Testing frequency distributions in a stream

Claire Mathieu
IRIF-CNRS
claire@irif.fr

Michel de Rougemont
University Paris II
IRIF-CNRS
mdr@irif.fr

Abstract

We study how to verify specific frequency distributions when we observe a stream of N data items taken from a universe of n distinct items. We introduce the *relative Fréchet distance* to compare two frequency functions in a homogeneous manner. We consider two streaming models: insertions only and sliding windows. We present a Tester for a certain class of functions, which decides if f is close to g or if f is far from g with high probability, when f is given and g is defined by a stream. If f is uniform we show a space $\Omega(n)$ lower bound. If f decreases fast enough, we then only use space $O(\log^2 n \cdot \log\log n)$. The analysis relies on the Spacesaving algorithm [18, 20] and on sampling the stream.

2012 ACM Subject Classification

Keywords and phrases Verification of a distribution, Property Testing, Frequent items, Fréchet distance

Digital Object Identifier 10.4230/LIPIcs...

Funding *Claire Mathieu*: [funding]
Michel de Rougemont: [funding]

1 Introduction

We study streams of data items and the distribution g of frequencies where $g(i)$ is the number of occurrences of the ith most frequent item in the stream. Here, we consider a stream of length N of elements from a domain U of size n and we want to approximately verify whether the frequency g of the stream is close to a fixed distribution f. We may also look at two different streams and ask whether their frequencies g_1 and g_2 are close to each other. In practice, of particular interest are settings with single-pass streams and very small memory [17]. What kind of properties can we hope to verify if we only allow poly-logarithmic space? We first prove an $\Omega(n)$ space lower bound on the space of the Tester, theorem 1, when f is the uniform distribution. We therefore need some additional conditions on the frequency function f.

The approximation follows the Property Testing framework, where we use the *relative Fréchet distance* between two frequency functions f and g as a new measure of distance. Given a stream and a frequency function f which satisfies a certain weak continuity property and is decreasing fast enough, we decide in space $O(\log^2 n \cdot \log\log n)$ whether the frequency g defined by the stream is close to f for the relative Fréchet distance.

Frequency functions. There are two different ways to study frequency functions. Either the function is from U to $\mathbf{N}_+$ and gives the frequency of each item, in which case the problem is easy; or the function f is from $\{1, 2, ..n\}$ to N such that $f(i)$ is the frequency of the i-th most frequent item; we take the latter viewpoint. A *frequency function* f is a non-negative integer-valued function over a set of elements such that $f(i)$ is the number of occurences of the ith most frequent element. The problem is harder as we don't know which element of

Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

U is the i-th most frequent, and, for example, the two streams *aaabba* and *bbbaab* that are identical up to permuting the items have identical frequency functions even though b has 2 occurrences in the first stream and 4 occurrences in the second stream.

Relative Fréchet distance. What is the relative Fréchet distance? The classical (discrete) Fréchet distance between two discrete distributions, viewed as sequences of points $\{(i, f(i))\}$ and $\{(i, g(i))\}$ is an absolute distance. It is the minimum distance of a coupling between the two sequences. The discrete Fréchet distance between discrete curves has been studied, in particular in computational geometry, including in the streaming context [8, 12], but with a different oracle model. We generalize this distance to a *relative Fréchet distance*: the distance of the coupling must preserve within $(1 + \varepsilon_1)$ the distance on the x-axis and within $(1 + \varepsilon_2)$ the distance on the y-axis.

Additional assumptions. The weak continuity property, called ε-step compatibility, assumes that the frequency function f may have discontinuities, i.e. large drops, but no double discontinuities. Points which are ε-close on the x-axis are also close on the y-axis.

We combined two well known techniques: the Spacesaving algorithm [18, 20] which deterministically selects the most frequent items approximately and the Minhash technique which approximates the low frequencies probabilistically. Our main results are:

- A link between the relative Fréchet distance of two discrete functions which are step-compatible, and a separating rectangle, theorem 13,
- A streaming Tester for a step compatible frequency function and the relative Fréchet distance, when f is γ-decreasing. The Tester uses $O(\log^2 n \cdot \log\log n)$ space, theorem 10.

In the second section, we present our main definitions. In the third section, we define the classical distributions with a compact representation, the Spacesaving algorithms whose fine analysis, lemma 21, is in the appendix A.2. In the fourth section, we introduce the relative Fréchet distance and the proof of theorem 13 is in the appendix B. In the fifth section we present the streaming Tester first for the insertion only model, then for the sliding window model.

1.1 Motivations and comparison with other approaches

Problems that are hard in the worst-case may be much simpler for inputs which follow specific distributions, for example power law distributions. It is therefore important to verify if some given data follow certain distributions, when the data arrive in a stream. The area of *Distribution testing* [5] studies this type of problems in general.

We first work in the insertion model, and then consider the *sliding window* model with insertions and deletions outside a window. We will study the turnstile model [19] with insertions and deletions for the bounded deletions model[1] from [14] in some later work[2]. Notice that the sliding window model is not a bounded deletion model, as I/D tends to 1 when I goes to ∞.

In [6], the verification of properties of a stream is studied with streaming interactive proofs. In [13], the verification is done efficiently thanks to prior work done by annotating the stream in advance in preparation for the task. In our setting, we use the Property Testing

[1] In such a model, the number D of deletions is related to the number I of insertions: $D \leq (1 - 1/\alpha)I$, for some constant $\alpha \geq 1$.

[2] Our study of this model is deferred because the bounded deletions model is studied in [20] but the algorithm therein has some issues currently in the process of being corrected.

framework without any annotations or other additional prior information. We propose this setting for the verification of the distribution of frequent items.

A standard problem in statistics is to check if some observed data, i.e. in the insertion only model, approximately fit some statistics F where $F(e_i)$ is the frequency of the element e_i. Let G be the frequency of the elements of the observed data. The standard χ^2 test computes:

$$\chi^2(F,G) = \sum_{i=1}^{n} (F(e_i) - G(e_i))^2 / F(e_i)$$

If $\chi^2(F,G) \leq a$, we know that G follows F with confidence $1-\alpha$, for example $a = 11,07$ and $1-\alpha = 95\%$. In this setting, [10] gives an algorithm which uses space $O(\log N.\sqrt{N})$ to decide if F and G are close or far for the χ^2 test. In fact, the AMS-sketch [2] can be adapted and requires only $O(\log n)$ space.

In this paper, we study the case when the frequency function g is given by a stream of N data items and we want to test if g approximately follows the frequency function f over the domain $\{1, 2, ...n\}$, in polylogarithmic space and without necessarily knowing the exact value of n. For example f might be a Zipf distribution. If we observe sliding windows of the stream, the frequency g may be stable in each window, although the most frequent items change over time.

Thus we are interested in making restrictive but reasonable assumptions that will imply that we can test in polylogarithmic space. We turn to a measure of proximity between distributions that we call *relative Fréchet distance*. We use the Spacesaving algorithm [18] with additional hypothesis on the function g, to be step-compatible and γ-decreasing, in order to obtain relative errors on the frequencies, as in [7] to approximate the rank of an item. Our main result is a Tester when f follows some continuity property for the relative Fréchet distance. If f satisfies a decreasing condition, the Tester uses $O(\log^2 n \cdot \log\log n)$ space.

2 Definitions and Main Result

The SpaceSaving algorithm was introduced in [18] to compute estimates of th frequencies of the k most frequent elements in a stream of elements from a universe of size n, using a table T with $K \leq n$ entries. Each table entry consists of an element and a counter (plus some auxiliary information), which is a rough estimate of the frequency of the element in the stream. The table is kept sorted by counters: $c_1 \geq c_2 \geq \cdots c_K$. The SpaceSaving algorithm is straightforward: if the next element e of the stream is in T, then the algorithm increments the corresponding counter; otherwise, it substitutes e for the element whose counter is minimum (in position K), and increments the corresponding counter. Let $\text{count}(e)$ be the value of the counter of elment e. See Appendix A for details.

The following additive error result was proved in the original paper. (Note that f_i is the ith largest frequency whereas c_i is the ith largest counter, so they count occurences of different elements in general).

▶ **Lemma 1.** *[18] Let K denote the size of the table, N denote the length of the stream, and c_K the variable defined in the Space Saving algorithm. Then for every $i \leq K$ we have $|f_i - c_i| \leq c_K$ and for every $i > K$ we have $f_i \leq c_K$; moreover, $c_K \leq N/K$.*

Here, we would like to leverage the power of the SpaceSaving algorithm to test whether the *entire* distribution of frequencies of the stream approximates a given frequency distribution, with small *relative* error. For example, this can be used to check whether a stream of graph edges defines a graph whose degree sequence is close to a predicted degree sequence.

First, we need to specify what we mean by "close". To that end, we first define a relative distance between points.

▶ **Definition 2.** *Let $0 < \varepsilon_1, \varepsilon_2 < 1$. We say that two non-negative numbers a, b are ε-*close*, and denote it by $a \simeq_\varepsilon b$, if $|a - b| \leq \varepsilon \cdot \min\{a, b\}$. We say that two points $p = (x, y)$ and $p' = (x', y')$ are $(\varepsilon_1, \varepsilon_2)$-close, and denote it by $p \simeq_{(\varepsilon_1, \varepsilon_2)} p'$, if $x \simeq_{\varepsilon_1} x'$ and $y \simeq_{\varepsilon_2} y'$.*

2.1 Algorithm 1

With that, we can describe our streaming algorithm to test whether the frequency distribution g defined by the elements of a stream is close to a specified frequency distribution f. Let $z_i = (1 + \varepsilon_1^2)^i$ for $i \geq 1$. We first define a partition of $\{1, 2, \ldots, n\}$ for the frequency function f into *Boxes* $[\ell_j, r_j]$ in Lemma 8 and only consider the z_i which are not close to the Boxes endpoints. The streaming Algorithm 1 consists of the following three steps in parallel for all $\lceil \log_{1+\varepsilon_1} n \rceil$ distinct values of i:

1. We sample each element of the stream s to define a substreams s_i. The sample probability is chosen so that (assuming that the frequency distribution g of the elements of stream s equals f), in expectation substream s_i contains $\Theta(1/\varepsilon_1^2)$ elements whose number of occurences is greater than $f(z_i)$.
2. We consider two cases, the case when $\varepsilon_2 f(z_i) \leq f(n)$ and we run the SpaceSaving algorithm with a table size $K_i = O(h(\gamma, \varepsilon_1, \varepsilon_2). \log n \cdot \log \log n)$, and the case when $\varepsilon_2 f(z_i) > f(n)$ and we do an exact counting. The two cases are determined by a value $t_0 = n/\gamma^{\log(1/\varepsilon_2)}$. In the first case, $z_i \leq t_0$ and in the second case $z_i > t_0$.
   Let r be the expected number of elements of s_i whose number of occurences is greater than $f(z_i)$, and let c_r be the corresponding value of the counter in the table.
3. We apply a simple Coherence test to check whether point (z_i, c_r) is close to $(t, f(t))$ for some t.

