S_j(a)$ be obtained by sampling uniformly at random $T := 1/\alpha_0$ times from $S_j(a)$.
> 3) For each $j \in [\eta_a]$ and each $b \in \mathbf{A}_j$, query $f(b)$.
> 4) If there exist indices $i_1, \ldots, i_k \in \mathbf{A}_1 \cup \cdots \cup \mathbf{A}_{\eta_i}$ satisfying $i_1 < \cdots < i_k$ and $f(i_1) < \cdots < f(i_k)$, **return** such indices $i_1, \ldots, i_k$. Otherwise, **return** fail.

Figure 9. Description of the $\texttt{Growing-Suffix}$ subroutine.

Lemma III.3. *Let $f\colon [n] \to \mathbb{R}$ be a function, let $\alpha, \alpha_0, \beta \in (0,1)$ be parameters satisfying $\beta \geq Ck\alpha$ and $\alpha_0 \leq \alpha$, and suppose that $a \in [n]$ starts a (α,β)-growing suffix in f. Then* $\texttt{Growing-Suffix}(f,\alpha_0,a)$ *finds a length-k monotone subsequence of f with probability at least $99/100$.*

Proof: Recall, from Definition II.4, that if $a \in [n]$ is the start of a (α,β)-growing suffix of f then there exist a collection of sets, $D_1(a), \ldots, D_{\eta_a}(a)$ and parameters $\delta_1(a), \ldots, \delta_{\eta_a}(a) \in (0,\alpha]$, where every $j \in [\eta_a]$ has $D_j(a) \subseteq S_j(a)$, $|D_j(a)| = \delta_j(a) \cdot |S_j(a)|$, and $\sum_{j=1}^{\eta_a} \delta_j(a) \geq \beta$. Further, if, for some $j_1, \ldots, j_k \in [\eta_i]$, we have $j_1 < \cdots < j_k$ and for all $\ell \in [k]$, $\mathbf{A}_{j_\ell} \cap D_{j_\ell}(a) \neq \emptyset$, then the union $D_{j_1}(a) \cup \ldots \cup D_{j_k}(a)$ contains a length-k monotone subsequence. In view of this, for each $j \in [\eta_a]$, consider the indicator random variable

$$\mathbf{E}_j := \mathbf{1}\{\mathbf{A}_j \cap D_j(a) \neq \emptyset\},$$

and observe that by the foregoing discussion $\texttt{Growing-Suffix}(f,\alpha_0,a)$ samples a length-k monotone subsequence of f whenever $\sum_{j=1}^{\eta_a} \mathbf{E}_j \geq k$. We note that the $\mathbf{E}_j$'s are independent, and that

$$\mathbf{Pr}[\mathbf{E}_j = 1] = 1 - (1 - \delta_j(a))^T \geq \min\left\{\frac{T \cdot \delta_j(a)}{10}, \frac{1}{10}\right\}.$$

Let $J \subseteq [\eta_a]$ be the set of indices satisfying $T \cdot \delta_j(a) \geq 1$ (recall that $T = 1/\alpha_0$). Then, if $|J| \geq Ck$ we have

$$\mathbb{E}\left[\sum_{j=1}^{\eta_a} \mathbf{E}_j\right] \geq \frac{Ck}{10},$$

since every variable $j \in J$ contributes at least $1/10$. On the other hand, if $|J| \leq Ck/2$, then, since $\delta_j(a) \leq \alpha$ for every

j, we have $\sum_{j\in[\eta_a]\setminus J}\delta_j(a) \geq \beta - |J|\cdot\alpha \geq \beta/2$ (using $\beta \geq Ck\alpha$) so that

$$\mathbb{E}\left[\sum_{j=1}^{\eta_a}\mathbf{E}_j\right] \geq \mathbb{E}\left[\sum_{j\in[\eta_a]\setminus J}\mathbf{E}_j\right] \geq \frac{T}{10}\cdot\frac{\beta}{2} \geq \frac{Ck}{20}.$$

In either case, $\mathbb{E}[\sum_{j\in[\eta_a]}\mathbf{E}_j] \geq Ck/20$, and since the events $\mathbf{E}_i$ are independent, via a Chernoff bound we obtain that $\sum_j \mathbf{E}_j$ is larger than k with probability at least $99/100$. ■

Subroutine `Sample-Suffix`$_k\,(f,\varepsilon)$

Input: Query access to a function $f\colon [n] \to \mathbb{R}$, and a parameter $\varepsilon \in (0,1)$.
Output: a subset of k indices $i_1 < \cdots < i_k$ where $f(i_1) < \cdots < f(i_k)$, or fail.

1) Repeat the following for all $j = 1,\ldots,O(\log(1/\varepsilon))$, letting $\alpha_j = 2^{-j}$:
   - For $t_j = \alpha_j \cdot \mathrm{polylog}(1/\varepsilon)/\varepsilon$ iterations, sample $\boldsymbol{a} \sim [n]$ uniformly at random and run `Growing-Suffix`$(f,\alpha_j,\boldsymbol{a})$, and if it returns a length-k monotone subsequence of f, **return** that subsequence.
2) If the algorithm has not already output a monotone subsequence, **return** fail.

Figure 10. Description of the `Sample-Suffix` subroutine.

With this in hand, we can now establish Lemma III.1.

Proof of Lemma III.1: First, note that the query complexity of `Sample-Suffix`$_k(f,\varepsilon)$ is

$$\sum_{j=1}^{O(\log(1/\varepsilon))} t_j\cdot O(\log n/\alpha_j) = \frac{\log n\cdot \mathrm{polylog}(1/\varepsilon)}{\varepsilon}.$$

Consider the iteration of j where $\alpha_j \leq \alpha \leq 2\alpha_j$ (note that since $\alpha \geq \varepsilon/\mathrm{polylog}(1/\varepsilon)$, there exists such j). Then, since $|H| \geq \varepsilon/(\alpha\cdot\mathrm{polylog}(1/\varepsilon))$, we have that $t_j \geq Cn/|H|$ (for a sufficiently large constant C). Thus, with probability at least $99/100$, some iteration satisfies $\boldsymbol{a} \in H$. When this occurs, `Growing-Suffix`$(f,\alpha_j,\boldsymbol{a})$ will output a length-k monotone subsequence with probability at least $99/100$, by Lemma III.3, and thus by a union bound we obtain the desired result. ■

C. Proof of Lemma III.2: an algorithm for splittable intervals

We now prove Lemma III.2. We consider a fixed setting of $k \in \mathbb{N}$ and $\varepsilon > 0$, and let $f\colon [n] \to \mathbb{R}$ be any sequence which is ε-far from being $(12\ldots k)$-free. Furthermore, as per case 2 of the algorithm, we assume that there exists a set $T \subseteq [n]^k$ of disjoint length-k monotone subsequences of f where

$$|T| \geq \frac{\varepsilon n}{\mathrm{polylog}(1/\varepsilon)},$$

and (G,ϱ,I) is a (k,k,α)-tree descriptor which represents $(f,T,[n])$, where $\alpha \geq \varepsilon/\mathrm{polylog}(1/\varepsilon)$. In what follows, we describe a sub-routine, `Sample-Splittable`$_k(f,\varepsilon)$ in terms of two parameters $\rho, q \in \mathbb{R}$. The parameter $\rho > 0$ is set to be sufficiently large and independent of n, satisfying

$$\rho \geq \frac{\varepsilon}{\mathrm{polylog}(1/\varepsilon)}. \tag{15}$$

One property which we will want to satisfy is that if we take a random subset of $[n]$ by including each element independently with probability $1/(\rho n)$, we will include an element belonging to $E(T)$ with probability at least $1-1/(Ck)$, for a large constant $C > 0$. The parameter q will be an upper bound on the query complexity of the algorithm, which we set to a high enough value satisfying: $q = O\left(\frac{1}{\rho}\left(\frac{\log n}{\rho}\right)^{\lfloor\log_2 k\rfloor}\right) \leq \frac{1}{\varepsilon}\cdot\left(\frac{\log n}{\varepsilon}\right)^{\lfloor\log_2 k\rfloor}\cdot \mathrm{polylog}(1/\varepsilon)$.

Subroutine `Sample-Splittable`$_k\,(f,\varepsilon)$

Input: Query access to a sequence $f\colon [n] \to \mathbb{R}$, and a parameter $\varepsilon \in (0,1)$.
Output: a subset of k indices $i_1 < \cdots < i_k$ where $f(i_1) < \cdots < f(i_k)$, or fail.

1) Let $r = \lfloor\log_2 k\rfloor$ and run `Sample-Helper`$(r,[n],\rho)$, to obtain a set $\mathbf{A} \subseteq [n]$.
2) If $|\mathbf{A}| > q$, **return** fail; otherwise, for each $a \in \mathbf{A}$, query $f(a)$. If there exists a monotone sequence of f of length k, then **return** that subsequence. If not, **return** fail.

Figure 11. Description of the `Sample-Splittable` subroutine.

Subroutine `Sample-Helper`$\,(r,I,\rho)$

Input: An integer $r \in \mathbb{N}$, an interval $I \subseteq [n]$, and a parameter $\rho \in (0,1)$.
Output: a subset of $A \subseteq I$.

