The quantum complexity of approximating the frequency moments

Ashley Montanaro*

May 1, 2015

Abstract

The k'th frequency moment of a sequence of integers is defined as $F_k = \sum_j n_j^k$, where n_j is the number of times that j occurs in the sequence. Here we study the quantum complexity of approximately computing the frequency moments in two settings. In the query complexity setting, we wish to minimise the number of queries to the input used to approximate F_k up to relative error ϵ. We give quantum algorithms which outperform the best possible classical algorithms up to quadratically. In the multiple-pass streaming setting, we see the elements of the input one at a time, and seek to minimise the amount of storage space, or passes over the data, used to approximate F_k. We describe quantum algorithms for F_0, F_2 and F_∞ in this model which outperform the best possible classical algorithms almost quadratically.

1 Introduction

Given a sequence of integers $a_1, \dots, a_n$, where $a_i \in [m] := \{1, \dots, m\}$ for each i, let n_j denote the number of elements in the sequence which are equal to the integer j. Then the k'th frequency moment is defined as

$$F_k := \sum_j n_j^k.$$

Thus, for example, F_0 is the number of distinct elements in the sequence, and $F_1 = n$. We also define $F_\infty := \max_j n_j$. We look to approximate F_k up to relative error ϵ with bounded failure probability, or in other words to output $\widetilde{F_k}$ such that

$$\Pr[|\widetilde{F_k} - F_k| > \epsilon F_k] \leq 1/3.$$

As well as the intrinsic mathematical interest of this fundamental problem, it also has many practical uses, with F_0 and F_2 in particular occuring in database applications (see e.g. [2, 19]). It is therefore unsurprising that a vast amount of work, in a variety of different contexts, has been done to characterise the complexity of approximating the frequency moments; we summarise some of this below. In this work we address the complexity of approximating the frequency moments using a quantum computer.

We consider two different models where one could hope to achieve quantum speedups, both of which correspond to well-studied versions of the problem classically:

- The streaming model. In this model, we receive each item a_i one at a time, in sequence. In the single-pass streaming model, we are asked to output an estimate for F_k at the end of the

*Department of Computer Science, University of Bristol, UK; `ashley@cs.bris.ac.uk`.

sequence. In the multiple-pass streaming model, the stream repeats a number of times and we are asked to output an estimate after some number of repetitions. The challenge is that we assume that we only have access to limited storage space, and in particular not enough to store the whole stream.

- The query complexity model. Here we can query arbitrary elements a_i, and seek to approximate F_k using the minimal number of queries.

We assume in both cases that we know the total number of elements n in advance, and allow probability of failure $1/3$. This can be improved to δ, for arbitrary fixed $\delta > 0$, by repetition and taking the median. In each of these models, which we define somewhat more formally below, we obtain quantum improvements over the best possible classical complexities. Several of these are quadratic or near-quadratic.

Our main results can be summarised as follows:

- In the query complexity model, F_0 can be approximated with $O(\sqrt{n}/\epsilon)$ quantum queries, as compared with the classical lower bound of $\Omega(n)$ for $\epsilon = O(1)$ [19].
- In the query complexity model, F_k can be approximated with $\widetilde{O}(n^{(1-1/k)(1-2^{k-2}/(2^k-1))}/\epsilon^2)$ quantum queries[1] for $k \geq 2$, as compared with the classical lower bound of $\Omega(n^{1-1/k}/\epsilon^{1/k})$ [6]. Observe that $(1 - 2^{k-2}/(2^k - 1)) \leq 3/4$ for all $k \geq 2$, so this gives an asymptotic separation for all such k. In the important special case of F_2, the quantum upper bound is $\widetilde{O}(n^{1/3}/\epsilon^2)$, as compared with the classical lower bound $\Omega(n^{1/2}/\epsilon^{1/2})$.
- The dependence on n of these algorithms is essentially optimal for F_0 and F_2, and the quantum query complexity of approximating F_∞ is lower-bounded by $\Omega(n^{2/3})$.
- In the streaming model, F_0 can be approximated by a quantum algorithm which stores $O((\log n) \log 1/\epsilon)$ qubits and makes $O(1/\epsilon)$ passes over the input. Any classical algorithm that makes T passes over the input must store $\Omega(1/(\epsilon^2 T))$ bits [18].
- In the streaming model, F_2 can be approximated by a quantum algorithm which stores $O(\log n + \log(1/\epsilon))$ qubits and makes $\widetilde{O}(1/\epsilon)$ passes over the input. Any classical algorithm that makes T passes over the input must store $\Omega(1/(\epsilon^2 T))$ bits [18].
- In the streaming model, F_∞ can be computed exactly by a bounded-error quantum algorithm which stores $O(\log^2 n)$ qubits and makes $O(\sqrt{n})$ passes over the input. For sufficiently large m, any classical algorithm that makes T passes must store $\Omega(n/T)$ bits [3].
- The above quantum upper bounds in the multiple-pass streaming model are all optimal, up to logarithmic factors.