Finally, the algorithm accepts with probability $1 - \delta$ if and only if the Coherence test succeeds for every substream s_i.

The frequency function g of the stream s and the reference frequency f are both from $\{1, 2, ..n\}$ to N.

▷ Notation 1. Let K_i denote the size of the table used by Algorithm 1 for the substream s_i. We set

$$K_i = \frac{4.z_i}{\varepsilon_2.a_i} \cdot \frac{2(\gamma - 1)}{2 - \gamma} \cdot \frac{\log n}{\delta} \cdot (1 + \varepsilon_1) = O(\log n \cdot \log \log n),$$

where $z_i = (1 + \varepsilon_1^2)^i$, $a_i = \varepsilon_1^2 z_i / \log \log n$, γ is such that f and g are γ-decreasing (see Definition 9), $\varepsilon_1, \varepsilon_2$ are the Fréchet parameters (see Definition 4), and δ is the desired error probability of the Tester (see Definition 5). Let U be the set of elements e and $\text{occ}(e)$ is the number of occurences of e. For each stream s_i, the counter c_r of the Spacesaving algorithm is compared with $f(z_i)$ where $r = \lceil z_i / a_i \rceil$.

Algorithm 1 gives the complete description.

Tester Algorithm 1 A$(\varepsilon_1, \varepsilon_2, \delta$**; step-compatible function** $f)$
Data: a stream s from a universe $\{e_1, e_2, \dots, e_n\}$.
Compute the decomposition of $[1, n]$ into Boxes according to Lemma 8 for f.
for *each* $i = 1, 2, \dots, \lceil \log_{1+\varepsilon_1} n \rceil$ *:* **do**
  $z_i \leftarrow (1 + \varepsilon_1^2)^i$; $K_i \leftarrow O(\log n \cdot \log\log n)$;
  If z_i is not ε_1^2-close to a Box endpoint then:
  1. Defining substreams ;
  $a_i \leftarrow \Theta(\varepsilon_1^2 . z_i / \log\log n)$; $h_i \leftarrow$ uniform hash function over $[1, a_i]$;
  Let s_i denote the substream consisting of those elements e s.t. $h_i(e) = 1$;
  2. Dealing with substreams s_i **in parallel** ;
  if $f(n) < \varepsilon_2 . f(z_i)$ **then**
    on substream s_i, run SpaceSaving with a table T_i of size K_i
  else
    on substream s_i, run exact counting algorithm with a table T_i of size equal to the number of distinct elements in s_i.
  end
  3. Coherence Test ;
  $r \leftarrow \lceil z_i / a_i \rceil$; $c_r \leftarrow$ the counter at position r of table T_i ;
  if $c_r \not\simeq_{3.\varepsilon_2} f(z_i)$ **then**
    break and output NO
  end
end
output YES

Algorithm 1: The Streaming Tester

2.2 Analysis of Algorithm 1

What does this algorithm accomplish? Before we answer that question, we first need to define what it means for two functions to be relatively close. We thus introduce the notion of *relative Fréchet distance* between two functions. The (absolute) Fréchet distance is based on the notion of coupling, defined in [9] and which we now recall. Here we also define the *relative length* of a coupling.

▶ **Definition 3.** *Let f and g be two functions with domain $\{1, \cdots, n\}$. For $1 \leq t \leq n$, consider the points $u_t = (t, f(t))$ and $v_t = (t, g(t))$. A* coupling *between f and g is a sequence $(u_{a_1}, v_{b_1}), (u_{a_2}, v_{b_2}), \cdots, (u_{a_m}, v_{b_m})$ such that $a_1 = 1, b_1 = 1, a_m = n, b_m = n$, and for all i we have $a_{i+1} \in \{a_i, a_i + 1\}$ and $b_{i+1} \in \{b_i, b_i + 1\}$. The* relative length *of the coupling is the minimum $\varepsilon_1, \varepsilon_2$ such that for all i we have $u_{a_i} \simeq_{(\varepsilon_1, \varepsilon_2)} v_{b_i}$.*

We now define the relative Fréchet distance.

▶ **Definition 4.** *(Relative Fréchet distance) Let f and g be two functions with domain $\{1, \cdots, n\}$. We say that f and g are $(\varepsilon_1, \varepsilon_2)$-close, denoted $f \sim_{(\varepsilon_1, \varepsilon_2)} g$, if there exists a coupling of relative length at most $\varepsilon_1, \varepsilon_2$.*

Note that unlike the absolute Fréchet distance, the relative Fréchet distance is invariant by scaling.

The relation $f \sim_{(\varepsilon_1, \varepsilon_2)} g$ is reflexive and symmetric. The relative Fréchet distance differs from the absolute Fréchet distance. For example, consider two families of step functions, depending on an integer parameter a:

$$f(i) = \begin{cases} 2a & \text{if } i \leq 10a \\ a & \text{if } i > 10a \end{cases} \qquad g(i) = \begin{cases} 2a & \text{if } i \leq 11a \\ a & \text{if } i > 11a \end{cases} \tag{1}$$

The absolute Frechet distance between f and g is a which is arbitrary large, whereas the relative Frechet distance is $\varepsilon = 10\%$, independent of a.

The notion of a Property Tester goes back to [4] and the streaming version to [11]. We use the tolerant version of a Tester.

▶ **Definition 5.** *Let $\varepsilon_1, \varepsilon_2, \delta \in (0,1)$. A* streaming δ-Tester *is a streaming algorithm A which, given a function f over $\{1, 2, \cdots, n\}$, takes as input a stream of elements from a universe of size n defining a frequency function g such that $g(j)$ is the number of occurrences of the jth most frequent element in the stream and:*

- *if $f = g$ then A accepts with probability at least $1 - \delta$; and*
- *if g is $(10\varepsilon_1, 10\varepsilon_2)$-far from f for the relative Fréchet distance then A rejects with probability at least $1 - 4\delta$.*

A more general *Tolerant δ-Tester* replaces the first condition with the tolerant version: if g is $(\varepsilon_1/10, \varepsilon_2/10)$-close to f for the relative Fréchet distance then A accepts with probability at least $1 - \delta$. We want Algorithm 1 to be a streaming δ-Tester. For that, we need two assumptions on the frequency distributions being tested: they must be $s$tep-compatible and γ-decreasing, two notions that we now define.

▶ **Definition 6.** *(Rectangle and Step compatibility).*

Let $0 < \varepsilon_1, \varepsilon_2 < 1$. An $(\varepsilon_1, \varepsilon_2)$-rectangle *is a set $R \subseteq [1, n] \times [0, \infty]$ with bottom left corner (x, y) and top right corner $(x(1+\varepsilon_1), y(1+\varepsilon_2))$. A function f with domain $\{1, \cdots, n\}$ is* $(\varepsilon_1, \varepsilon_2)$-step-compatible *if for every t, $1 \leq t \leq n$, there exists an $(\varepsilon_1, \varepsilon_2)$-rectangle R containing $(t, f(t))$ and all the points of f within the horizontal span of R.*

Zipf distributions assume $f_i = \frac{c}{i^\alpha}$ for $\alpha > 0$, and power laws assume $\alpha > 1$. We ignore rounding problems as each f_i is an integer value. Power laws and Zipf distributions are $(\varepsilon, \varepsilon')$-*step-compatible* whereas the geometric distribution is not *step-compatible*, as it has large consecutive discontinuities.

▶ **Lemma 7.** *If f is the frequency function of a Zipf distribution of parameter α, then f is $(\varepsilon/\alpha, \varepsilon)$-step-compatible.*

Proof. Let us find $j > i$ such that $f(j) \simeq f(i)/1 + \varepsilon$. We have:

$$f(j) = \frac{c}{j^\alpha} \simeq \frac{c}{i^\alpha.(1+\varepsilon)}$$

Then $j \simeq i.(1+\varepsilon)^{1/\alpha} \simeq i.(1 + \varepsilon/\alpha)$. ◀

▶ **Lemma 8.** *(Step-compatible property).*

Let f be an $(\varepsilon_1, \varepsilon_2)$-step-compatible frequency function. *Then there exists a partition of $\{1, 2, \ldots, n\}$ into* Boxes $[\ell_j, r_j]$ *such that for all j:*

- *$\ell_{j+1} > (1 + \varepsilon_1)\ell_j$; and*
- *$f(\ell_j) \leq (1 + 4\varepsilon_2) f(r_j)$.*

Proof. The intervals are defined in a 2-step process. The first step is greedy: let $(x_i)_{i \geq 1}$ denote the sequence of distinct values of $\lceil (1 + \varepsilon_1/3)^j \rceil$ and $y_i = x_{i+1} - 1$ (or $y_i = n$ if i is the last term of the sequence). Using the fact that f is $(\varepsilon_1, \varepsilon_2)$-step-compatible, let R_i denote the $(\varepsilon_1, \varepsilon_2)$ rectangle containing $(x_i, f(x_i))$ and note that R_i must contain $(x_{i+1}, f(x_{i+1}))$ or $(x_{i-1}, f(x_{i-1}))$ (otherwise its relative horizontal span would be less than $(1+\varepsilon_1)^2 < 1+\varepsilon_1$), so it intersects R_{i-1} or R_{i+1}. Extract a maximal subsequence $R_{i_1}, R_{i_2}, R_{i_3}, \cdots$ of R_i's containing R_1 and among which no two intersect. The sequence ℓ_j then consists of the left endpoints of the rectangles in that subsequence. Finally, we set $r_j = \ell_{j+1} - 1$ (except that we set $r_j = n$ for the last interval).