1) Let $\mathbf{A}_0 = \emptyset$. For every index $a \in I$, let $\mathbf{A}_0 \leftarrow \mathbf{A}_0 \cup \{a\}$ with probability $1/(\rho|I|)$.
2) If $r = 0$, **return** $\mathbf{A}_0$.
3) If $r > 0$, proceed with the following:
   - For every index $a \in \mathbf{A}_0$, consider the $O(\log n)$ intervals given by $B_{a,j} = [a-\ell_j, a+\ell_j]$, for $j = 1,\ldots,O(\log n)$ and $\ell_j = 2^j$, and let $\mathbf{R}_{a,j} \leftarrow$ `Sample-Helper`$(r-1, B_{a,j}, \rho)$.
   - Let $\mathbf{A}$ be the set
   $$\mathbf{A} \leftarrow \bigcup_{a\in\mathbf{A}_0,\, j=O(\log n)} \mathbf{R}_{a,j}.$$
   - **return** the set $(\mathbf{A}_0 \cup \mathbf{A}) \cap I$.

Figure 12. Description of the `Sample-Helper` subroutine.

The descriptions of the main algorithm `Sample-Splittable`$_k$ and the sub-routine `Sample-Helper`, are given in Figure 11 and

Figure 12. Note that, for any $r \in \mathbb{N}$, if we let $\mathcal{D}_r$ be the distribution of $|\mathbf{A}|$, where $\mathbf{A}$ is the output of a call to `Sample-Helper`$(r, [n], \rho)$. Then, we have that $\mathcal{D}_0 = \mathrm{Bin}(n, \rho)$, and for $r > 0$, $\mathcal{D}_r$ is stochastically dominated by the random variable

$$\sum_{i=1}^{\boldsymbol{y}_0} \sum_{j=1}^{O(\log n)} \boldsymbol{x}_{r-1}^{(i,j)},$$

where $\boldsymbol{y}_0 \sim \mathrm{Bin}(n, 1/(\rho n))$ and $\boldsymbol{x}_{r-1}^{(i,j)} \sim \mathcal{D}_{r-1}$ for all $i \in \mathbb{N}$ and $j \in [O(\log n)]$ are all mutually independent. As a result, for $r \geq 1$,

$$\mathbb{E}[|\mathbf{A}|] \leq \frac{1}{\rho} \cdot \log n \cdot \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{D}_{r-1}} [\boldsymbol{x}],$$

and since $\mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_0}[\boldsymbol{x}] = 1/\rho$, we have:

$$\mathbb{E}[|\mathbf{A}|] \leq \frac{1}{\rho} \left(\frac{\log n}{\rho} \right)^r.$$

We may then apply Markov's inequality to conclude that $|\mathbf{A}| \leq q$ with probability at least $99/100$. As a result, we focus on proving that the probability that the set $\mathbf{A}$ contains a monotone subsequence of f of length k is at least $99/100$. This would imply the desired result by taking a union bound.

In addition to the above, we define another algorithm, `Sample-Helper`*, in Figure 13, which will be a *helper* sub-routine. We emphasize that `Sample-Helper`* is not executed in the algorithm itself, but will be useful in order to analyze `Sample-Helper`.

Subroutine `Sample-Helper`* $(r, I, \rho, \mathcal{I})$

Input: An integer $r \in \mathbb{N}$, an interval $I \subseteq [n]$, a parameter $\rho \in (0, 1)$, and a collection of disjoint intervals $\mathcal{I}$ of $[n]$.
Output: two subsets $\mathbf{A}, \mathbf{A}_0 \subseteq I$.

1) Let $\mathbf{A}_0 = \emptyset$. For every index $a \in I$ which lies inside an interval in $\mathcal{I}$, let $\mathbf{A}_0 \leftarrow \mathbf{A}_0 \cup \{a\}$ with probability $1/(\rho|I|)$.
2) If $r = 0$, **return** $\mathbf{A}_0$.
3) If $r > 0$, proceed with the following:
   - For every index $a \in \mathbf{A}_0$, consider the $O(\log n)$ intervals given by $B_{a,j} = [a - \ell_j, a + \ell_j]$, for $j = 1, \ldots O(\log n)$, and $\ell_j = 2^j$, and let $(\mathbf{R}_{a,j}, \mathbf{R}_{a,j,0}) \leftarrow$ `Sample-Helper`$^*(r - 1, B_{a,j}, \rho, \mathcal{I})$.
   - Let $\mathbf{A}$ to be the set
   $$\mathbf{A} \leftarrow \bigcup_{a \in \mathbf{A}_0,\ j = O(\log n)} \mathbf{R}_{a,j}.$$
   - **return** the set $(\mathbf{A} \cap I, \mathbf{A}_0 \cap I)$.

Figure 13. Description of the `Sample-Helper`* subroutine.

Before proceeding, we require a "coupling lemma." Its main purpose is to prove the intuitive fact that if $\mathcal{I}_0, \mathcal{I}_1$ are collections of disjoint intervals, and the latter is a refinement of the former (namely, each intervals in $\mathcal{I}_1$ is contained in an interval of $\mathcal{I}_0$), then `Sample-Helper`$^*(r, [n], \rho, \mathcal{I}_0)$ is more likely to find a length-k monotone subsequence than `Sample-Helper`$^*(r, [n], \rho, \mathcal{I}_1)$ does.

Lemma III.4. *Let $r \in \mathbb{N}$ be an integer, $f \colon [n] \to \mathbb{R}$ a function, $\rho \in (0, 1)$ a parameter, and $\mathcal{I}_0$ and $\mathcal{I}_1$ collections of disjoint intervals in $[n]$, such that each interval in $\mathcal{I}_1$ lies inside an interval from $\mathcal{I}_0$. Denote by $(\mathbf{A}^{(i)}, \mathbf{A}_0^{(i)})$ the random pair of sets given by the output of* `Sample-Helper`$^*(r, [n], \rho, \mathcal{I}_i)$*, for $i = 0, 1$. Lastly, let $\mathcal{E} \colon \mathcal{P}([n]) \times \mathcal{P}([n]) \to \{0, 1\}$ be any* monotone *function; that is, it satisfies $\mathcal{E}(S_1, S_2) \leq \mathcal{E}(S_1', S_2')$ for any $S_1 \subseteq S_1' \subseteq [n]$ and $S_2 \subseteq S_2' \subseteq [n]$. Then,*

$$\mathbf{Pr}[\mathcal{E}(\mathbf{A}^{(0)}, \mathbf{A}_0^{(0)}) = 1] \geq \mathbf{Pr}[\mathcal{E}(\mathbf{A}^{(1)}, \mathbf{A}_0^{(1)}) = 1].$$

Proof: Consider an execution of `Sample-Helper`$^*(r, [n], \rho, \mathcal{I}_0)$ which outputs a pair $(\mathbf{A}^{(0)}, \mathbf{A}_0^{(0)})$. Let $\mathbf{A}^{(1)}$ and $\mathbf{A}^{(1)}$ be the subsets of $\mathbf{A}^{(0)}$ and $\mathbf{A}^{(0)}$, respectively, obtained by running a parallel execution of `Sample-Helper`$^*(r, [n], \rho, \mathcal{I}_1)$, which follows the execution of `Sample-Helper`$^*(r, [n], \rho, \mathcal{I}_0)$, but whenever an element which is not in an interval of $\mathcal{I}_1$ is considered, it is simply ignored (i.e. it is not included in $\mathbf{A}^{(0)}$ or in $\mathbf{A}_0^{(0)}$ and no recursive calls based on such elements are made). It is easy to see that this coupling yields a pair $(\mathbf{A}^{(1)}, \mathbf{A}_0^{(1)})$ with the same distribution as that given by running `Sample-Helper`$^*(r, [n], \rho, \mathcal{I}_1)$. As $\mathcal{E}(\cdot, \cdot)$ is increasing, if $\mathcal{E}(\mathbf{A}^{(0)}, \mathbf{A}_0^{(0)})$ holds then so does $\mathcal{E}(\mathbf{A}^{(1)}, \mathbf{A}_0^{(1)})$. The lemma follows. ■

The following corollary is a direct consequence of Lemma III.4. Specifically, we use the facts that `Sample-Splittable`$_k(f, \varepsilon)$ calls `Sample-Helper`$(\lfloor \log_2 k \rfloor, [n], \rho)$, which is equivalent to calling `Sample-Helper`$(\lfloor \log_2 k \rfloor, [n], \rho, \{[n]\})$, and that finding a $(12 \ldots k)$-pattern in $\mathcal{I}$ is a monotone event.