For simplicity, we assume in these bounds and throughout that m is quite large, $m \geq 2n$. We can also always assume that $m = O(n^2)$ because we can hash all the input elements to a set of size $O(n^2)$ without affecting the frequency moments, with 99% probability; thus $\log m = \Theta(\log n)$.

The near-quadratic separations we obtain in the multiple-pass streaming model are, arguably, the first demonstration of a quantum advantage over classical computation for computing functions of practical interest in this model. Considerably stronger exponential separations have been shown

[1]The $\widetilde{O}$ notation hides polylogarithmic factors.

in the one-pass streaming model by Gavinsky et al. [24] for a partial function, and by Le Gall [34] for a total function. Unfortunately, these functions seem somewhat contrived. (However, it is possible to reinterpret the result of Le Gall as applying to computing the Disjointness function in the multiple-pass streaming model. In this setting the problem becomes more natural but the complexity reduction becomes only quadratic.)

1.1 Prior work

There has been a huge amount of work characterising the classical complexity of approximating the frequency moments in various settings, only a fraction of which we mention here. See, for example, [14, 30] for further references.

In the streaming model:

- (F_0) Flajolet and Martin gave a single-pass streaming algorithm which uses $O(\log m)$ bits of space and computes F_0 up to a constant factor [23]. Alon, Matias and Szegedy improved this by replacing the randomness used in the Flajolet-Martin algorithm with a family of simple hash functions [3]. Bar-Yossef et al. gave several different algorithms for approximating F_0 up to a $(1+\epsilon)$ factor, using as little as $\widetilde{O}(1/\epsilon^2+\log m)$ space [7]. Kane, Nelson and Woodruff have now completed this line of research by giving a single-pass streaming algorithm which approximates F_0 using $O(1/\epsilon^2+\log m)$ space [30]. This is optimal for single-pass streaming algorithms; a space lower bound of $\Omega(\log m)$ was shown by Alon, Matias and Szegedy [3], and a lower bound of $\Omega(1/\epsilon^2)$ was shown by Woodruff [39]. This was improved to an $\Omega(1/(\epsilon^2 T))$ lower bound for T-pass streaming algorithms by Chakrabarti and Regev [18].

- (F_2) Alon, Matias and Szegedy gave an $O((\log m)/\epsilon^2)$ single-pass streaming algorithm [3], and also showed an $\Omega(\log m)$ lower bound. An $\Omega(1/\epsilon^2)$ lower bound for single-pass streaming algorithms was proven by Woodruff [39], which was similarly extended to an $\Omega(1/(\epsilon^2 T))$ lower bound for T-pass streaming algorithms by Chakrabarti and Regev [18].

- (F_k, $k>2$) Alon, Matias and Szegedy gave single-pass streaming algorithms using space $\widetilde{O}(m^{1-1/k})$ [3]. An almost-optimal $\widetilde{O}(m^{1-2/k}/\epsilon^{10+4/k})$ algorithm was later given by Indyk and Woodruff for any $k>2$ [29]. This was simplified by Bhuvanagiri et al., who also improved the dependence on ϵ [12]. Very recently, Braverman et al. gave an $O(m^{1-2/k})$ algorithm for $k>3$ and $\epsilon=\Omega(1)$ [14]. This effectively matches the tightest known general space lower bound on T-pass streaming algorithms, $\Omega(m^{1-2/k}/(\epsilon^{4/k}T))$ shown by Woodruff and Zhang [40].

- (F_∞) Alon, Matias and Szegedy showed an $\Omega(m)$ space lower bound [3], even for multiple-pass streaming algorithms with constant ϵ, by a reduction from the communication complexity of Disjointness.

Near-optimal time-space tradeoffs for the related problem of exactly computing frequency moments over sliding windows were proven by Beame, Clifford and Machmouchi [9].

The classical *query* complexity of approximating the frequency moments has also been studied, under the name of sample complexity. Charikar et al. [19] gave a lower bound of $\Omega(n(1-\epsilon)^2)$ queries for approximating F_0. For any $k\geq 2$, Bar-Yossef [6] showed a lower bound of $\Omega(n^{1-1/k}/\epsilon^{1/k})$ for any $k\geq 2$, and a nearly matching upper bound (for $\epsilon=\Omega(1)$, $k=O(1)$) of $O(n^{1-1/k}/\epsilon^2)$.

In the quantum setting, remarkably little seems to be known about the complexity of approximately computing frequency moments. Coffey and Prezkuta [20] propose a quantum algorithm based on quantum counting which computes F_∞ exactly for a sequence of n elements, each picked from a set of size m, using $O(m\sqrt{n}\log m)$ queries. However, this complexity does not seem to be correct (cf. the lower bound of $\Omega(n)$ for exact computation of F_∞ with $m = 2$ which we prove below).