Each interval $[\ell_j, r_j]$ contains at least the horizontal span of a rectangle R_{i_j} of the subsequence, so the first property holds: $\ell_{j+1} > (1+\varepsilon_1)\ell_j$. Consider the rightmost rectangle R_k that intersects R_{i_j}, and the leftmost rectangle $R_{k'}$ that intersects R_{i_j}. All the points $(t, f(t))$ with $\ell_j \le t \le r_j$ are in the horizontal span of $R_{k'} \cup R_{i_j} \cup R_k$. The vertical span is therefore at most that of 3 $(\varepsilon_1, \varepsilon_2)$ rectangles, i.e. $f(\ell_j) \le (1+\varepsilon_2)^3 f(r_j) < (1+4\varepsilon_2) f(r_j)$. ◀

▶ **Definition 9.** *(γ-decreasing) Let $\gamma > 1$. A non-increasing function f with domain $\{1, \cdots, n\}$ is γ-decreasing if for all t such that $1 \le \gamma.t \le n$:*

$$f(\lceil \gamma.t \rceil) \le f(t)/2$$

Notice that Zipf distributions are γ-decreasing. We detail some key properties of step-compatible functions in section 3.1 and of γ-decreasing functions in section 3.2. We then obtain the main result for the Insertion model:

▶ **Theorem 10.** *Let $\varepsilon_1, \varepsilon_2, \delta$, a frequency function f and a stream s with insertions only be given. If the distributions f and g are $(3\varepsilon_1, \varepsilon_2)$-step-compatible and γ-decreasing then Algorithm $A(s, \varepsilon_1, \varepsilon_2, f)$ is a streaming 4δ-Tester that uses space $O(\log^2 n \cdot \log\log n)$.*

3 Properties of the Step-compatible and γ-decreasing functions

The relation $\simeq_\varepsilon$ is reflexive and symmetric and satisfies a variant of the triangle inequality: $a \simeq_\varepsilon b$ and $b \simeq_{\varepsilon'} c$ imply that $a \simeq_{(\varepsilon+\varepsilon'+\varepsilon\varepsilon')} c$. Indeed, the largest gap between a, c is when the $a < b < c$ and the error is:

$$(b-a) + (c-b) \le \varepsilon.a + \varepsilon.b \le \varepsilon.a + \varepsilon'(a + \varepsilon.a) \le (\varepsilon + \varepsilon' + \varepsilon\varepsilon')a = ((1+\varepsilon)(1+\varepsilon') - 1)a.$$

▶ **Lemma 11.** *Let $p_j = (x_j, y_j)$ be a sequence of j_0 points such that $p_j \simeq_{(\varepsilon_j, \eta_j)} p_{j+1}$ for $j = 1, 2, \ldots j_0 - 1$. Then*

$$p_1 \simeq_{(\prod_{1\le j\le j_0}(1+\varepsilon_j)-1, \prod_{1\le j\le j_0}(1+\eta_j)-1)} p_{j_0}.$$

If $\sum_j \varepsilon_j < 1$ and $\sum_j \eta_j < 1$ then

$$p_1 \simeq_{(2\sum_{1\le j\le j_0}\varepsilon_j, 2\sum_{1\le j\le j_0}\eta_j)} p_{j_0}.$$

Proof. Induction on j_0 and standard approximation. ◀

3.1 Properties of step-compatible functions, and Separating rectangles

We will show that functions that are far according to the relative Frechet distance are separated by a certain type of rectangle defined as follows.

▶ **Definition 12.** *We say that such a rectangle* separates *two functions f and g with domain $\{1,\ldots,n\}$ if*

$$\max_{j\in(x,x(1+\varepsilon_1))} g(j) \leq y \quad \textit{and} \quad y(1+\varepsilon_2) \leq \min_{j\in(x,x(1+\varepsilon_1))} f(j)$$

or conversely (exchanging f and g).

In other words, f is below the rectangle R and g is above R. No points $(t, f(t))$ of f or $(t, g(t))$ of g is in R.

Notice that the point $(t, f(t))$ is the left of the rectangle for $t = 1$ and at the right of the rectangle for $t = n$. We now present a central result used by the analysis of the streaming Tester of the subsequent section.

▶ **Theorem 13** (Separation theorem)**.** *If f and g are $(3\varepsilon_1, \varepsilon_2)$-step-compatible and $f \not\sim_{(3\varepsilon_1,3\varepsilon_2)} g$ then there exists an $(\varepsilon_1, \varepsilon_2)$-rectangle which separates f and g.*

The proof is in the appendix B.

3.2 Properties of γ-decreasing functions

Let $F^{res(k)} = \sum_{k+1\leq i\leq n} f_i$ be the tail of the frequency distribution.

▶ **Lemma 14.** *If f is γ-decreasing then*

$$\frac{\varepsilon}{k}.F^{res(k)} \leq \varepsilon.f_k.\frac{2(\gamma-1)}{2-\gamma}$$

Proof. If f is γ-decreasing then for $j \geq 0$:

$$\sum_{i>\gamma^j.k}^{i=\gamma^{j+1}.k} f_i \leq \frac{f_k \cdot (\gamma^{j+1}.k - \gamma^j.k)}{2^j}$$

Hence:

$$F^{res(k)} = \sum_{k+1\leq i\leq n} f_i \leq k.f_k.(\gamma-1).\sum_{j\geq 0}\frac{\gamma^j}{2^j} = k.f_k.(\gamma-1).\frac{1}{1-\gamma/2} = k.f_k.\frac{2(\gamma-1)}{2-\gamma}$$

$$\frac{\varepsilon}{k}.F^{res(k)} \leq \varepsilon.f_k.\frac{2(\gamma-1)}{2-\gamma}$$

◀

We use this bound in section 4.1 to obtain a relative error on the estimation of the Top frequencies.

4 Frequency distributions, the Spacesaving algorithms and a simple lower bound

Given a stream of N elements drawn from a universe U of size n, let f_j denote the frequency (number of occurences) of the jth most frequent element, so that $f_1 \geq f_2 \geq \cdots \geq f_n \geq 0$ and $\sum_{i=1}^n f_j = N$. For example, in the case of a graph given as a stream of m edges, *i.e.* a stream of pairs of vertices, we can define the elements of the stream as the vertices, so the length of the stream is $N = 2m$, and (f_j) is the degree sequence of the graph.

We are particularly interested in frequencies which have a compact representation. For example, *uniform frequencies* where $f_i = N/n$, *Zipf* frequencies (also called heavy-tailed, or

Frequency of the Top k elements for $K = O(k/\varepsilon)$	Error bound
SpaceSaving[18]	$\lvert f_i - c_i \rvert \leq 2\varepsilon.N$
SpaceSaving with strong error Bounds[3]	$\lvert f_i - c_i \rvert \leq \frac{\varepsilon}{k}.F^{res(k)}$
SpaceSaving for $\gamma -$ decreasing frequency functions	$\lvert f_i - c_i \rvert \leq \varepsilon.f_k.\frac{2(\gamma-1)}{2-\gamma} \leq \varepsilon.f_i.\frac{2(\gamma-1)}{2-\gamma}$

Table 1 Error bounds for the top k elements, $i \leq k$, $K = O(k/\varepsilon)$

scale-free, or power-law) with parameter α, where $f_i = cN/i^\alpha$ with $c = 1/\sum_{1\leq j\leq n}(1/j^\alpha)$, and *geometric* frequencies where $f_i = cN/2^i$ with $c = 1/\sum_{1\leq j\leq n} 1/2^j$.

For Zipf frequencies with parameter α the maximum frequency is $f_1 = \Theta(N)$ if $\alpha > 1$ and $f_1 = \Theta(N/\log n)$ if $\alpha = 1$.

4.1 The Spacesaving algorithms

The classical Spacesaving [18] gives a solution to the Top k most frequent elements for the *insertion only* model and an additive error. In [3] a better bound is given, which is a lower bound in the worst-case. We need however to obtain the Top k elements with a relative error and show that it is possible for γ-decreasing frequency functions f, in section A.1 of the appendix A. We can summarize the various previous additive bounds in table 1. If we take the strong bound from [3] and combine it with Lemma 14 of the previous section, we obtain the relative error bound, where for the top-k frequencies f_i where $i \leq k$:

$$\lvert f_i - c_i \rvert \leq \varepsilon.f_k.\frac{2(\gamma-1)}{2-\gamma} \leq \varepsilon.f_i.\frac{2(\gamma-1)}{2-\gamma}$$

The Spacesaving$\pm$ [20] generalizes for the *insertion and α-bounded deletion* model. We will analyse it in some other work. We consider another model, the *sliding window* model, an *insertion and window deletion* model which is not a bounded deletion model in section A.5 of the appendix A. In both cases, we have a solution to the Top-k problem, the building block used by the Tester.

4.2 A lower bound when f is uniform

A classical observation is that in the worst-case, the approximation of $F_\infty = \text{Max}_j\ f_j$ requires space $\Omega(n)$, using a standard reduction from Communication Complexity. [15] reduces the Unique-Disjointness problem for $x, y \in \{0,1\}^n$ to the approximation of F_∞ on a stream s . Another standard problem which requires space $\Omega(n)$ for the One-way Communication complexity is the Index(x,y) problem, see [16], where $x \in \{0,1\}^n$, $y \in \{1,2,...n\}$ and the goal is to compute $x_y \in \{0,1\}$. We write Index$(x,y) = x_y$, as Alice holds x of length n, Bob holds y of length $\log n$ and only Alice can send information to Bob. Notice that we can assume that $\lvert\{i:\ x_i = 1\}\rvert = O(n)$ for example $n/2$, otherwise Alice would directly send these positions to Bob.

We show in the next result a simple reduction from the Index problem to the the *streaming Test problem* which given f and a stream s over the items $a_1, ...a_n$, which defines a frequency g, decides: either $f \sim_{\varepsilon/10} g$ or $f \not\sim_{10\varepsilon} g$ with h.p.

▷ Theorem 1. The streaming Test problem requires space $\Omega(n)$.

Proof. Consider the following reduction from Index to Test. Given $x \in \{0,1\}^n$ and $y \in \{1, 2, ...n\}$ the inputs to Index, let f be the uniform distribution on the a_i such that $x_i = 1$. The stream s is determined by the elements of x of weight 1, followed by the element a_y associated with y, i.e. $a_{i_1}, ...a_{i_k}$ where $x_{i_j} = 1$ and $k = O(n)$, followed by a_y.

If Index$(x, y) = 1$ then the relative frequency g has an element of frequency $2/k$. The point $(1, 1/k)$ of f is far from the closest point $(1, 2/k)$ of g. Hence $f \not\sim_{10\varepsilon} g$.

If Index$(x, y) = 0$ then g is uniform over $k+1$ elements. The points $(i, 1/k)$ of f for $i = 1, 2...k$ are at relative distance $\frac{1/k - 1/(k+1)}{1/k} = 1/(k+1)$ from the closest point $(i, 1/k+1)$ of g for $i = 1, 2...k$. The point $(k+1, 1/(k+1))$ of g is at relative distance $(1/k, 1/(k+1))$ from the point $(k, 1/k)$ of f. Hence $f \sim_{\varepsilon/10} g$ for n large enough.

We reduced a Yes-instance to Index to a No-instance of Test, and a No-instance of Index to a Yes-instance of Test.

As Index requires space $\Omega(n)$, so does the streaming Test problem. ◀

5 Analysis of Algorithm 1, a Streaming Tester

A stream s of N elements of a universe $\{e_1, e_2, \cdots, e_n\}$ of size n determines an integer frequency function g whose domain is $\{1, ...n\}$, such that $g(i)$ is the number of occurences of the ith most frequent element in the stream. Suppose we are given a frequency function f whose domain is $\{1, 2, \cdots, n\}$ in a compact form, such that *Heavy-tail*, power-law or Zipf. We want to verify that the frequencies of elements in a stream approximately follows this law. We propose the following streaming Tester for this problem.