Corollary III.5. *Let $\mathcal{I}$ be any collection of disjoint intervals in $[n]$. Suppose $(\mathbf{A}, \mathbf{A}_0)$ is the random pair of sets given by the output of* `Sample-Helper`$^*(\lfloor \log_2 k \rfloor, n, \rho, \mathcal{I})$*, then,*

$$\mathbf{Pr}\left[\substack{\texttt{Sample-Splittable}_k(f,\varepsilon) \textit{ finds a} \\ (12\ldots k)\textit{-pattern of } f}\right] \geq \mathbf{Pr}[\mathbf{A} \textit{ contains a } (12 \ldots k)\textit{-pattern in } f_{|\mathcal{I}}].$$

Definition III.6. *Let $k_0 \in \mathbb{N}$ be a positive integer, and let (G, ϱ) be a k_0-tree descriptor (for this definition we do not care about the third component of the descriptor,* l*). We say that $p \in [k_0]$ is the* primary index *of (G, ϱ) if the leaf with rank p under $\leq_G$ is the unique leaf whose root-to-leaf path $(u_1, \ldots, u_d)$ satisfies the following: for each $d' \in [d-1]$, denoting the left and right children of $u_{d'}$ by v_l and v_r, respectively, $u_{d'+1}$ is v_l if the number of leaves in the subtree rooted at v_l is at least the number of leaves in the subtree rooted at v_r, and otherwise, $u_{d'+1}$ is v_r.*

From Corollary III.5, we note that Lemma III.2 follows from the following lemma.

Lemma III.7. *Let $k, k_0, n \in \mathbb{N}$ satisfy $1 \le k_0 \le k$, let C be a large enough constant, and let $\alpha, \rho \in (0,1)$ be such that $\rho \ge C\alpha$ and $\alpha \ge \rho/\mathrm{polylog}(1/\rho)$. Let $f\colon [n] \to \mathbb{R}$ be a function, let $\mathcal{I}$ be a collection of disjoint intervals in $[n]$, for each $I \in \mathcal{I}$ let $T_I \subseteq I^{k_0}$ be a set of disjoint, length-k_0 monotone subsequence of f, and suppose that*

$$\sum_{I\in\mathcal{I}} |T_I| \ge \alpha n/4.$$

Suppose that (G,ϱ) is a (k,k_0,α)-weighted-tree such that for every $I \in \mathcal{I}$ there exists a function $\mathsf{I}_I : V(G) \to \mathcal{S}(I)$, such that (G,ϱ,I_I) is a tree descriptor that represents (f, T_I, I). Given any $r \in \mathbb{N}$ satisfying $\lfloor \log_2 k_0 \rfloor \le r$, let $(\mathbf{A}, \mathbf{A}_0)$ be the pair of sets output by the sub-routine `Sample-Helper`$^*(r,[n],\rho,\mathcal{I})$. *With probability at least $1 - k_0/(100k)$, there exist indices $i_1,\ldots,i_{k_0} \in [n]$ with the following properties.*

1) *$(i_1,\ldots,i_{k_0})$ is a length-k_0 monotone subsequence of f.*
2) *There is an interval $I \in \mathcal{I}$ such that $i_1,\ldots,i_{k_0} \in I \cap E(T_I)$.*
3) *$i_1,\ldots,i_{k_0} \in \mathbf{A}$ and $i_p \in \mathbf{A}_0$, where p is the primary index of (G,ϱ).*

Proof:

The proof proceeds by induction on k_0. Consider the base case, when $k_0 = 1$. In this case, $\lfloor \log_2 k_0 \rfloor = 0$, so for any $r \ge 0$, `Sample-Helper`$^*(r,[n],\rho,\mathcal{I})$ runs step 1. As a result, `Sample-Helper`$^*(r,[n],\rho,\mathcal{I})$ samples each element inside an interval of $\mathcal{I}$ independently with probability $1/(\rho n)$. In order to satisfy the requirements of the lemma in this case, we need $\mathbf{A}_0$ to contain an element of $\cup_{I\in\mathcal{I}} T_I$. By the assumption on the size of this union, and because each of the elements of the union lives inside some interval from $\mathcal{I}$, such an element will exist with sufficiently high probability via a Chernoff bound.

For the inductive step, assume that Lemma III.7 is fulfilled whenever $k_0 < K$, for $K \in \mathbb{N}$ satisfying $1 < K \le k$, and we will prove, assuming this inductive hypothesis, that Lemma III.7 holds for $k_0 = K$. So consider a setting $k_0 = K$. Let $\mathcal{I}$, (G,ϱ) and I_I be as in the statement of the lemma. Denote the root of (G,ϱ) by v_{root}, and its left and right children by v_{left} and v_{right}. Let c be the number of leaves in the subtree $(G_{\mathsf{left}}, \varrho_{\mathsf{left}})$ rooted at v_{left}, so $k_0 - c$ is the number of leaves in the subtree $(G_{\mathsf{right}}, \varrho_{\mathsf{right}})$ rooted at v_{right}. We shall assume that $c \ge k_0 - c$; the other case follows by an analogous argument.

For each $I \in \mathcal{I}$, the collection of pairs $(J, T_{I,J})$, where $J \in \mathsf{I}_I(v_{\mathsf{root}})$ and $T_J = T_I \cap J^{k_0}$ is the restriction of T_I to J, is a $(c, 1/(6k), \alpha)$-splittable collection of I. Let $\mathcal{J}$ be the collection of all such intervals J (note that they are pairwise disjoint and that $\mathcal{J}$ is a refinement of $\mathcal{I}$). Let (L_J, M_J, R_J) be the partition of J into left, middle and right intervals, respectively, and let $T_J^{(L)}$ and $T_J^{(R)}$ be sets of c-prefixes and $(k_0 - c)$-suffixes of k_0-tuples from $T_{I,J}$, as given by Definition II.5. Set $\mathcal{L} = \{L_J : J \in \mathcal{J}\}$, $\mathcal{R} = \{R_J : J \in \mathcal{J}\}$, $T^{(L)} = \bigcup_{J\in\mathcal{J}} T_J^{(L)}$, and $T^{(R)} = \bigcup_{J\in\mathcal{J}} T_J^{(R)}$. Note that $(G_{\mathsf{left}}, \varrho_{\mathsf{left}}, \mathsf{I}_{J,\mathsf{left}})$ is a (k,c,α)-tree descriptor for (f, T_J, J), with appropriate $\mathsf{I}_{J,\mathsf{left}}$. Similarly, $(G_{\mathsf{right}}, \varrho_{\mathsf{right}}, \mathsf{I}_{J,\mathsf{right}})$ is a $(k, k_0 - c, \alpha)$-tree descriptor for (f, T_J, J), with appropriate $\mathsf{I}_{J,\mathsf{right}}$.

We consider an execution of `Sample-Helper`$^*(r,[n],\rho,\mathcal{I})$ which outputs a random pair of sets $(\mathbf{A}, \mathbf{A}_0)$. Let $\mathbf{A}^{(L)}$ and $\mathbf{A}_0^{(L)}$ be the subsets of $\mathbf{A}$ and $\mathbf{A}_0$, respectively, obtained by running a parallel execution of `Sample-Helper`$^*(r,[n],\rho,\mathcal{L})$, where, as in the proof of Lemma III.4, we follow the execution of `Sample-Helper`$^*(r,[n],\rho,\mathcal{I})$, but whenever an element which is not in $\mathcal{L}$ is considered, we ignore it. As stated above, this coupling yields a pair $(\mathbf{A}^{(L)}, \mathbf{A}_0^{(L)})$ with the distribution given by running `Sample-Helper`$^*(r,[n],\rho,\mathcal{L})$.

For $a \in \mathbf{A}_0^{(L)}$, and any $j \in [O(\log n)]$, let $(\mathbf{A}^{(a,j)}, \mathbf{A}_0^{(a,j)})$ be the output of the recursive call (inside the execution of `Sample-Helper`$^*(r,[n],\rho,\mathcal{I})$) of `Sample-Helper`$^*(r-1, B_{a,j}, \rho, \mathcal{R})$.

We define the collection:

$$\mathcal{S} = \left\{ (S_0, S) : \substack{S_0 \subseteq S \subseteq E(T^{(L)}),\\ \exists i_1,\ldots,i_c \in S,\ \text{a } (12\ldots c)\text{-pattern with } i_p \in S_0,\\ \exists J \in \mathcal{J} \text{ s.t. } i_1,\ldots,i_c \in L_J} \right\}.$$

For each $(S_0, S) \in \mathcal{S}$, we let $\mathsf{a}(S_0,S) \in E(T^{(L)})$ be some $i_p \in S$ such that there exist $c-1$ indices $i_1,\ldots,i_{p-1}, i_{p+1}, i_c$, such that $(i_1,\ldots,i_c)$ forms a $(12\ldots c)$-pattern in S, and $i_1,\ldots,i_p \in L_J$ for some $J \in \mathcal{J}$. Let $\mathtt{seg}(S_0,S)$ be this interval J, and let $\mathtt{len}(S_0,S) \in [O(\log n)]$ be the smallest j for which $R_J \subseteq B_{a,j}$, where $a = \mathsf{a}(S_0,S)$.

Let $\mathbf{E}_L$ be the event that

$$\left(\mathbf{A}^{(L)} \cap E(T^{(L)}), \mathbf{A}_0^{(L)} \cap E(T^{(L)})\right) \in \mathcal{S},$$

and let $\mathbf{E}_L(S_0,S)$ be the event that

$$\mathbf{A}^{(L)} \cap E(T^{(L)}) = S_0 \qquad \mathbf{A}_0^{(L)} \cap E(T^{(L)}) = S,$$

so $\mathbf{E}_L = \cup_{(S_0,S)\in\mathcal{S}} \mathbf{E}_L(S_0,S)$, and the events $\mathbf{E}_L(S_0,S)$ are pairwise disjoint.