Kara [31] gave a quantum algorithm for approximating the modal value up to a $1+\epsilon$ factor using $O((m^{3/2}\log m)/\epsilon)$ queries, where m is again the size of the set of values. Note that once the modal value is determined, F_∞ itself can be approximately computed using quantum counting at the cost of an additional $O(\sqrt{n})$ queries. A quantum algorithm for computing F_0 over sliding windows was given in [9], and achieves better time-space tradeoffs than are possible classically.

In terms of lower bounds, it was shown by Buhrman et al. [17] that computing F_0 exactly requires $\Omega(n)$ quantum queries. This result was later sharpened by Beame and Machmouchi [10] to show that even distinguishing between the cases of a function being 2-to-1 and almost 2-to-1 requires $\Omega(n)$ quantum queries.

Very recent independent work of Ambainis et al. [5] has considered a related problem to approximating F_0: testing the image size of a function. The quantum algorithm of [5] is based in the setting of property testing, and has subtly different parameters to the algorithm for F_0 presented here. Given oracle access to a function $f : [n] \to [m]$, their algorithm distinguishes between two cases: a) the image of f is of size at most k; b) an ϵ fraction of the output values of f need to be changed to reduce its image to size at most k. The algorithm uses $O(\sqrt{k/\epsilon}\log k)$ quantum queries.

1.2 Techniques

The new quantum algorithms we obtain are based on combining a number of different, previously known ingredients. Interestingly, ideas from classical streaming algorithms turn out to be useful for developing efficient quantum query algorithms; on the other hand, previously known efficient quantum query algorithms help to develop new quantum streaming algorithms.

The quantum query algorithm for F_0 is rather straightforward and is based around the idea from the streaming setting of Bar-Yossef et al. [7] that it suffices to compute the $O(1/\epsilon^2)$ smallest values of a pairwise independent hash function to estimate F_0. This can be done efficiently using a quantum algorithm of Dürr et al. [21]. The algorithm for F_k, $k \geq 2$, is more involved, and starts from the observation [6] that a good approximation of F_k can be found by counting k-wise collisions in a large enough random subset S of the inputs. On the other hand, if there are not too many k-wise collisions in S, the number of k-wise collisions can be computed efficiently using a quantum algorithm for k-distinctness, the problem of finding k equal elements within S [11]. The algorithm therefore runs the k-distinctness subroutine on random subsets S which are exponentially increasing in size, until it finds a k-wise collision. It then switches to estimating the number of k-wise collisions using the k-distinctness algorithm.

In the quantum streaming model, to approximately compute F_0 we modify a different algorithm of Bar-Yossef et al. [7]. The idea is to use quantum amplitude estimation [13] to approximate the probability that a random hash function $h : [m] \to [R]$, where $R = \Theta(F_0)$, maps any of the elements in the stream to 1. This enables a quadratic improvement, in terms of the scaling with ϵ, over the classical algorithm in [7]. The main technical difficulty is to ensure that checking whether any of the elements in the stream are mapped to 1 can be implemented reversibly and space-efficiently. The efficient quantum algorithm for F_2 applies a quantum subroutine for efficient estimation of the expected value of random variables with bounded variance [35] to an estimator defined by Alon, Matias and Szegedy [3]. Finally, the algorithm for F_∞ implements the quantum algorithm of Dürr and Høyer for finding the maximum [22] in a streaming setting.

The lower bounds in both the query and streaming models are based around the use of reductions. In the case of the query model, we reduce from well-studied problems in query complexity such as the threshold and element distinctness functions. In the case of the streaming model, we reduce from the Gap-Hamming, Disjointness and Equality problems in communication complexity.

2 Quantum query complexity

In this section, we describe quantum query algorithms for approximately computing F_k, followed by lower bounds. We use the standard model of quantum query complexity [16, 26]. The algorithm is given access to the input via a unitary operator O which maps $O|i\rangle|x\rangle \mapsto |i\rangle|x+a_i\rangle$ for $i \in [n]$, $x \in \mathbb{Z}_{m'}$ for some $m' \geq m$, and the goal is to approximately compute F_k with the minimal number of queries to O.

2.1 F_0

Our quantum algorithm for computing F_0 is based on a classical algorithm of Bar-Yossef et al. [7]. The starting point is the following idea of Flajolet and Martin [23] and Alon, Matias and Szegedy [3]: Given a uniformly random function $h : [m] \to [0,1]$, the value $\min_i h(a_i)$ should provide a good approximation of $1/F_0$. Indeed, the expected minimum of F_0 random variables uniformly distributed in $[0,1]$ is precisely $1/(F_0+1)$. To achieve an estimate of F_0 accurate up to relative error ϵ, it turns out to be sufficient to know the $O(1/\epsilon^2)$ smallest distinct values of $h(a_i)$, for a pairwise independent hash function h [7]. We can calculate these efficiently using a quantum algorithm from Dürr et al. [21] for finding the d smallest values of a function f, with the additional constraint that the values have to be of different types.