5.1 Analysis of the space used by Algorithm 1

If f is γ-decreasing, we can write: $f(\gamma.t) < f(t)/2$. Hence for $\alpha = \log(1/\varepsilon_2)$ we have

$$f(\gamma^\alpha.t) < f(t)/2^\alpha = \varepsilon_2.f(t)$$

For $n = \gamma^\alpha.t_0$, we find the threshold $t_0 = n/\gamma^{\log(1/\varepsilon_2)}$. For $z_i \leq t_0$, we run the Spacesaving with a table of size $K_i = \frac{4.z_i}{\varepsilon_2.a_i} \cdot \frac{2(\gamma-1)}{2-\gamma} \cdot \frac{\log n}{\delta}$ and for $z_i > t_0$ we do an exact counting.

▶ **Lemma 15.** *Algorithm 1 uses* $O((\log n)^2 \cdot \log\log n)$ *space.*

Proof. For $z_i \leq t_0$, we run the Spacesaving with a table of size K_i where $a_i = \Theta(\varepsilon_1^2.z_i/\log\log n)$. Hence:

$$K_i = \frac{4.z_i}{\varepsilon_2.a_i} \cdot \frac{2(\gamma-1)}{2-\gamma} \cdot \frac{\log n}{\delta} \leq \frac{4 \cdot \log\log n}{\varepsilon_2.\varepsilon_1^2} \cdot \frac{2(\gamma-1)}{2-\gamma} \cdot \frac{\log n}{\delta} = O(\log n \cdot \log\log n)$$

When $z_i > t_0 = n/\gamma^{\log(1/\varepsilon_2)}$, we do an exact counting. In this case, $K_i = n/a_i$. Therefore

$$K_i = n/a_i = n/\varepsilon_1^2.z_i \leq n/\varepsilon_1^2.t_0 < \gamma^{\log(1/\varepsilon_2)}/\varepsilon_1^2$$

In this case, K_i only depends on the parameters $\varepsilon_1, \varepsilon_2$ and γ and is independent of n.

Since we run the algorithm in parallel for $\log_{1+\epsilon_1} n$ values of z_i, for fixed values of $\varepsilon_1, \varepsilon_2$ and γ the total space used is $O((\log n)^2 \cdot \log\log n)$. ◀

5.2 Analysis of the error probability of Algorithm 1

▷ Notation 2. Let $\tilde{e_i}$ be the element whose counter value is c_r, i.e. count$(\tilde{e_i}) = c_r$ and e'_i the element whose rank is r in the stream s_i, for the frequency function g_i, i.e. occ$(e'_i) = g_i(r)$ or

$rank_{s_i}(e'_i) = r$. The functions occ, count, $rank$ are from U to N. We assume that tie-breaking rules are consistent over s and the substreams s_i: $U = \{e^1, e^2, \cdots, e^n\}$ and if two elements e^j and e^k, with $j < k$, have the same number of occurrences, then $rank_s(e^j) < rank_s(e^k)$ and $rank_{s_i}(e^j) < rank_{s_i}(e^k)$ for all substreams.

We recall the following classic Hoeffding probabilistic bound.

▶ **Lemma 16.** *Let $X = \sum_{j=1}^{p} X_i$ where $X_j = 1$ with probability q_j and $X_j = 0$ with probability $1 - q_j$, and the X_j's are independent. Let $\mu = \mathbb{E}(X)$. Then for all $0 < \beta < 1$ we have*

$$\Pr(|X - \mu| > \beta\mu) \leq 2e^{-\mu\beta^2/3}.$$

We now prove the probabilistic Lemma 17, which analyzes the sampling that is used to create the substram s_i and relates e'_i to z_i. This depends on the sampling process alone and not on the Spacesaving algorithm and analysis. The main Lemma 18 guarantees an error bound on Spacesaving on each s_i with high probability.

▶ **Lemma 17.** *Recall that each element is kept in substream s_i with probability $1/a_i$ and that e'_i denotes the element with rank z_i/a_i in substream s_i (when sorted in non-increasing order of number of occurences): $rank_{s_i}(e'_i) = z_i/a_i$. Then, the rank of e'_i in stream s (when sorted in non-increasing order of number of occurences) satisfies*

$$\Pr(z_i(1 - \varepsilon_1^2) \leq rank_s(e'_i) \leq z_i(1 + \varepsilon_1^2)) \geq 1 - 4\delta/\log n.$$

Moreover, if $f = g$ then $f(z_i) \sim_{\varepsilon_2} occ(e'_i)$.

Proof. By definition of e'_i, the rank of $\mathrm{occ}(e'_i)$ in the substream s_i equals $r = z_i/a_i$. We will prove the following: With probability at least $1 - 4\delta/\log n$, the following properties hold:

1. The number of elements that appear in s_i and have rank less than $z_i(1 - \varepsilon_1^2)$ in s is less than z_i/a_i
2. The number of elements that appear in s_i and have rank less than $z_i(1 + \varepsilon_1^2)$ in s is more than z_i/a_i

This will imply the Lemma.

For the first item, we apply Lemma 16 with X denoting the number of elements that appear in s_i and have rank less than $p = z_i(1 - \varepsilon_1^2)$ in s, so that $X_j = 1$ if and only if the element of rank $j \leq z_i(1 - \varepsilon_1^2)$ in s appears in s_i. We have $\mu = z_i(1 - \varepsilon_1^2)/a_i$. We set $\beta = \varepsilon_1^2/(1 - \varepsilon_1^2)$. We obtain that the probability that the statement does not hold is at most $2exp(-\frac{z_i\varepsilon_1^2}{3a_i(1-\varepsilon_1^2)}) \leq 2exp(-\frac{z_i\varepsilon_1^2}{3a_i(1+\varepsilon_1^2)})$.

For the second item, we apply Lemma 16 with X denoting the number of elements that appear in s_i and have rank less than $p = z_i(1 + \varepsilon_1^2)$ in s, so that $X_j = 1$ if and only if the element of rank $j \leq z_i(1 + \varepsilon_1^2)$ in s appears in s_i. We have $\mu = z_i(1 + \varepsilon_1^2)/a_i$. We set $\beta = \frac{\varepsilon_1^2}{(1+\varepsilon_1^2)}$. We obtain that the probability that the statement does not hold is at most $2exp(-\frac{z_i\varepsilon_1^2}{3a_i(1+\varepsilon_1^2)})$.

By the union bound, the probability that the two statements do not both hold is bounded by $4exp(-\frac{z_i\varepsilon_1^2}{3a_i(1+\varepsilon_1^2)})$. Let $a_i = \varepsilon_1^2 z_i/(6\ln((\ln n)/\delta))$. Then this probability is at most $4\delta/\ln n$.

Since $f = g$ and z_i is not close to one of the endpoints of the boxes of f, we also have $f(z_i) \sim_{\varepsilon_2} \mathrm{occ}(e'_i)$. ◀

Now we turn to the analysis of the SpaceSaving algorithm.

▶ **Lemma 18.** *Assume that g is step-compatible and γ-decreasing. Consider Algorithm 1 and recall that $K_i = 4(z_i/a_i) \cdot \frac{2(\gamma-1)}{2-\gamma} \cdot (1-\varepsilon_1^2) \cdot (1+\varepsilon_2) \cdot \frac{\log n}{\varepsilon_2 \delta}$. We have:*

$$\Pr[c_{K_i} \leq \varepsilon_2.g(z_i)] \geq 1 - 5\delta/\log n.$$

Proof. Let g_i be the frequency function of substream i. For table T_i of size K_i used by the algorithm. Let n_i denote the number of distinct elements in stream s_i. Then the domain of g_i is $[1, n_i]$, and n_i is a random variable with expectation equal to n/a_i. Let N_i denote the length of substream s_i: we have $N_i = \sum_{x=1}^{x=n_i} g_i(x)$. Let $G_i(u) = \sum_{j=1}^{u} g_i(j)$ denote the cumulative frequency, and $G_i^{res(u)} = \sum_{j=u+1}^{n_i} g_i(j)$. Let $\widehat{z_i} = z_i/a_i$. We apply Lemma 21 to table T_i, using $u = \widehat{z_i}$ and noting that $K_i - 2\widehat{z_i} > K_i/2$:

$$c_{K_i} \leq \min_{u<K_i/2} \frac{G_i^{res(u)}}{K-2u} \leq \frac{\sum_{\widehat{z_i}+1}^{n_i} g_i(x)}{K_i - 2\widehat{z_i}} \leq \frac{2.}{K_i} \sum_{\widehat{z_i}+1}^{n_i} g_i(x). \tag{2}$$

As in Lemma 17, let e_i' denote the element of substream such that $rank_{s_i}(e_i') = z_i/a_i$. We have:

$$\sum_{\widehat{z_i}+1}^{n_i} g_i(x) = \sum_{y=rank_s(e_i')+1}^{n} g(y)\mathbf{1}(\text{the element of } s \text{ with rank } y \text{ is in } s_i).$$

Let A denote the following event:

$$rank_s(e_i') \geq z_i(1-\varepsilon_1^2)$$

Assume that A holds. Then

$$\sum_{\widehat{z_i}+1}^{n_i} g_i(x) \leq \sum_{y=z_i(1-\varepsilon_1^2)+1}^{n} g(y)\mathbf{1}(\text{the element of } s \text{ with rank } y \text{ is in } s_i)$$

Observe that the value of the right-hand side is determined by which elements of s are put in s_i, among the ones with $rank_s$ greater than $z_i(1-\varepsilon_1^2)$. Also observe that event A is determined by how many elements of s are put in s_i, among the ones with $rank_s$ smaller than or equal to $z_i(1-\varepsilon_1^2)$. Thus the expression in the right-hand side is independent of event A, and we can write:

$$\begin{aligned}
\mathbb{E}[\sum_{\widehat{z_i}<x\leq n_i} g_i(x)|A] &\leq \mathbb{E}[\sum_{y=z_i(1-\varepsilon_1^2)+1}^{n} g(y)\mathbf{1}(\text{the element of } s \text{ with rank } y \text{ is in } s_i)|A] \\
&= \mathbb{E}[\sum_{y=z_i(1-\varepsilon_1^2)+1}^{n} g(y)\mathbf{1}(\text{the element of } s \text{ with rank } y \text{ is in } s_i)] \\
&= \frac{1}{a_i} \sum_{y=z_i(1-\varepsilon_1^2)+1}^{n} g(y)
\end{aligned}$$

Now, since g is γ-decreasing, applying Lemma 14 to $g(z_i(1-\varepsilon_1^2))$ and rewriting, we have:

$$\sum_{y=z_i(1-\varepsilon_1^2)+1}^{n} g(y) \leq \frac{2(\gamma-1)}{2-\gamma}.z_i(1-\varepsilon_1^2).g(z_i(1-\varepsilon_1^2)) \tag{3}$$