By the induction hypothesis, applied with the family $\{L_J : J \in \mathcal{J}\}$ and the corresponding sets $T_J^{(L)}$ (using $\sum_{J\in\mathcal{J}} |T_J^{(L)}| = \sum_{J\in\mathcal{J}} |T_J| \ge \alpha n/4$), we have

$$\mathbf{Pr}[\mathbf{E}_L] \ge 1 - c/(100k).$$

Let $\mathbf{E}_R(a,j)$ be the event that $a \in \mathbf{A}_0$, and in the recursive run of `Sample-Helper`$^*(r-1, B_{a,j}, \rho, \mathcal{R})$ inside `Sample-Helper`$^*(r,[n],\rho,\mathcal{I})$, there exist indices $i'_1,\ldots,i'_{k_0-c}$ such that

- $(i'_1,\ldots,i'_{k_0-c})$ form a length $(k_0 - c)$-monotone subsequence.

- $i'_1, \ldots, i'_{k_0-c} \in E(T_J^{(R)})$, where J is the interval in $\mathcal{J}$ with $i \in J$.
- $i'_1, \ldots, i'_{k_0-c} \in \mathbf{A}^{(a,j)}$ and $i'_q \in \mathbf{A}_0^{(a,j)}$, where q is the primary index of $(G_{\mathsf{right}}, \varrho_{\mathsf{right}})$.

Let $\mathbf{F}_R(a,j)$ be the event that in a run of $\texttt{Sample-Helper}^*(r-1, B_{a,j}, \rho, \mathcal{R})$, there exist $i'_1, \ldots, i'_{k_0-c}$ as above. Fix some $(S_0, S) \in \mathcal{S}$, and let $a = \mathsf{a}(S_0,S)$, $J = \texttt{seg}(S_0,S)$ and $j = \texttt{len}(S_0,S)$. We claim that

$$\mathbf{Pr}[\mathbf{E}_R(a,j) \mid \mathbf{E}_L(S_0,S)] = \mathbf{Pr}[\mathbf{F}_R(a,j)].$$

Indeed, by conditioning on $\mathbf{E}_L(S_0,S)$ we know that $a \in \mathbf{A}_0$, so there will be a recursive run of $\texttt{Sample-Helper}^*(r-1, B_{a,j}, \rho, \mathcal{R})$, and moreover the event $\mathbf{E}_L(S_0,S)$ will have no influence on the outcomes of this run.

Note that $|T_J^{(R)}| \geq \alpha|R_J| \geq \alpha|B_{a,J}|/4$. By the induction hypothesis, applied with the interval $B_{a,J}$ in place of $[n]$, the family $\{R_J\}$ and the corresponding set $T_J^{(R)}$, and the tree $(G_{\mathsf{right}}, \varrho_{\mathsf{right}})$, we find that $\mathbf{Pr}[\mathbf{F}_R(a,j)] \geq 1 - (k_0 - c)/(100k)$. We note that if both $\mathbf{E}_L(S_0,S)$ and $\mathbf{E}_R(a,j)$ hold, then there are indices $i_1, \ldots, i_c, i'_1, \ldots, i'_{k_0-c}$ such that

- $(i_1, \ldots, i_c)$ is a length-c monotone subsequence in $E(T_J^{(L)})$, and $(i'_1, \ldots, i'_{k_0-c})$ is a length-c monotone subsequence in $E(T_J^{(R)})$. In particular, $(i_1, \ldots, i_c, i'_1, \ldots, i'_{k_0-c})$ is a length-k_0 monotone subsequence that lies in $E(T_j)$.
- $i_1, \ldots, i_c, i'_1, \ldots, i'_{k_0-c} \in \mathbf{A}$ and $i_p \in \mathbf{A}_0$ (recall that p is the primary index of both G and G_{left}).

I.e. if these two events hold, then the requirements ot the lemma are satisfied. It follows that the requirements ot the lemma are satisfied with at least the following probability, using the fact that the events $\mathbf{E}_L(S_0,S)$ are disjoint.

$$\begin{aligned}
&\sum_{(S_0,S)\in\mathcal{S}} \mathbf{Pr}[\mathbf{E}_R(\mathsf{a}(S_0,S), \texttt{len}(S_0,S)) \text{ and } \mathbf{E}_L(S_0,S)] \\
&\geq \sum_{(S_0,S)\in\mathcal{S}} \mathbf{Pr}[\mathbf{F}_R(\mathsf{a}(S_0,S), \texttt{len}(S_0,S))]\, \mathbf{Pr}[\mathbf{E}_L(S_0,S)] \\
&\geq \left(1 - \frac{k_0 - c}{100k}\right) \cdot \sum_{(S_0,S)\in\mathcal{S}} \mathbf{Pr}[\mathbf{E}_L(S_0,S)] \\
&\geq \left(1 - \frac{k_0 - c}{100k}\right) \cdot \mathbf{Pr}[\mathbf{E}_L] \\
&\geq \left(1 - \frac{k_0 - c}{100k}\right) \cdot \left(1 - \frac{c}{100k}\right) \geq 1 - \frac{k_0}{100k}.
\end{aligned}$$

This completes the proof of Lemma III.7. ■

IV. Lower Bounds

In this section, we prove our lower bound for non-adaptive testing of $(12\ldots k)$-freeness with one-sided error, Theorem I.2. Below we give a precise quantitative version of our lower bound statement for the case where k and n are both a power of 2, from which one can derive the general case, as we shall explain soon.

Theorem IV.1. *Let $k \leq n \in \mathbb{N}$ be powers of 2 and let $0 < p < 1$. There exists a constant $\varepsilon_0 > 0$ such that any non-adaptive algorithm which, given query access to a function $f\colon [n] \to \mathbb{R}$ that is ε_0-far from $(12\ldots k)$-free, outputs a length-k monotone subsequence with probability at least p, must make at least $p\binom{\log_2 n}{\log_2 k}$ queries. Moreover, one can take $\varepsilon_0 = 1/k$.*

As is usual for arguments of this type, to prove Theorem IV.1 we follow Yao's minimax principle [40]. We construct a distribution $\mathcal{D}_{n,k}$ over sequences that are $(1/k)$-far from $(12\ldots k)$-free, such that any deterministic algorithm, that makes fewer than $p\binom{\log_2 n}{\log_2 k}$ queries, fails to find a $(12\ldots k)$-copy in a sequence drawn from this distribution, with probability larger than $1-p$. Here, a deterministic non-adaptive algorithm that makes q queries amounts to deterministically picking a q-element subset Q of $[n]$ in advance (without seeing any values in the sequence), and querying all elements of Q.

Handling general k and n.: We first explain how to prove our general lower bound, Theorem I.2, using the lower bound distribution $\mathcal{D}_{n,k}$ of the case where n and k are powers of 2, given in Theorem IV.1, as a black box. The reduction relies on standard "padding" techniques. Given integers k,n with $k \leq n$, write $k = 2^h + t$ for $h,t \in \mathbb{N}$ with $t < 2^h$, and let $k' = 2^h$. Let n' be the largest power of 2 which is not larger than nk'/k, and note that $n' \geq n/4$ and $k' \leq n'$. We construct our lower bound distribution $\mathcal{D}_{n,k}$ as follows. Given any $f'\colon [n'] \to \mathbb{R}$ in $\mathcal{D}_{n',k'}$, we partition the set $\{n'+1, n'+2, \ldots, n\}$ into t consecutive intervals $I_1, \ldots, I_t$, each of size at least n'/k', and extend f' to a sequence $f\colon [n] \to \mathbb{R}$ satisfying the following conditions.

- $f(x) = f'(x)$ for any $x \in [n']$.
- f is decreasing within any I_i, that is, $f(x) > f(y)$ for $x < y \in I_i$.
- $f(x) < f(y)$ for any $x \in [n']$ and $y \in I_1$, and for any $x \in I_i$ and $y \in I_j$ where $i < j$.

Clearly, we can construct such a sequence f from any given sequence f'. Moreover, it is possible to make sure that the values $f(x)$ with $x \in [n]$ are distinct, and thus by relabeling f can be taken to be a permutation. Furthermore, any $(12\ldots k')$-copy in f' can be extended to a $(12\ldots k)$-copy in f by appending exactly one arbitrary element from each I_i to it, for a total of $t = k - k'$ additional elements.

Building on the fact that f' is $(1/k')$-far from $(12\ldots k')$-free and that $n' \geq n/4$ and $n - n' \geq n(k-k')/k$, we conclude that f is $(1/4k')$-far from $(12\ldots k)$-free. Form a distribution $\mathcal{D}_{n,k}$ by picking a random $\boldsymbol{f}$ according to the distribution $\sim \mathcal{D}_{n',k'}$ and extending it to a sequence f' as above.

The rest of this section is devoted to the proof of Theorem IV.1.

A. Basic binary profiles and monotonicity testing

In a sense, the proof of our lower bound, Theorem IV.1, is a (substantial) generalization of the non-adaptive lower bound for testing monotonicity. In order to introduce the machinery required for the proof, we present, in this subsection, a simple proof of the classical $\Omega(\log n)$ non-adaptive one-sided lower bound for monotonicity testing [12] using basic versions of the tools we shall use for the full proof. Then, in Subsection IV-B we proceed to present our tools in their full generality, and provide the proof of Theorem IV.1.

Intuitively, one way to explain why monotonicity testing requires $\Omega(\log n)$ queries relies on the following reasoning. There exist $\Omega(\log n)$ different distance "profiles" our queries should capture; and it can be shown that in general, a small set of queries cannot capture many types of different profiles all at once. At a high level, our new lower bound is an extension of this argument, which uses a more general type of profiles. We start, then, with a formal definition of the basic profiles required for the case of monotonicity testing. Below we restate the required definitions related to the binary representation of numbers in $[n]$.

Definition IV.1 (Binary representation)**.** *For any $n \in \mathbb{N}$ which is a power of 2 and $t \in [n]$, the* binary representation *$B_n(t)$ of t is the unique tuple $(b_1^t, b_2^t, \dots, b_{\log_2 n}^t) \in \{0,1\}^{\log_2 n}$ satisfying $t = b_1^t \cdot 2^0 + b_2^t \cdot 2^1 + \cdots + b_{\log_2 n}^t \cdot 2^{\log_2 n - 1}$. For $i \in [\log_2 n]$, the* bit-flip operator*, $F_i \colon [n] \to [n]$, is defined as follows. Given $t \in [n]$ with $B_n(t) = (b_1^t, \dots, b_{\log n}^t)$, we set $F_n(t) = t'$ where $t' \in [n]$ is the unique integer satisfying $B_n(t') = (b_1^t, \dots, b_{i-1}^t, 1 - b_i^t, b_{i+1}^t, \dots, b_{\log n}^t)$. Finally, for any two distinct elements $x, y \in [n]$, let $M(x,y) \in [\log_2 n]$ denote the index of the most significant bit in which they differ, i.e., the largest i with $b_i^x \neq b_i^y$.*

Note that the bit-flip operator F_i is a permutation on $[n]$.

The construction.: We start by providing our lower bound construction $\mathcal{D}_{n,2}$, supported on sequences that are far from (12)-free.

Let $f^{\downarrow} \colon [n] \to [n]$ denote the (unique) decreasing permutation on $[n]$, i.e., the function $f^{\downarrow}(x) = n + 1 - x$ for any $x \in [n]$. For any $i \in [\log n]$, define $f_i \colon [n] \to [n]$ to be the composition of $f^{\downarrow}$ with the bit-flip operator F_i, that is, $f_i(x) = f^{\downarrow}(F_i(x))$ for any $x \in [n]$. Note that f_i is a permutation, as a composition of permutations. See Figure 1 for a visualization of the construction. Finally, define $\mathcal{D}_{n,2}$ as the uniform distribution over the sequences $f_1, f_2, \dots, f_{\log n}$.

The next lemma characterizes the set of all $(1,2)$-patterns in f_i.

Lemma IV.2. *Let $i \in [\log n]$. A pair $x < y \in [n]$ forms a $(1,2)$-copy in f_i if and only if $M(x,y) = i$.*

Proof: Let $x < y \in [n]$. If $M(x,y) > i$, then $F_i(x) < F_i(y)$ holds and so $f_i(x) = f^{\downarrow}(F_i(x)) > f^{\downarrow}(F_i(y)) = f_i(y)$, implying that (x,y) is not a $(1,2)$-copy. If $M(x,y) < i$ then x and y share the bit in index i of the binary representation, and thus flipping it either adds 2^{i-1} to both x and y or decreases 2^{i-1} from both of them. In both cases, $F_i(x) < F_i(y)$, and like the previous case we get $f_i(x) > f_i(y)$. Finally, if $M(x,y) = i$ then one can write $x = z + 0 \cdot 2^{i-1} + x'$ and $y = z + 1 \cdot 2^{i-1} + y'$, where z corresponds to the $\log n - i$ most significant bits in the binary representation (which are the same in x and y), and $x', y' < 2^{i-1}$ correspond to the $i-1$ least significant bits. Therefore, $F_i(x) = z + 1 \cdot 2^{i-1} + x' > z + 0 \cdot 2^{i-1} + y' = F_i(y)$ and thus $f_i(x) = f^{\downarrow}(F_i(x)) < f^{\downarrow}(F_i(y)) = f_i(y)$, as desired. ∎

We conclude that each of the sequences f_i is $(1/2)$-far from (12)-free.

Lemma IV.3. *For any $i \in [\log n]$, the sequence f_i contains a collection $\mathcal{C}$ of $n/2$ disjoint $(1,2)$-copies.*

Proof: For any $x \in [n]$ whose binary representation $B_n(x) = (b_1^x, \dots, b_{\log n}^x)$ satisfies $b_i^x = 0$, we have $M(x, F_i(x)) = i$. By Lemma IV.2, $(x, F_i(x))$ is thus a $(1,2)$-copy. Picking

$$\mathcal{C} = \{(x, F_i(x)) \ :\ x \in [n],\ b_i^x = 0\},$$

and noting that the pairs in $\mathcal{C}$ are disjoint, the proof follows. ∎

Binary Profiles.: We now formally define our notion of *binary profiles*, and describe why they are useful for proving lower bounds for problems of this type.

Definition IV.4 (Binary profiles captured)**.** *Let $n \in \mathbb{N}$ be a power of 2 and let $Q \subseteq [n]$. The set of* binary profiles *captured by Q is defined as*

$$\textsf{bin-prof}(Q) = \{i \in [\log n] \ :\ \exists x, y \in Q \text{ s.t. } M(x,y) = i\}.$$

The next lemma asserts that the number of binary profiles that set captures does not exceed (or even match) the size of the set.

Lemma IV.5. *Let $Q \subseteq [n]$ be a subset of size $q > 0$. Then $|\textsf{bin-prof}(Q)| \leq q - 1$.*

Proof: We proceed by induction on q. For $q \leq 2$, the statement clearly holds. Otherwise, let $i_{\max} = \max \textsf{bin-prof}(Q)$ be the maximum index of a bit in which two elements $x, y \in Q$ differ. For $j = 0, 1$, define

$$Q_j = \left\{x \in Q \ :\ \substack{\text{the binary representation of } x \\ \text{is } B_n(x) = (b_1^x, \dots, b_{\log n}^x), \\ \text{and } b_{i_{\max}}^x = j}\right\}.$$

Clearly, for any $x \in Q_0$ and $y \in Q_1$, we have $M(x,y) = i_{\max}$. We can thus write $\textsf{bin-prof}(Q)$ as

$$\textsf{bin-prof}(Q) = \textsf{bin-prof}(Q_0) \cup \textsf{bin-prof}(Q_1) \cup \{i_{\max}\},$$

from which we conclude that $|\textsf{bin-prof}(Q)| \leq |\textsf{bin-prof}(Q_0)| + |\textsf{bin-prof}(Q_1)| + 1 \leq |Q_0| - 1 + |Q_1| - 1 + 1 = |Q| - 1$, where the second inequality follows from the induction hypothesis. ∎

Proof for the case $k = 2$ using binary profiles.: After collecting all the ingredients required to prove the case $k = 2$ of Theorem IV.1, we now conclude the proof. Fix $0 < p < 1$, let n be a power of two, and consider the distribution $\mathcal{D}_{n,2}$ defined above, supported on sequences that are $(1/2)$-far from (12)-free (see Lemma IV.3). Let $Q \subseteq [n]$ be any subset of size at most $p \log n$. It suffices to show that, for $\boldsymbol{f} \sim \mathcal{D}_{n,2}$, the probability that Q contains a (12)-copy in $\boldsymbol{f}$ is less than p. By Lemma IV.2, Q contains a (12)-copy with respect to f_i if and only if $i \in \textsf{bin-prof}(Q)$. Thus, the above probability is equal to $|\textsf{bin-prof}(Q)|/\log n$, which, by Lemma IV.5, is at most $(|Q| - 1)/\log n < p$, as desired.

B. Hierarchical binary profiles and the lower bound

To prove Theorem IV.1 in its full generality, we significantly extend the proof presented in Subsection IV-A for the case $k = 2$, relying on a generalized hierarchical (and more involved) notion of a binary profile. Let $n > k \geq 2$ be powers of 2, and write $k = 2^h$ (so $h \in \mathbb{N}$). We show that there exist $\binom{\log_2 n}{h} = \binom{\log_2 n}{\log_2 k}$ different types of *binary h-profiles* (see Definition IV.6) with the following properties. First, a subset $Q \subseteq [n]$ can capture at most $|Q| - 1$ such profiles (Lemma IV.15 below, generalizing Lemma IV.3); and second, for each such profile there exists a sequence (in fact, a permutation) that is $(1/k)$-far from $(12 \ldots k)$-free, such that any set of queries Q that finds $(12 \ldots k)$-pattern with respect to this sequence must capture the given profile (Lemma IV.11 below, generalizing Lemma IV.2).

Hierarchical binary profiles.: While the proof for the case $k = 2$ relied on a rather basic variant of a binary profile, our lower bound for general k requires a more sophisticated, hierarchical type of profile, described below.