Since z_i is not close to a Box endpoint of g, by Lemma 8 we have

$$g(z_i(1-\varepsilon_1^2)) \le g(z_i)(1+\varepsilon_2).$$

Combining the inequalities (2) and (3) gives:

$$\mathbb{E}[c_{K_i}|A] \le \frac{2}{K_i} \cdot \frac{1}{a_i} \cdot \frac{2(\gamma-1)}{2-\gamma} z_i(1-\varepsilon_1^2) \cdot g(z_i)(1+\varepsilon_2).$$

As $K_i = 4(z_i/a_i) \cdot \frac{2(\gamma-1)}{2-\gamma} \cdot (1-\varepsilon_1^2) \cdot (1+\varepsilon_2) \cdot \frac{\log n}{\varepsilon_2 \delta}$, we have:

$$\mathbb{E}[c_{K_i}|A] \le \frac{\delta}{\log n}.\varepsilon_2.g(z_i)$$

We use Markov's inequality to conclude that, conditioned on event A we have:

$$\Pr(c_{K_i} \le \frac{\log n}{\delta} \cdot \mathbb{E}[c_{K_i}|A] \ \mid \ A] \ge 1 - \delta/\log n.$$

By Lemma 17 event A has probability at least $1 - 4\delta/\log n$. We conclude that

$$\Pr[c_{K_i} \le \varepsilon_2.g(z_i))] \ge (1-4\delta/\log n)(1-\delta/\log n) \ge 1 - 5\delta/\log n.$$

◀

We can now prove our main Theorem:

▷ Theorem 10. Let $\varepsilon_1, \varepsilon_2, \delta$, a frequency function f and a stream s with insertions only be given. If the distributions f and g are $(3\varepsilon_1, \varepsilon_2)$-step-compatible and γ-decreasing then Algorithm $A(s, \varepsilon_1, \varepsilon_2, f)$ is a streaming 4δ-Tester that uses space $O(\log^2 n \cdot \log\log n)$.

Proof. First, we assume that $f = g$ and aim to prove that the algorithm outputs YES with probability $1 - O(\delta)$. To that end, for each i such that z_i is not ε_1^2-close to a Box endpoint, we will prove that with probability at least $1 - O(\delta/\log n)$ we have $|g(z_i) - c_r| \le 3\varepsilon_2 g(z_i)$, and then apply the union bound. We conclude that $c_r \simeq_{3\varepsilon_2} f(z_j)$ and the test is positive with high probability.

Focus on one value of i such that z_i is not ε_1^2-close to a Box endpoint of f, and consider the substream s_i. We first write:

$$|g(z_i) - c_r| \le |g(z_i) - \mathrm{occ}(e'_i)| + |\mathrm{occ}(e'_i) - \mathrm{count}(e'_i)| + |\mathrm{count}(e'_i) - \mathrm{count}(\tilde{e}_i)| \tag{4}$$

and analyze the right-hand side term by term.

First we will prove that with probability $1 - 4\delta/\log n$ we have

$$|g(z_i) - \mathrm{occ}(e'_i)| \le \varepsilon_2 g(z_i). \tag{5}$$

To that end, we let $I = [z_i/(1+\varepsilon_1^2), z_i(1+\varepsilon_1^2)]$. Since g is step-compatible and z_i it is not ε_1^2-close to a Box endpoint, g is near-constant inside the entirety of interval I: the maximum exceeds the minimum by a $(1+\varepsilon_2)$ factor at most. By Lemma 17, with probability at least $1 - 4\delta/\log n$ we have that $\mathrm{rank}_s(e'_i)$ is inside I, hence Equation 5.

Secondly, we observe that by Property 3 of Spacesaving (see page 19),

$$|\mathrm{occ}(e'_i) - \mathrm{count}(e'_i)| \le c_{K_i}. \tag{6}$$

Thirdly, we will argue that

$$|\mathrm{count}(e'_i) - \mathrm{count}(\tilde{e}_i)| \le c_{K_i}. \tag{7}$$

To that end, we refer the reader to Figure 1. By Property 3, for any element e of s_i we have $\mathrm{occ}(e) \leq \mathrm{count}(e) \leq \mathrm{occ}(e) + c_{K_i}$, so when we plot the points $(\mathrm{occ}(e), \mathrm{count}(e))$ for the elements occuring in stream s_i, all points are inside the strip of equation $x \leq y \leq x + c_{K_i}$. Consider the point $(\mathrm{occ}(e'_i), \mathrm{count}(\tilde{e}_i))$. We partition the strip into three parts (see Figure 1):

1. P_1 consisting of the points (x, y) such that $x > \mathrm{count}(\tilde{e}_i)$. Since $\tilde{e}_i$ has rank r according to count, there are at most $r - 1$ points in P_1.
2. P_2 consisting of the points (x, y) such that $x < \mathrm{count}(\tilde{e}_i) - c_{K_i}$. Since $\tilde{e}_i$ has rank r according to count, there are fewer than $n_i - r$ where n_i is the number of elements in the stream s_i.
3. P_3 consisting of the rest. All points of P_1 have occ value larger than all points of P_3, and all points of P_2 have occ value smaller than all points of P_3.

Recall that e'_i has rank r according to occ. Thus the point $(\mathrm{occ}(e'_i), \mathrm{count}(\tilde{e}_i))$ cannot be in P_1 nor in P_2. This implies that e'_i is in P_3, hence Equation 7.

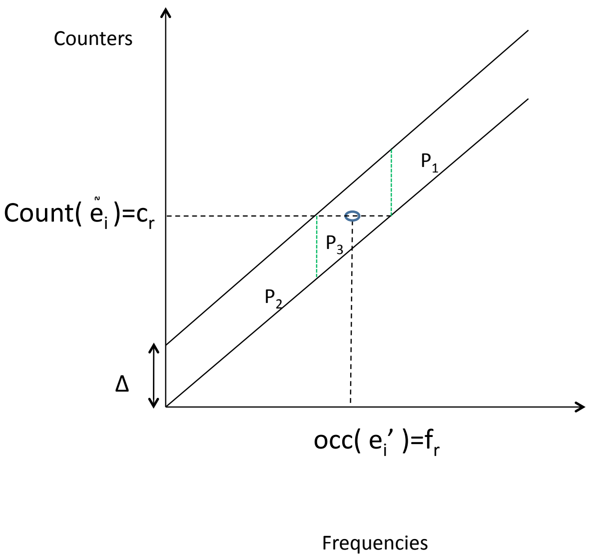

Figure 1 Counters and Frequencies for a stream s_i. The error $\Delta = c_{K_i}$ and $|\mathrm{occ}(e'_i) - \mathrm{count}(\tilde{e}_i)| < c_{K_i}$.

Finally, we apply Lemma 18: with probability at least $1 - 5\delta/\log n$ we have $c_{K_i} \leq \varepsilon_2 g(z_i)$. Combining with Equations 4,5,6 and 7 we obtain that with probability at least $1 - 9\delta/\log n$ we have $|g(z_i) - \mathrm{occ}(e'_i)| \leq 3\varepsilon_2 g(z_i)$. By the union bound, with probability at least $1 - O(\delta)$ test test is positive and Algorithm 1 outputs YES, as desired.

Assume that g is far from f, i.e. $f \not\sim_{(20\varepsilon_1, 20\varepsilon_2)} g$. By Theorem 13 there exists a separating rectangle $R = [b, b(1 + 6\varepsilon_1)] * [c, c(1 + 6\varepsilon_2)]$ which separates f from g.

Let j be the smallest integer such that $b(1 + 3.\varepsilon_1) < z_j = (1 + \varepsilon_1^2)^j$. Consider the streams s_j or s_{j-1} or s_{j+1} so that z_j avoids the limits of the Boxes of f and g.

As the relative width $(1 + 3\varepsilon_1)$ is larger than $(1 + \varepsilon_1^2)$, the point z_j is close to the center on the x-axis of the separating rectangle R. Consider the two cases, f is above the rectangle (case 1) or f is below the rectangle (case 2).

• Assume that g is below R and f is above R (case 1). The value c_r is the count of an element $\tilde{e_j}$ which with high probability is close to $\mathrm{occ}(e'_j)$ for an element e'_j of the stream s_j. The triangle inequality gives:

$$|c_r - f(z_j)| \geq |\mathrm{occ}(e'_j) - f(z_j)| - |\mathrm{occ}(e'_j) - c_r|$$

By equations (6) and (7): $|\mathrm{occ}(e'_j) - c_r| \leq |\mathrm{occ}(e'_i) - \mathrm{count}(e'_i)| + |\mathrm{count}(e'_i) - \mathrm{count}(\tilde{e_i})| \leq 2.c_{K_i}$ and by Lemma 18 with high probability:

$$|\mathrm{occ}(e'_j) - c_r| \leq 2\varepsilon_2.g(z_j)$$

Because g is below the rectangle R, then $|\mathrm{occ}(e'_j) - f(z_j)| \geq 6\varepsilon_2.g(z_j)$. Then with high probability:

$$|c_r - f(z_j)| \geq 6\varepsilon_2.g(z_j) - 2\varepsilon_2.g(z_j) \geq 4\varepsilon_2.g(z_j) \geq 3\varepsilon_2.c_r$$

Hence $c_r \not\simeq_{3\varepsilon_2} f(z_j)$ with high probability as $c_r \leq f(z_j)$, so the algorithm will reject, as desired.

• Assume that f is below R and g is above R (case 2). Select the position of the separating rectangle $R = [b, b.(1+6\varepsilon_1)] * [c_L, c_L.(1+6\varepsilon_2)]$ so that the top of the rectangle coincides with the bottom of the Box of $g(z_j)$. Notice that $c_L \geq f(z_j)$. As z_j is not close to the limits of the Boxes of f and g, we can make the separating rectangle narrower, i.e. $R' = [b, b.(1+\varepsilon_1^2)] * [c_L, c_L.(1+6\varepsilon_2)]$

We can therefore write: $g(z_j) \leq c_L.(1+6\varepsilon_2).(1+\varepsilon_2) \simeq c_L.(1+7\varepsilon_2)$. Hence:

$$-2\varepsilon_2.g(z_j) \geq -2\varepsilon_2.c_L.(1+7\varepsilon_2) \tag{8}$$

The previous triangle inequality gives:

$$|c_r - f(z_j)| \geq |\mathrm{occ}(e'_j) - f(z_j)| - |\mathrm{occ}(e'_j) - c_r|$$

As $|\mathrm{occ}(e'_j) - c_r| \leq 2.\varepsilon_2.g(z_j)$ by Lemma 18 with high probability as in case 1, and f is below the rectangle R', we can then bound $|\mathrm{occ}(e'_j) - f(z_j)| \geq 6\varepsilon_2.c_L$. Then, with high probability, using the inequality (8):

$$|c_r - f(z_j)| \geq 6\varepsilon_2.c_L - 2\varepsilon_2.g(z_j) \geq 6\varepsilon_2.c_L - 2\varepsilon_2.c_L.(1+7\varepsilon_2) \geq 3\varepsilon_2.c_L \geq 3\varepsilon_2.f(z_j)$$

Hence $c_r \not\simeq_{3\varepsilon_2} f(z_j)$ with high probability as $c_r \geq f(z_j)$, so the algorithm will reject, as desired.

◀

5.3 Streaming δ-Tester for sliding windows

Theorem 10 can be extended to the sliding windows model defined in the Appendix A.5. We want to test if the last window defined by the parameters λ, Δ follows a frequency function f.

▶ **Corollary 19.** *If f and g are $(3\varepsilon_1, \varepsilon_2)$-step-compatible and γ-decreasing in each window, then Algorithm $A(s, \varepsilon_1, \varepsilon_2, f)$ is a streaming 4δ-Tester which uses uses space $O(\log^2 n \cdot \log\log n)$.*

Proof. As f is γ-decreasing, we apply Lemma 14 to the Spacesaving version of the sliding window (see Appendix A.5) and obtain the relative error $|f_k - c_k| \leq \varepsilon.f_k.\frac{2(\gamma-1)}{2-\gamma}$. Both Lemmas 17 on the sampling and 18 on Spacesaving generalize. Hence the main Theorem in section 5.2 also applies. ◀

6 Conclusion

We introduced a scale free distance between two frequency distributions, the relative version of the Fréchet distance. We then studied how to verify a frequency distribution g defined by a stream of N items among n distinct items. We first proved a $\Omega(n)$ lower bound on the space required in general. If we assume that the frequency distribution f and the frequency g defined by the stream satisfy a step-compatibility condition and decrease fast enough, we presented a Tester that uses $O(\log^2 n \cdot \log\log n)$ space. Zipf and Power law distributions are both step-compatible and γ-decreasing.

References

1. Pankaj K. Agarwal, Graham Cormode, Zengfeng Huang, Jeff M. Phillips, Zhewei Wei, and Ke Yi. Mergeable summaries. *ACM Transactions on Database Systems*, 38(4), 2013.
2. Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58(1):137–147, 1999.
3. Radu Berinde, Piotr Indyk, Graham Cormode, and Martin J. Strauss. Space-optimal heavy hitters with strong error bounds. *ACM Trans. Database Syst.*, 35(4), 2010.
4. Blum Manuel, Luby Michael, and Rubinfeld Ronitt. Self-testing/correcting with applications to numerical problems. *Journal of Computer and System Sciences*, 1993.
5. Clément L. Canonne. *A Survey on Distribution Testing: Your Data is Big. But is it Blue?* Theory of Computing Library, 2020.
6. Amit Chakrabarti, Graham Cormode, Andrew McGregor, Justin Thaler, and Suresh Venkatasubramanian. Verifiable stream computation and arthur–merlin communication. *SIAM Journal on Computing*, 48(4):1265–1299, 2019.
7. Graham Cormode, Zohar S. Karnin, Edo Liberty, Justin Thaler, and Pavel Veselý. Relative error streaming quantiles. *JACM*, abs/2004.01668, 2023.
8. Anne Driemel, Ioannis Psarros, and Melanie Schmidt. Sublinear data structures for short frechet queries. *CoRR*, abs/1907.04420, 2019.
9. Thomas Eiter and Heikki Mannila. Computing discrete frechet distance. In *Tech. Report CD-TR 94/64, Christian Doppler Laboratory for Expert Systems, TU Vienna, Austria*, 1994.
10. Emily Farrow, Junbo Li, Farhan Zaki, and Ashwin Lall. Accessible streaming algorithms for the chi-square test. In *SSDBM*. Association for Computing Machinery, 2020.
11. Joan Feigenbaum, Sampath Kannan, Martin J. Strauss, and Mahesh Viswanathan. Testing and spot-checking of data streams. *Algorithmica*, 34(1):67, 2002.
12. Arnold Filtser and Omrit Filtser. Static and streaming data structures for fréchet distance queries. *CoRR*, abs/2007.10898, 2020.
13. Chris Hickey and Graham Cormode. Cheap checking for cloud computing: Statistical analysis via annotated data streams. In Amos Storkey and Fernando Perez-Cruz, editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 1318–1326. PMLR, 2018.
14. Rajesh Jayaram and David P. Woodruff. Data streams with bounded deletions. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS '18, page 341?354. Association for Computing Machinery, 2018.
15. Akshay Kamath, Eric Price, and David P. Woodruff. A simple proof of a new set disjointness with applications to data streams. In *Proceedings of the 36th Computational Complexity Conference*, 2021.
16. Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1996.

17 Gurmeet Singh Manku and Rajeev Motwani. Chapter 31 - approximate frequency counts over data streams. In Philip A. Bernstein, Yannis E. Ioannidis, Raghu Ramakrishnan, and Dimitris Papadias, editors, *VLDB '02: Proceedings of the 28th International Conference on Very Large Databases*, pages 346–357. Morgan Kaufmann, San Francisco, 2002.

18 Ahmed Metwally, Divyakant Agrawal, and Amr El Abbadi. Efficient computation of frequent and top-k elements in data streams. In *Proceedings of the 10th International Conference on Database Theory, ICDT'05*, pages 398–412. Springer-Verlag, 2005.

19 Senthilmurugan Muthukrishnan, Martin Strauss, and Xian Zheng. Workload-optimal histograms on streams. pages 734–745, 07 2005.

20 Fuheng Zhao, Divyakant Agrawal, Amr El Abbadi, and Ahmed Metwally. Spacesaving±: An optimal algorithm for frequency estimation and frequent items in the bounded-deletion model. *Proc. VLDB Endow.*, 15(6):1215?1227, 2022.

A Appendix A: The Spacesaving algorithms

A.1 The SpaceSaving algorithm with insertions only [18]

The SpaceSaving algorithm introduced in [18] computes an approximation of the frequencies of the k most frequent items elements in a stream. It uses a table T of triplets $T[j] = (e, c_j, \varepsilon_j)$ where $e \in A$ is an element of the universe $A = \{e_1, e_2, \cdots, e_n\}$, $c_j \in N$ is a counter approximating the number of occurences of element e and $\varepsilon_j \in N$, $\varepsilon_j < c_j$ is a bound on the error between the counter and the correct number of occurences of e in the stream. The table, of size K, is ordered by counters: $c_1 \geq c_2, ... \geq c_K$. Assume $k < K$.

```
Algorithm Top-k(k, K)
Data: a stream s of length N, from a universe A = {e_1, e_2, ..., e_n}.
T[j] ← (−, 0, 0) for every j ∈ [1, K];
while stream S is flowing do
    read next element e of S;
    if e is in the table T at position j then
        increment c_j;
    else
        Replace T[K] = (e', c_K, ε_K) by T[K] = (e, c_K + 1, c_K) ;
        Reorder T by non-increasing values of c_j ;
    end
end
Result: the sequence S of the first k elements
```

Algorithm 2: The Top-k algorithm

▷ Notation 3. For a table position $j \in [1, K]$ and an element $e \in A$, let $\sigma(j) = i$ if $T[j] = (e, c_j, \varepsilon_j)$ and the frequency of element e is f_i.

Thus $f_{\sigma(i)}$ is the frequency associated with the element whose counter is c_i. Algorithm Top-k guarantees that σ is injective. In the ideal case in which $\sigma(i) = i$ for all $i \in [1, K]$, then T contains the K most frequent elements of A, ordered by non-increasing frequency. Algorithm Top-k satisfies the following properties:

1. $\sum_{1 \leq j \leq K} c_j = N$
2. For all $j \leq K$, $\epsilon_j \leq c_K$.
3. For all $j \leq K$, $c_j - \epsilon_j \leq f_{\sigma(j)} \leq c_j$.
4. For each element $e \in A$ not in T, i.e. for any index $i \notin Im(\sigma)$: $f_i \leq c_K$.

The size K of the table can be tuned to provide the approximate Top-k elements or the exact Top-k elements, in some special cases. Let S^* be the set of top k most frequent elements. The following Lemma is implicitly present in [18].

▶ **Lemma 20.** *(adapted from [18])*

1. *(Exact result) If $c_K \leq f_k - f_{k+1}$, then the Top-k algorithm gives the exact solution S^*.*
2. *(Approximate result) If $c_K \leq \varepsilon.f_k$, then S contains every element e_i such that $f_i \geq (1 + \varepsilon).f_k$ and no element e_i such that $f_i \leq (1 - \varepsilon).f_k$.*

Proof. Assume $c_K \leq f_k - f_{k+1}$. From property 4, if $e \in A$ is not in the table T, its frequency $f_i \leq c_K$. As $c_K \leq f_k - f_{k+1} < f_k$, hence $f_i < f_k$ and $e \notin S^*$. Let us show that if $e \in T - S$, then $e \notin S^*$.

Let i, j two elements of T such that $f_i > f_j + c_K$. The corresponding counters $c_{\sigma^{-1}(i)}$ and $c_{\sigma^{-1}(j)}$ are in the right order, i.e.

$$c_{\sigma^{-1}(i)} > c_{\sigma^{-1}(j)}$$

Apply properties 2 and 3:

$$c_{\sigma^{-1}(i)} \geq f_i > f_j + c_K \geq f_j + \varepsilon_j \geq c_{\sigma^{-1}(j)}$$

If $i \in \{1, 2, ...k\}$ and $j \notin \{1, 2, ...k\}$, then:

$$f_i - f_j > f_k - f_{k+1} \geq c_K$$

Hence the counters $c_{\sigma^{-1}(1)}, ...c_{\sigma^{-1}(k)}$ are all greater than the counters $c_{\sigma^{-1}(j)}$ for $j > k$. Hence $S = S^*$.