Definition IV.6 (binary h-profiles). *Let $(x_1, \ldots, x_k) \in [n]^k$ be a k-tuple of indices satisfying $x_1 < \cdots < x_k$. For an h-tuple $(i_1, \ldots, i_h) \in [\log_2 n]^h$ satisfying $i_1 < \cdots < i_k$, we say that $(x_1, \ldots, x_k)$ has h-*profile of type $(i_1, \ldots, i_h)$ *if,*

$$M(x_j, x_{j+1}) = i_{M(j-1,j)} \qquad \textit{for every } j \in [k-1].$$

For example, when $h = 3$ (and $k = 8$), a tuple $(x_1, \ldots, x_8) \in [n]^8$ with $x_1 < \ldots < x_8$ has binary 3-profile of type (i_1, i_2, i_3) if the sequence $(M(x_j, x_{j+1}))_{j=1}^7$ is $(i_1, i_2, i_1, i_3, i_1, i_2, i_1)$. See Figure 3 for a visual depiction of such a binary 3-profile.

Similarly to the case $k = 2$, given a set of queries $Q \subseteq [n]$, we shall be interested in the collection of h-profiles captured by Q.

Definition IV.7 (Binary h-profiles captured). *Let $n \geq k \geq 2$ be powers of 2 where $k = 2^h$. For any $Q \subseteq [n]$, we denote the set of all h-profiles captured by Q by*

$$\textsf{bin-prof}_h(Q) = \left\{ (i_1, \ldots, i_h) : \substack{\text{there exist } x_1,\ldots,x_k \in Q \text{ with} \\ x_1 < \cdots < x_k \text{ and } (x_1,\ldots,x_k) \\ \text{has } h\text{-profile of type } (i_1,\ldots,i_h)} \right\}.$$

The next lemma is one of the main ingredients of our proof, generalizing Lemma IV.5. It shows that a set Q of queries cannot capture $|Q|$ or more different h-profiles.

Lemma IV.8. *Let $h, n \in \mathbb{N}$ where $n \geq 2^h$ is a power of 2. For any $\emptyset \neq Q \subseteq [n]$, we have $|\textsf{bin-prof}_h(Q)| \leq |Q| - 1$.*

Proof: We proceed by induction on h. The case $h = 1$ was settled in Lemma IV.5. Suppose now that $h > 1$, and define $\emptyset = B_{\log n+1} \subseteq B_{\log n} \subseteq \ldots \subseteq B_1 = Q$ as follows. Set $B_{\log n+1} = \emptyset$, and given B_{i+1}, define the set $B_i \supseteq B_{i+1}$ as an arbitrary maximal subset of Q containing B_{i+1} which does not have two elements with $M(x, y) < i$.

Additionally, for each $j \in [\log_2 n]$, define

$$N_j = \left\{ (i_2, \ldots, i_h) : \substack{1 \leq j < i_2 \cdots < i_h \leq \log_2 n \text{ and} \\ (j, i_2, \ldots, i_h) \in \textsf{bin-prof}_h(Q)} \right\}.$$

Claim IV.9. *Let $j < i_2 < \ldots < i_h \in [\log n]$, and suppose that $(j, i_2, \ldots, i_h) \in \textsf{bin-prof}_h(Q)$. Then $(j, i_2, \ldots, i_h) \in \textsf{bin-prof}_h(B_j)$.*

Proof: Suppose that a tuple $(x_1, \ldots, x_k)$ with $x_1 < \cdots < x_k \in Q$ has h-profile $(j, i_2, \ldots, i_h)$. By the maximality of B_j, we know that for every $1 \leq \ell \leq k$ there exists $y_\ell \in B_j$ such that either $x_\ell = y_\ell$ or $M(x_\ell, y_\ell) < j$. Indeed, if this were not the case, then $B'_j := B_j \cup \{x_\ell\}$ would be a set that strictly contains B_j and does contain two elements $x \neq y$ with $M(x, y) = j$, a contradiction to the maximality of B_j. By definition of a profile, we conclude that $\{y_1, \ldots, y_k\} \subseteq B_j$ has h-profile $(j, i_2, \ldots, i_h)$. ■

Claim IV.10. *For any $j \in [\log n]$, we have $N_j \subseteq \textsf{bin-prof}_{h-1}(B_j \setminus B_{j+1})$.*

Proof: Suppose that $(i_2, \ldots, i_h) \in N_j$, then $(j, i_2, \ldots, i_h) \in \textsf{bin-prof}_h(Q)$. By the previous lemma, we know that $(j, i_2, \ldots, i_h) \in \textsf{bin-prof}_h(B_j)$. Therefore, there exists a tuple $(y_1, \ldots, y_k)$ where $y_1 < \ldots < y_k \in B_j$, that has h-profile of type $(j, i_2, \ldots, i_h)$.

For any $t \in k/2$, it holds that $M(y_{2t-1}, y_{2t}) = j$. Therefore, at most one of y_{2t-1}, y_{2t} is in B_{j+1}, and hence, for any such t there exists $z_t \in \{y_{2t-1}, y_{2t}\} \setminus B_{j+1} \subseteq B_j \setminus B_{j+1}$. Consider the tuple $(z_1, \ldots, z_{k/2})$, whose elements are contained in $B_j \setminus B_{j+1}$. It follows from our choice of z_t that $M(z_t, z_{t+1}) = M(y_{2t}, y_{2t+2})$ for any $t \in [k/2]$, from which we conclude that $(z_1, \ldots, z_{k/2})$ has $(h-1)$-profile $(i_2, \ldots, i_h)$. In other words, $(i_2, \ldots, i_h) \in \textsf{bin-prof}_h(B_j \setminus B_{j+1})$, as desired. ■

We are now ready to finish the proof of Lemma IV.8. Observe that $\textsf{bin-prof}_h(Q)$ and Q can be written as the following disjoint unions:

$$\textsf{bin-prof}_h(Q) = \bigcup_{j=1}^{\log_2 n} \{(j, i_2, \ldots, i_h) : (i_2, \ldots, i_h) \in N_j\}$$

and $Q = \bigcup_{j=1}^{\log_2 n} (B_j \setminus B_{j+1})$. It follows from the last claim and the induction assumption that

$$|N_j| \le |\mathsf{bin\text{-}prof}_{h-1}(B_j \setminus B_{j+1})| \le |B_j \setminus B_{j+1}|, \quad (16)$$

where for j with $N_j \neq \emptyset$ there is a strict inequality. Now, if N_j is empty for all j then, trivially, $|\mathsf{bin\text{-}prof}_h(Q)| = 0 \le |Q| - 1$. Otherwise, there exists some non-empty N_j, for which (16) yields a strict inequality, and we get

$$|Q| = \sum_{j=1}^{\log_2 n} |B_j \setminus B_{j+1}| > \sum_{j=1}^{\log_2 n} |N_j| = |\mathsf{bin\text{-}prof}_h(Q)|,$$

establishing the proof of the Lemma IV.8. ■

The construction.: For any $i_1 < i_2 < \ldots < i_h \in [\log n]$, we define $f_{i_1,\ldots,i_h} : [n] \to [n]$ as

$$f_{i_1,\ldots,i_h} := f^{\downarrow} \circ F_{i_h} \circ \ldots \circ F_{i_1},$$

where, as before, $\circ$ denotes function composition. In other words, for any $x \in [n]$ we have $f_{i_1,\ldots,i_h}(x) = f^{\downarrow}(F_{i_h}(F_{i_{h-1}}(\ldots(F_{i_1}(x)\ldots))))$. Note that $f_{i_1,\ldots,i_h}$ is indeed a permutation, as a composition of permutations. (See Figure 2, which visually describes the construction of $f_{i_1,\ldots,i_h}$ recursively, as a composition of F_{i_h} with $f_{i_1,\ldots,i_{h-1}}$.) We take $\mathcal{D}_{n,k}$ to be the uniform distribution over all sequences of the form $f_{i_1,\ldots,i_h}$ with $i_1 < i_2 < \ldots < i_k$. The size of the support of $\mathcal{D}_{n,k}$ is $\binom{\log_2 n}{h} = \binom{\log_2 n}{\log_2 k}$.

Structural properties of the construction.: Recall that our lower bound distribution $\mathcal{D}_{n,k}$ is supported on the family of permutations $f_{i_1,\ldots,i_h}$, where $i_1 < \ldots < i_h \in [\log n]$, described above. We now turn to show that these $f_{i_1,\ldots,i_h}$ satisfy two desirable properties. First, to capture a $(12\ldots k)$-copy in $(f_{i_1,\ldots,i_h})$, our set of queries Q must satisfy $(i_1,\ldots,i_h) \in \mathsf{bin\text{-}prof}_h(Q)$ (Lemma IV.11). And second, each such $f_{i_1,\ldots,i_k}$ is $(1/k)$-far from $(12\ldots k)$-free (Lemma IV.15).

Lemma IV.11. *Let $(x_1,\ldots,x_k) \in [n]^k$ be a k-tuple where $x_1 < \ldots < x_k$, and let $f = f_{i_1,\ldots,i_h}$ be defined as above. Then $f(x_1) < f(x_2) < \ldots < f(x_k)$ (i.e., $(x_1,\ldots,x_k)$ is a $(12\ldots k)$-copy with respect to $f_{i_1,\ldots,i_h}$) if and only if $(x_1,\ldots,x_k)$ has binary h-profile of type $(i_1, i_2\ldots, i_h)$. Furthermore, $f_{i_1,\ldots,i_h}$ does not contain increasing subsequences of length $k+1$ or more.*

Proof: The proof is by induction on h, with the base case $h = 1$ covered by Lemma IV.2; in particular, it follows from Lemma IV.2 that f_i has no increasing subsequence of length 3, since there exist no $x < y < z \in [n]$ with $M(x,y) = M(y,z) = i$.