Assume $c_K \leq \varepsilon.f_k$, the figure 2 shows that if $f_i < (1-\varepsilon)f_k$ then $c_{\sigma^{-1}(i)}$ is smaller than all the counters of elements of S^*, hence $i \notin S$. If $f_j > (1+\varepsilon)f_k$, then $c_{\sigma^{-1}(j)}$ is larger than all the counters of elements of $A - S^*$, hence $i \in S$. ◀

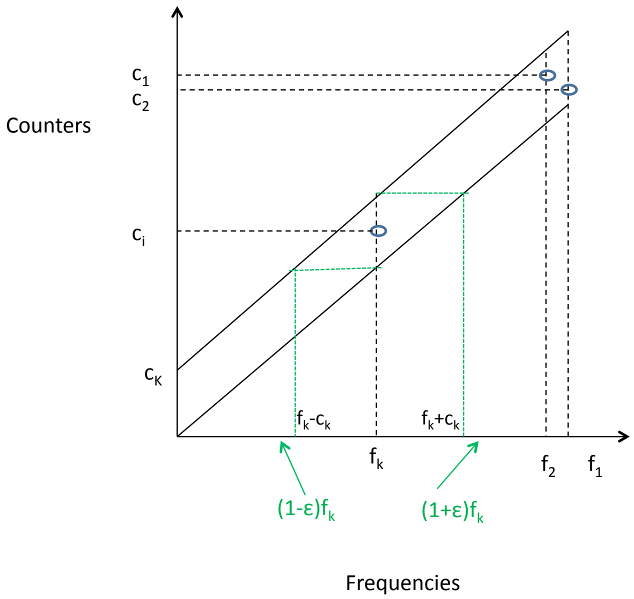

Figure 2 Frequencies-Counters relation: for each $1 \leq k \leq n$, the i-th element of the table T is (e, c_i, ε_i) where c_i is the i-th counter and $\sigma(i) = k$. Then $f_k \leq c_i \leq f_k + \varepsilon_k \leq f_k + c_K$. By properties 2 and 3 the points $(f_k, c_{\sigma^{-1}(k)})$ are above the diagonal and below the diagonal shifted by c_K.

When the table T of size $K > k$ is such that:

$$c_K \leq f_k - f_{k+1} \tag{2}$$

Lemma 20 for the condition (2) guarantees that the Top-k algorithm gives an exact solution. In fact the k first elements of the table T are in the right order. In the original paper[18] , Lemma 1 bounded the additive error c_K using a simple averaging argument.

Notice that $c_K \leq N/K$ by the uniform bound on the minimum value, hence $K = O(\frac{1}{\varepsilon})$, then $c_K \leq \varepsilon.N$ and the frequency f_i can be approximated with an additive error less than $\varepsilon.N$. If we want a relative error for the Top-k algorithm, i.e the hypothesis $c_K \leq \varepsilon.f_k$ of lemma 20, we need to use the γ-decreasing hypothesis.

A.2 A tighter analysis of the SpaceSaving algorithm

Here, we prove the following improvement to Lemma 1's bound on c_K, which is a special case when $u = 0$. In [3], a specific stream shows that the bound is tight.

Consider the cumulative distribution of frequencies, denoted by $F_t = \sum_{1\leq i\leq t} f_i$ and $F_0 = 0$ and the residual cumulative distribution of frequencies $F^{res(t)} = \sum_{t+1\leq i\leq n} f_i$

▶ **Lemma 21.** *(inspired from [3]) Let K denote the size of the table and c_K be an integer as defined in the Space Saving algorithm. Then*

$$c_K \leq \min_{u<K/2} \frac{F^{res(u)}}{K-2u}$$

Proof. Let u be an integer in interval $[1, K/2[$. To prove the Lemma, it suffices to argue that $c_K \leq \frac{F^{res(u)}}{K-2u}$. The minimum value c_K of the counters is less than the average of the counters over the interval $[u+1, K]$, so (using properties 1 and 3):

$$c_K \leq \frac{\sum_{j=u+1}^{K} c_j}{K-u} = \frac{N - \sum_{j=1}^{u} c_j}{K-u} \leq \frac{N - \sum_{j=1}^{u} f_\sigma(j)}{K-u}.$$

Notice that $f_{\sigma(1)} + f_{\sigma(2)} + ...f_{\sigma(u)} \leq f_1 + f_2 + ...f_u$ because (f_j) is a non-increasing sequence.

Let us prove that for each i,

$$f_{\sigma(i)} \geq f_i - c_K$$

For every $j \in [1, i]$ we have $f_j \leq c_{\sigma^{-1}(j)}$ by Property 3. Hence $f_i = \min_{1\leq j\leq i} f_j \leq \min_{1\leq j\leq i} c_{\sigma^{-1}(j)}$. But by definition of c_i, $\min_{1\leq j\leq i} c_{\sigma^{-1}(j)} \leq c_i$, and by Property 3 again, $c_i \leq f_{\sigma(i)} + c_K$. Therefore $f_{\sigma(i)} \geq f_i - c_K$.

Hence: $f_1 - c_K + f_2 - c_K + ...f_u - c_K \leq f_{\sigma(1)} + f_{\sigma(2)} + ...f_{\sigma(u)}$ and:

$$N - \sum_{j=1}^{u} f_\sigma(j) \leq N - \sum_{j=1}^{u} f_j + u.c_K$$

$$c_K \leq \frac{N - \sum_{j=1}^{u} f_\sigma(j)}{K-u} \leq \frac{N - \sum_{j=1}^{u} f_j + u.c_K}{K-u} = \frac{N - \sum_{j=1}^{u} f_j}{K-u} + \frac{u.c_K}{K-u}$$

$$c_K.\frac{K-2u}{K-u} \leq \frac{N - \sum_{j=1}^{u} f_j}{K-u} = \frac{\sum_{j=u+1}^{n} f_j}{K-u} = \frac{F^{res(u)}}{K-u}$$

If $u < K/2$ then $K - 2u > 0$:

$$c_K \leq \frac{F^{res(u)}}{K-2u}$$

As this true for all $u < K/2$, then

$$c_K \leq \min_{u<K/2} \frac{F^{res(u)}}{K-2u}$$

◀

A.3 Application to Zipf distributions

Assume a Zipf distribution of parameter $\alpha > 1$: $f_i = cN/i^\alpha$, where $c = 1/(\sum_{i=1}^{n} 1/i^\alpha)$. We apply Lemma 21 to upper bound the main uncertainty parameter c_K.

▶ **Lemma 22.** *Let K denote the size of the table. Then:*

$$c_K \leq \frac{\Theta(N)}{K^\alpha}.$$

Proof. Since $\alpha > 1$, we have $c = \Theta(1)$ as $n \to \infty$.

$$F^{res(u)} = cN. \sum_{i=u+1}^{n} \frac{1}{i^\alpha}$$

$$\int_{u+1}^{n} \frac{dx}{x^\alpha} \leq \sum_{i=u+1}^{n} \frac{1}{i^\alpha} \leq \int_{u}^{n} \frac{dx}{x^\alpha}$$

$$F^{res(u)} \leq \frac{\Theta(N)}{u^{\alpha-1}}$$

By lemma 21, for $u < K/2$, $c_K \leq \frac{F^{res(u)}}{K-2u}$. Hence for $u = K/3$:

$$c_K \leq \frac{F^{res(K/3)}}{K/3} \leq \frac{\Theta(N)}{K^\alpha}$$

◀

We need to analyse the size of K in the Top-k algorithm 2 as a function of k for Zipf distributions.

$$f_k - f_{k+1} = cN.(\frac{1}{k^\alpha} - \frac{1}{(k+1)^\alpha}) \simeq cN.\frac{k^{\alpha-1}}{k^{2\alpha}} = \frac{cN}{k^{\alpha+1}}$$

By lemma 1, $c_K \leq \frac{N}{K}$, the uniform average. If

$$c_K \leq \frac{N}{K} \leq \frac{cN}{k^{\alpha+1}}$$

the condition (2) on $f_k - f_{k+1}$ is guaranteed and we have an exact solution. Hence:

$$K = \Omega(k^{\alpha+1})$$

The new analysis of lemma 21 gives a better bound on K.

▶ **Lemma 23.** *For the Zipf distribution with parameter $\alpha > 1$, $K = \Omega(k^{1+1/\alpha})$ guarantees an exact solution.*

Proof. By lemma 22, $c_K \leq \frac{\Theta(N)}{K^\alpha}$ hence if:

$$c_K \leq \frac{\Theta(N)}{K^\alpha} \leq \frac{\Theta(N)}{k^{\alpha+1}}$$

the condition (2) is guaranteed. Hence

$$K \geq \Theta(k^{1+1/\alpha})$$

◀

A similar bound is given in [18], by arguing that $f_i < c_K$ for $i > K$. In particular for $i = K+1$:

$$f_{K+1} = \frac{cN}{(K+1)^\alpha} \leq c_K \leq \frac{cN}{k^{\alpha+1}}$$

which gives $K \geq \Theta(k^{1+1/\alpha})$.

For an approximate solution, we take a table T such that:

$$c_K \leq \varepsilon . f_k \tag{2}$$

▶ **Lemma 24.** *For the Zipf distribution with parameter $\alpha > 1$, $K = \Omega(k.(\frac{1}{\varepsilon})^{1/\alpha})$ guarantees an approximate solution.*

Proof. By lemma 21, $c_K \leq \frac{\Theta(N)}{K^\alpha}$ hence if:

$$c_K \leq \frac{\Theta(N)}{K^\alpha} \leq \varepsilon . f_k = \varepsilon . cN. \frac{c}{k^\alpha}$$

the condition (2) is guaranteed. Hence:

$$K \geq \Omega(k.(\frac{1}{\varepsilon})^{1/\alpha})$$

◀

A.4 The SpaceSaving algorithm $\pm$ [20]

This algorithm introduced in [20] computes an approximation of the frequencies of the k most frequent items elements in a stream of insertions and deletions, with the *bounded deletions hypothesis* [14]. If D is the number of deletions and I the number of insertions, then $D \leq (1 - 1/\alpha)I$, for some constant $\alpha \geq 1$. We will analyze this model in some other publication.

A.5 The SpaceSaving algorithm for sliding windows

Given a stream s of items, we may want to test the frequency g in a time interval $[\tau_i, \tau_i + \Delta]$ of width Δ, where τ_i is a timestamp, $\tau_{i+1} = \tau_i + \lambda$ and λ, the shift, divides Δ. Assume we want to test the frequency g of the last window of the stream. Notice that this model does not follow the *bounded deletion* hypothesis: for the last window, $I - D$ can be small and not larger than I/α for some constant α. The error of the SpaceSaving$\pm$ algorithm accumulates for each window over the time and can't correctly approximate the Top-k elements in the last window.

Suppose without loss of generality that $\lambda = \Delta/2$ and consider Blocks B_i of the stream for the time intervals $[\tau_i, \tau_i + \lambda[$. Each window consists of two consecutive Blocks. Assume the last entry e_N ends the Block B_i. We apply the Spacesaving for each Block B_i but only keep the last two tables T_{i-1} and T_i. The Top-k elements of the last window uses the *merge* of the last two tables, defined below. We then read the next Block B_{i+1}, construct T_{i+1}, remove T_{i-1} and use the merge of T_i and T_{i+1}, as in [1].

Algorithm Top$_{sw}$-k(k, K, λ, Δ)
Data: a stream S of length N, from a universe $A = \{e_1, e_2, \ldots, e_n\}$.
$p = \Delta/\lambda$;
while *stream S is flowing* **do**
  read next Block B_{i+1} of S and build T_{i+1} by Spacesaving;
  maintain $T_{i-p+1}, ...T_i$ built by Spacesaving for the previous Blocks ;
  when B_{i+1} is read, remove the oldest table T_{i-p+1} and keep T_{i-p}, T_{i+1};
  i=i+1 ;
end
Result: the sequence S of the first k elements of the Merge of the last p tables

Algorithm 3: The Top$_s w$-k algorithm, or SpaceSaving± algorithm

Each Block B_i, with N_i elements and a Table T_i of size K satisfies the Spacesaving invariants, with the index i: $f^i_{\sigma(j)}$, ε^i_j,c^i_j are the frequency, counter, error of the j-th element of the table.

1. $\sum_{1\leq j\leq K} c^i_j = N_i$
2. For all $j \leq K_i$, $\varepsilon^i_j \leq c_{K_i}$.
3. For all $j \leq K_i$, $c^i_j - \varepsilon^i_j \leq f^i_{\sigma(j)} \leq c^i_j$.
4. For each element $e \in A$ not in T_i, i.e. for any index $j \notin Im(\sigma)$: $f^i_j \leq c_{K_i}$.

We can merge T_i and T_{i-1} into a large T of size K at most $K_{i-1} + K_i$ as follows:

Merge of T_{i-1}, T_i into T.

1. for items j both in T_{i-1}, T_i, $c_j = c^{i-1}_j + c^i_j$ and $\varepsilon_j = \varepsilon^{i-1}_j + \varepsilon^i_j$. For all $j \leq K$, then $\epsilon_j \leq c_{K_{i-1}} + c_{K_i}$.
2. for items j in T_i and not in T_{i-1}, $c^i_j - \varepsilon^i_j \leq f_{\sigma(j)} \leq c^i_j + c_{K_{i-1}}$
3. for items j in T_{i-1} and not in T_i, $c^{i-1}_j - \varepsilon^{i-1}_j \leq f_{\sigma(j)} \leq c^{i-1}_j + c_{K_i}$
4. For each element $e \in A$ not in T_{i-1} nor in T_i, i.e. for any index $j \notin Im(\sigma)$: $f_j \leq c^{i-1}_K + c^i_K$.

Notice that $\sum_{1\leq j\leq K} c_j = N_{i-1} + N_i$ and therefore we satisfy the invariants of a Block with different parameters. We can then obtain a result similar to lemma 20.

▶ **Lemma 25.** *If $c_K \leq \varepsilon.f_k$, then S contains every element e_i such that $f_i \geq (1+\varepsilon).f_k$ and no element e_i such that $f_i \leq (1-\varepsilon).f_k$.*

Proof. The proof is as in lemma 20, as the new table T, obtained by the merge, follows the same invariant conditions, as the Spacesaving algorithm. ◀

B Appendix B: proof of the Separation theorem

We start the proof by establishing the following Lemma.

▶ **Lemma 26** (Distance lemma). *If f and g are two functions describing frequencies and such that for every point of f, there is a point of g which is $(\varepsilon_1, \varepsilon_2)$-close and conversely. Then*

$$f \sim_{(\varepsilon_1,\varepsilon_2)} g$$

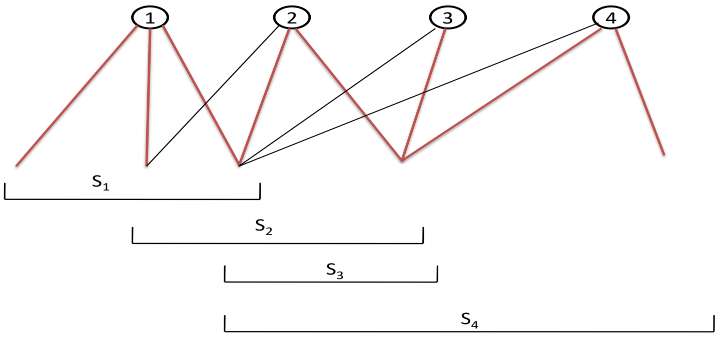

Figure 3 Proof of lemma 26. The thick red edges are the coupling edges.

Proof. Given a point $u_i = (i, f(i))$ of f, we first claim that the set S_i of j such that $v_j = (j, g(j))$ satisfies $v_j \simeq_{(\varepsilon_1,\varepsilon_2)} u_i$ is an non-empty interval, as shown in figure 3. Indeed, it is non-empty by assumption. Let $j_{\min}$ and $j_{\max}$ be its minimum and maximum elements respectively. Then for every $j \in [j_{\min}, j_{\max}]$, we have $i/(1+\varepsilon_1) \le j_{\min} \le j$ and $j \le j_{\max} \le i(1+\varepsilon_1)$, so $j \simeq_{\varepsilon_1} i$; and by monotonicity, $f(j) \le f(j_{\min}) \le (1+\varepsilon_2)g(i)$ and $g(i)/(1+\varepsilon_2) \le g(j_{\max}) \le g(j)$, so $f(i) \simeq_{\varepsilon_2} g(j)$, proving the claim.

Let $S_i = [\ell_i, r_i]$. We also claim that the sequence $(\ell_i)_i$ and $(r_i)_i$ are monotone non-decreasing. Indeed, assume, for a contradiction, that $\ell_i > \ell_{i+1}$. Then $i < i+1 < \ell_{i+1}(1+\varepsilon_1)$ and $i > \ell_i/(1+\varepsilon_1) > \ell_{i+1}/(1+\varepsilon_1)$, so $i \simeq_{\varepsilon_1} \ell_{i+1}$; moreover, $g(\ell_{i+1}) \le f(i+1)(1+\varepsilon_2) \le f(i)(1+\varepsilon_2)$, and $g(\ell_{i+1})(1+\varepsilon_2) \ge g(\ell_i)(1+\varepsilon_2) \ge f(i)$, so $u_i \simeq_{(\varepsilon_1,\varepsilon_2)} v_{\ell_{i+1}}$, a contradiction. The proof of the monotonicity of (r_i) is similar.

Moreover, the collection of intervals $(S_i)_i$ covers $[1, n]$ because every point of g is $(\varepsilon_1, \varepsilon_2)$-close to some point of f.

The coupling then simply consists of the pairs

$$\{(i, j) : \max(\ell_i, r_{i-1}) \le j \le r_i\},$$

in lexicographic order, i.e. the red edges of figure 3. Let us verify that this is a correct coupling sequence. Since $r_{i-1} \le r_i$, every i belongs to at least one pair. Every j will appear in the pair (i, j) where i is minimum such that $r_i \ge j$. Such an i exists because every j belongs to at least one S_i. From one element of the sequence to the next, we either keep i unchanged and move from one element of S_i to the next element of S_i, incrementing the count by 1 on g; or we switch from S_i to S_{i+1}, incrementing i and possibly incrementing j by one as well, in the case in which $r_i \notin S_{i+1}$. Thus this forms a correct coupling sequence such that $f \sim_{(\varepsilon_1,\varepsilon_2)} g$.

◀

Proof. (Proof of Theorem 13)

By contraposition of Lemma 26, and up to symmetry between f and g, there exists a point $u = (i, f(i))$ of f such that no point of g is $(3\varepsilon_1, 3\varepsilon_2)$-close to it. All the points of g are outside the rectangle $R_d = [i/(1+3\varepsilon_1), i.(1+3\varepsilon_1)] * [f(i)/(1+3\varepsilon_2), f(i).(1+3\varepsilon_2)]$ which includes the rectangle R_s defined below.

Since f is $(3\varepsilon_1, \varepsilon_2)$-step-compatible, there exist x, y such that:

$$x \leq i \leq x.(1+3\varepsilon_1)$$
$$y \leq f(i) \leq y.(1+\varepsilon_2)$$

and the points of f whose x coordinate is in the interval $[x, x.(1+3\varepsilon_1)]$ have a y coordonate in the interval $[y, y.(1+\varepsilon_2)]$. The points are inside this rectangle $R_s = [x, x.(1+3\varepsilon_1)] * [y, y.(1+\varepsilon_2)]$.

Notice that $R_s \subseteq R_d$. The points $(j, g(j))$ are outside R_d and there are two cases: either the curve g does or does not cross the rectangle R_s. It crosses R_s if there is a point t such that:

$$x \leq t \leq x.(1+3\varepsilon_1)$$
$$g(t) \leq f(i)/(1+3\varepsilon_2)$$
$$f(i).(1+3\varepsilon_2) \leq g(t-1)$$

In the first case, if g does not cross R_s, it is either above or below. Assume it is below, then the rectangle below R_s in R_d, i.e.

$$R = [x, x.(1+3\varepsilon_1)] * [y/(1+3\varepsilon_2), y]$$

is an $(\varepsilon_1, \varepsilon_2)$-rectangle separating f and g. Its relative width is $(1+3\varepsilon_1)$ and its relative height is $(1+3\varepsilon_2)$.

In the second case, if g crosses R_s, then consider the two rectangles R_1 above R_s and R_2 below R_s within the span of R_s on each side of t, as shown in figure 4:

$$R_1 = [x, t) * [y.(1+\varepsilon_2), f(i).(1+3\varepsilon_2)]$$
$$R_2 = [t, x.(1+3\varepsilon_1)] * [y, f(i).(1+3\varepsilon_2)]$$

Their relative height is larger than $(1+\varepsilon_2)$ because $f(i).(1+3\varepsilon_2)/y.(1+\varepsilon_2) > (1+3\varepsilon_2)/(1+\varepsilon_2) > (1+\varepsilon_2)$. At least one of them has a relative width larger than $(1+\varepsilon_1)$ because the product of their relative width is greater then $t/x * x.(1+3\varepsilon_1)/t = (1+3\varepsilon_1)$. At least one of the rectangle has a width greater than $\sqrt{1+3\varepsilon_1} > 1+\varepsilon_1$. ◀

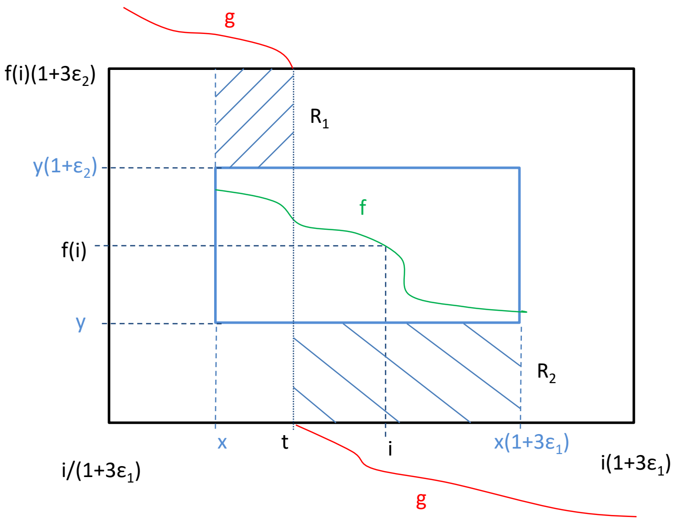

Figure 4 Separating rectangles in theorem 13