For the inductive step, we need the following claim, which generalizes Lemma IV.2.

Claim IV.12. *A pair $x < y \in [n]$ satisfies $f_{i_1,\ldots,i_h}(x) < f_{i_1,\ldots,i_h}(y)$ if and only if $M(x,y) \in \{i_1,\ldots,i_h\}$.*

Proof: Let $F_{i_1,\ldots,i_h} = F_{i_h} \circ \ldots \circ F_{i_1}$. Since $f_{i_1,\ldots,i_h} = f^{\downarrow} \circ F_{i_1,\ldots,i_h}$, it suffices to show that $F_{i_1,\ldots,i_h}(x) > F_{i_1,\ldots,i_h}(y)$ if any only if $M(x,y) \in \{i_1,\ldots,i_h\}$. To do so, we prove the following two statements.

- For any $x < y \in [n]$, $F_i(x) > F_i(y)$ if and only if $M(x,y) = i$.
- For any $x < y \in [n]$, $M(F_i(x), F_i(y)) = M(x,y)$.

Indeed, using these two statements, the proof easily follows by induction: the value of $M(x,y)$ never changes regardless of which bit-flips we simultaneously apply to x and y. Now, applying any of the bit-flips F_i to x and y, where $i \neq M(x,y)$, does not change the relative order between them, while applying $F_{M(x,y)}$ does change their relative order. This means that a change of relative order occurs if and only if $M(x,y) \in \{i_1,\ldots,i_h\}$, which settles the claim.

The proof of the first statement was essentially given, word for word, in the proof of Lemma IV.2. The second statement follows by a simple case analysis of the cases where i is bigger than, equal to, or smaller than $M(x,y)$, showing that in any of these cases, $M(F_i(x), F_i(y)) = M(x,y)$. ■

Suppose now that $(x_1,\ldots,x_k) \in [n]^k$ is a tuple with $x_1 < \ldots < x_k$ and a binary h-profile of type $(i_1,\ldots,i_h)$ is a $(12\ldots k)$-copy in $f_{i_1,\ldots,i_h}$. By definition of a binary h-profile, we have that $M(x_j, x_{j+1}) \in \{i_1,\ldots,i_h\}$ for any $j \in [k-1]$, which, by the claim, implies that $f_{i_1,\ldots,i_h}(x_j) < f_{i_1,\ldots,i_h}(x_{j+1})$. It thus follows that $(x_1,\ldots,x_k)$ is a $(12\ldots k)$-copy in $f_{i_1,\ldots,i_h}$, as desired.

Conversely, suppose that a tuple $(x_1,\ldots,x_k) \in [n]^k$ with $x_1 < \ldots < x_k$ is a $(12\ldots k)$-copy in $f_{i_1,\ldots,i_k}$. We need to show that $(x_1,\ldots,x_k)$ has binary h-profile of type $(i_1,\ldots,i_h)$, that is, $M(x_j, x_{j+1}) = i_{M(j-1,j)}$ for every $j \in [k-1]$. Define $r = \operatorname{argmax}_j\{M(x_j, x_{j+1})\}$, and note that r is unique; otherwise, we would have $x < y < z \in [n]$ so that $M(x,y) = M(y,z)$, a contradiction.

Claim IV.13. $M(x_r, x_{r+1}) = i_h$.

Proof: By Claim IV.12, we know that $M(x_r, x_{r+1}) \in \{i_1,\ldots,i_h\}$. Suppose to the contrary that $M(x_r, x_{r+1}) \le i_{h-1}$. Then, $M(x_j, x_{j+1}) \le M(x_r, x_{r+1}) \le i_{h-1}$ for every $j \in [k-1]$, and by Claim IV.12, for any $j \in [k-1]$ we have $f_{i_1,\ldots,i_{h-1}}(x_j) \le f_{i_1,\ldots,i_{h-1}}(x_{j+1})$, that is, $(x_1, x_2, \ldots, x_k)$ is a $(12\ldots k)$-copy in $f_{i_1,\ldots,i_{h-1}}$. This contradicts the last part of the inductive hypothesis. ■

Claim IV.14. $r = k/2$.

Proof: Without loss of generality, suppose to the contrary that $r > k/2$ (the case where $r < k/2$ is symmetric). As the tuple $(x_1,\ldots,x_r)$ is an increasing subsequence for $f_{i_1,\ldots,i_h}$, we have $M(x_j, x_{j+1}) \in \{i_1,\ldots,i_h\}$ for any $j \in [r-1]$. By the maximality and uniqueness of r, $M(x_j, x_{j+1}) < i_h$ for any $j \in [r-1]$. Thus, it follows from Claim IV.12 that $(x_1,\ldots,x_r)$ is a $(12\ldots r)$-copy in $f_{i_1,\ldots,i_{h-1}}$, contradicting the last part of the inductive hypothesis. ■

It thus follows from the two claims that $M(x_{k/2}, x_{k/2+1}) = i_h$. Since $M(x_j, x_{j+1}) \in \{i_1,\ldots,i_{h-1}\}$ for any $j \in [k-1] \setminus \{k/2\}$, we conclude, again from Claim IV.12, that $(x_1, \ldots, x_{k/2})$ and $(x_{k/2+1}, \ldots, x_k)$ both induce length-$(k/2)$ increasing subsequences in $f_{i_1,\ldots,i_{h-1}}$. By the inductive hypothesis, they both have binary $(h-1)$-profile $(i_1, \ldots, i_{h-1})$. Combined with the last two claims, we conclude that $(x_1, \ldots, x_k)$ has binary h-profile $(i_1, \ldots, i_h)$, as desired.

It remains to verify that $f_{i_1,\ldots,i_h}$ does not contain an increasing subsequence of length $k+1$. If, to the contrary, it does contain one, induced on some tuple $(x_1, \ldots, x_{k+1}) \in [n]^{k+1}$ where $x_1 < \ldots < x_{k+1}$, then, applying the last two claims to the length-k two tuples $(x_1, \ldots, x_k)$ and $(x_2, \ldots, x_{k+1})$, we conclude that $M(x_{k/2}, x_{k/2+1}) = M(x_{k/2+1}, x_{k/2+2}) = i_h$. However, as discussed above, there cannot exist $x < y < z \in [n]$ with $M(x, y) = M(y, z)$ – a contradiction. ■

It remains to prove that each $f_{i_1,\ldots,i_h}$ is indeed $(1/k)$-far from $(12\ldots k)$-free. After we spent quite some effort to characterize *all* $(12\ldots k)$-copies in $f_{i_1,\ldots,i_h}$, this upcoming task is much simpler.

Lemma IV.15. *Let $n \geq k \geq 2$ be powers of two and write $k = 2^h$. The sequence $f_{i_1,\ldots,i_h} \colon [n] \to [n]$, defined above, contains n/k disjoint $(12\ldots k)$-copies.*

Proof: Fix $i_1 < \ldots < i_h$ as in the statement of the lemma. We say that $x, y \in [n]$ with binary representations $B_n(x) = (b_1^x, \ldots, b_{\log n}^x)$ and $B_n(y) = (b_1^y, \ldots, b_{\log n}^y)$ are $(i_1, \ldots, i_h)$-*equivalent* if $b_i^x = b_i^y$ for any $i \in [\log n] \setminus \{i_1, \ldots, i_h\}$. Clearly, this is an equivalence relation, partitioning $[n]$ into n/k equivalence classes, each of size exactly $k = 2^h$. Moreover, it is straightforward to verify that the elements $x_1 < x_2 < \ldots < x_k$ of any equivalence class satisfy $M(x_j, x_{j+1}) \in \{i_1, \ldots, i_h\}$ for any $j \in [k-1]$, and thus, by Claim IV.12, $(x_1, \ldots, x_k)$ constitutes a $(12\ldots k)$-copy in $f_{i_1,\ldots,i_k}$. ■

It now remains to connect all the dots for the proof of Theorem IV.1.

Proof of Theorem IV.1: Fix $0 < p < 1$, let $n \geq k$ be powers of 2, and write $k = 2^h$. As before, we follow Yao's minimax principle [40], letting $\mathcal{D}_{n,k}$ be the uniform distribution over all $\binom{\log_2 n}{h} = \binom{\log_2 n}{\log_2 k}$ sequences (in fact permutations) $f_{i_1,\ldots,i_h} \colon [n] \to [n]$, where $i_1 < \ldots < i_h \in [\log n]$. Recall that, by Lemma IV.15, this distribution is supported on sequences that are $(1/k)$-far from $(12\ldots k)$-free.

It suffices to show that, for $\boldsymbol{f} \sim \mathcal{D}_{n,k}$, the probability for any subset $Q \subseteq [n]$ of size at most $p\binom{\log_2 n}{h}$ to capture a $(12\ldots k)$-copy in $\boldsymbol{f}$ is less than p. Indeed, by Lemma IV.11, Q captures a copy in $f_{i_1,\ldots,i_h}$ if and only if $(i_1, \ldots, i_h) \in \textsf{bin-prof}_h(Q)$, so the success probability for any given Q is exactly $|\textsf{bin-prof}_h(Q)|/\binom{\log_2 n}{h} < |Q|/\binom{\log_2 n}{h} \leq p$ for any $Q \subseteq [n]$ with $|Q| \leq p\binom{\log_2 n}{h}$, where the first inequality follows from Lemma IV.8. The proof of Theorem IV.1 follows. ■

References

[1] I. Newman, Y. Rabinovich, D. Rajendraprasad, and C. Sohler, "Testing for forbidden order patterns in an array," in *Proceedings of the 28th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2017, pp. 1582–1597.

[2] O. Ben-Eliezer and C. L. Canonne, "Improved bounds for testing forbidden order patterns," in *Proceedings of the 29th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2018, pp. 2093–2112.

[3] Y. Dodis, O. Goldreich, E. Lehman, S. Raskhodnikova, D. Ron, and A. Samorodnitsky, "Improved testing algorithms for monotonicity," in *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, 1999, pp. 97–108.

[4] O. Goldreich, S. Goldwasser, E. Lehman, D. Ron, and A. Samordinsky, "Testing monotonicity," *Combinatorica*, vol. 20, no. 3, pp. 301–337, 2000.

[5] E. Fischer, E. Lehman, I. Newman, S. Raskhodnikova, R. Rubinfeld, and A. Samorodnitsky, "Monotonicity testing over general poset domains," in *Proceedings of the 34th ACM Symposium on the Theory of Computing (STOC)*, 2002, pp. 474–483.

[6] P. Afshani, K. Matulef, and B. T. Wilkinson, "Property testing on linked lists," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 20, p. 187, 2013.

[7] P. Berman, S. Raskhodnikova, and G. Yaroslavtsev, "L_p-testing," in *Proceedings of the 46th ACM Symposium on the Theory of Computing (STOC)*, 2014, pp. 164–173.

[8] A. Belovs, "Adaptive Lower Bound for Testing Monotonicity on the Line," in *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, 2018, pp. 31:1–31:10.

[9] R. K. S. Pallavoor, S. Raskhodnikova, and N. M. Varma, "Parameterized property testing of functions," *ACM Transactions on Computation Theory*, vol. 9, no. 4, pp. 17:1–17:19, 2018.

[10] O. Ben-Eliezer, "Testing local properties of arrays," in *Proceedings of the 10th Conference on Innovations in Theoretical Computer Science (ITCS)*, 2019, pp. 11:1–11:20.

[11] O. Goldreich, *Introduction to property testing*. Cambridge University Press, 2017.

[12] F. Ergün, S. Kannan, S. R. Kumar, R. Rubinfeld, and M. Vishwanthan, "Spot-checkers," *Journal of Computer and System Sciences*, vol. 60, no. 3, pp. 717–751, 2000.

[13] E. Fischer, "On the strength of comparisons in property testing," *Information and Computation*, vol. 189, no. 1, pp. 107–116, 2004.

[14] E. Blais, J. Brody, and K. Matulef, "Property testing lower bounds via communication complexity," *Computational Complexity*, vol. 21, no. 2, pp. 311–358, 2012.

[15] J. Briët, S. Chakraborty, D. García-Soriano, and A. Matsliah,

"Monotonicity testing and shortest-path routing on the cube," *Combinatorica*, vol. 32, no. 1, pp. 35–53, 2012.

[16] D. Chakrabarty and C. Seshadhri, "Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids," in *Proceedings of the 45th ACM Symposium on the Theory of Computing (STOC)*, 2013, pp. 419–428.

[17] X. Chen, R. A. Servedio, and L.-Y. Tan, "New algorithms and lower bounds for monotonicity testing," in *Proceedings of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2014, pp. 285–295.

[18] X. Chen, A. De, R. A. Servedio, and L.-Y. Tan, "Boolean function monotonicity testing requires (almost) $n^{1/2}$ non-adaptive queries," in *Proceedings of the 47th ACM Symposium on the Theory of Computing (STOC)*, 2015, pp. 519–528.

[19] S. Khot, D. Minzer, and M. Safra, "On monotonicity testing and boolean isoperimetric type theorems," in *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2015, pp. 52–58.

[20] A. Belovs and E. Blais, "Quantum algorithm for monotonicity testing on the hypercube," *Theory of Computing*, vol. 11, no. 16, pp. 403–412, 2015.

[21] D. Chakrabarty and C. Seshadhri, "An $o(n)$ monotonicity tester for boolean functions over the hypercube," *SIAM Journal on Computing*, vol. 45, no. 2, pp. 461–472, 2016.

[22] X. Chen, E. Waingarten, and J. Xie, "Beyond Talagrand functions: new lower bounds for testing monotonicity and unateness," in *Proceedings of the 49th ACM Symposium on the Theory of Computing (STOC)*, 2017, pp. 523–536.

[23] D. Chakrabarty and C. Seshadhri, "Adaptive boolean monotonicity testing in total influence time," in *Proceedings of the 10th Conference on Innovations in Theoretical Computer Science (ITCS)*, 2019, pp. 20:1–20:7.

[24] E. Blais, S. Raskhodnikova, and G. Yaroslavtsev, "Lower bounds for testing properties of functions over hypergrid domains," in *Proceedings of the 29th Conference on Computational Complexity (CCC)*, 2014, pp. 309–320.

[25] D. Chakrabarty and C. Seshadhri, "An optimal lower bound for monotonicity testing over hypergrids," *Theory of Computing*, vol. 10, no. 17, pp. 453–464, 2014.

[26] H. Black, D. Chakrabarty, and C. Seshadhri, "A $o(d)\cdot \text{polylog} n$ monotonicity tester for boolean functions over the hypergrid $[n]^d$," in *Proceedings of the 29th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2018, pp. 2133–2151.

[27] R. Rubinfeld and M. Sudan, "Robust characterization of polynomials with applications to program testing," *SIAM Journal on Computing*, vol. 25, no. 2, pp. 252–271, 1996.

[28] O. Goldreich, S. Goldwasser, and D. Ron, "Property testing and its connection to learning and approximation," *Journal of the ACM*, vol. 45, no. 4, pp. 653–750, 1998.

[29] S. Guillemot and D. Marx, "Finding small patterns in permutations in linear time," in *Proceedings of the 25th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2014, pp. 82–101.

[30] J. Fox, "Stanley–Wilf limits are typically exponential," *arXiv preprint arXiv:1310-8378*, 2013, also: Advances in Mathematics, to appear.

[31] L. Kozma, "Faster and simpler algorithms for finding large patterns in permutations," *arXiv preprint arXiv:1902-08809*, 2019. [Online]. Available: http://arxiv.org/abs/1902.08809

[32] M. Parnas, D. Ron, and R. Rubinfeld, "Tolerant property testing and distance approximation," *Journal of Computer and System Sciences*, vol. 72, no. 6, pp. 1012–1042, 2006.

[33] N. Ailon, B. Chazelle, S. Comandur, and D. Liu, "Estimating the distance to a monotone function," *Random Structures and Algorithms*, vol. 31, no. 3, pp. 371–383, 2007.

[34] M. E. Saks and C. Seshadhri, "Estimating the longest increasing sequence in polylogarithmic time," *SIAM J. Comput.*, vol. 46, no. 2, pp. 774–823, 2017, short version in FOCS'10 [41].

[35] P. Gopalan, T. S. Jayram, R. Krauthgamer, and R. Kumar, "Estimating the sortedness of a data stream," in *Proceedings of the 18th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2007, pp. 318–327.

[36] A. Gál and P. Gopalan, "Lower bounds on streaming algorithms for approximating the length of the longest increasing subsequence," *SICOMP*, vol. 39, no. 8, pp. 3463–3479, 2010, short version in FOCS'07 [42].

[37] M. Saks and C. Seshadhri, "Space efficient streaming algorithms for the distance to monotonicity and asymmetric edit distance," in *Proceedings of the 24th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2013, pp. 1698–1709.

[38] F. Ergün and H. Jowhari, "On the monotonicity of a data stream," *Combinatorica*, vol. 35, no. 6, pp. 641–653, 2015, short version in SODA'08 [43].

[39] T. Naumovitz and M. E. Saks, "A polylogarithmic space deterministic streaming algorithm for approximating distance to monotonicity," in *Proceedings of the 26th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2015, pp. 1252–1262.

[40] A. C. Yao, "Probabilistic computations: Toward a unified measure of complexity," in *18th Annual Symposium on Foundations of Computer Science (SFCS)*, 1977, pp. 222–227.

[41] M. Saks and C. Seshadhri, "Estimating the longest increasing sequence in polylogarithmic time," in *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2010, pp. 458–467.

[42] A. Gál and P. Gopalan, "Lower bounds on streaming algorithms for approximating the length of the longest increasing subsequence," in *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2007, pp. 294–304.

[43] F. Ergün and H. Jowhari, "On distance to monotonicity and longest increasing subsequence of a data stream," in *Proceedings of the 19th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2008, pp. 730–736.