Theorem 1 (Dürr et al. [21])**.** *Given oracle access to two functions $f, g : [n] \to \mathbb{Z}$ and an integer d, there is a quantum algorithm which uses $O(\sqrt{dn})$ queries to f and g and outputs a set of d' indices I, where $d' = \min\{d, |\{g(j) : j \in [n]\}|\}$, such that:*

- *$g(i) \neq g(j)$ for all $i, j \in I$;*
- *For all $i \in I$ and $j \in [n] \backslash I$, if $f(j) < f(i)$ then $f(i') \leq f(j)$ for some $i' \in I$ with $g(i') = g(j)$.*

The algorithm fails with probability at most δ, for arbitrary $\delta = \Omega(1)$.

The quantum algorithm for approximately computing F_0 is formally described as Algorithm 1.

```
Set $d = \lceil 96/\epsilon^2 \rceil$, $M = m^3$.

  1. Let $h : [m] \to [M]$ be picked at random from a pairwise independent family of hash
     functions.

  2. Use the algorithm of Theorem 1 with $f(i) = g(i) = h(a_i)$ and $\delta = 1/15$ to compute the $d$
     smallest distinct values of $h(a_i)$. Let $v$ be the $d$'th smallest value.

  3. Output $dM/v$.
```

Algorithm 1: Computing F_0

Theorem 2. *Algorithm 1 makes $O(\sqrt{n}/\epsilon)$ queries and outputs an estimate of F_0 accurate up to relative error ϵ with probability at least $3/5 - 1/m$.*

Proof. It is shown in the proof of Theorem 1 in [7] that, for the value of d chosen by Algorithm 1, dM/v approximates F_0 up to a $1+\epsilon$ multiplicative factor with probability at least $2/3 - 1/m$. The claim then follows from the bounds of Theorem 1. □

2.2 F_k for $k > 1$

We begin by describing the technical tools required for the efficient quantum query algorithm for approximating F_k, starting with the underlying quantum subroutine for the so-called k-distinctness problem.

Theorem 3 (Belovs [11])**.** *Fix integer $k \geq 2$ and real δ such that $0 < \delta < 1$. There is a quantum algorithm which, given query access to a sequence $S = s_1, \dots, s_n$, determines whether there exists a set I of k distinct indices such that $s_i = s_j$ for all $i, j \in I$. The output of the algorithm is either such a set I or "no". The algorithm uses $O(n^{1-2^{k-2}/(2^k-1)} \log(1/\delta)) = o(n^{3/4} \log(1/\delta))$ queries to S. It outputs an incorrect answer with probability at most δ.*

This algorithm is normally presented with failure probability $1/3$, but this can be reduced to δ by repetition, noting that we can check a claimed equal k-tuple using k additional queries.

We will also need a technical lemma, which shows that F_k can be expressed in terms of the number of k-wise collisions occurring in a random subset of the input integers. This lemma was essentially previously shown in [6], generalising the proof of [25] for the case $k = 2$. However, as the terminology and parameter choice of these previous works is somewhat different to our usage here, we state and prove it afresh. Let $\binom{[\ell]}{k}$ denote the set of k-subsets of $[\ell]$.

Lemma 4. *Fix ℓ such that $1 \leq \ell \leq n$. Let $s_1, \dots, s_\ell \in [n]$ be picked uniformly at random and define*

$$C_k(s_1, \dots, s_\ell) := |\{T \in \binom{[\ell]}{k} : a_{s_i} = a_{s_j} \text{ for all } i, j \in T\}|.$$

Then

$$\mathbb{E}_{s_1,\dots,s_\ell}[C_k(s_1, \dots, s_\ell)] = \frac{\binom{\ell}{k} F_k}{n^k}$$

and

$$\operatorname{Var}(C_k) \leq \sum_{q=k}^{2k-1} \left(\frac{\ell F_k^{1/k}}{n} \right)^q .$$

Proof. See Appendix A. □

We are now ready to describe the algorithm for computing F_k, as Algorithm 2 below. Informally, the algorithm first uses the k-distinctness subroutine to determine a size ℓ such that a random subset of size ℓ contains some, but not too many, k-wise collisions. This is already enough to compute F_k up to constant multiplicative accuracy. The algorithm then switches to estimating the expected number of k-wise collisions in a random subset of size ℓ, and uses this to approximate F_k more precisely.

```
1. Set $\ell = n$.
2. For $i = 0, \dots, \lceil \log_2 n \rceil$:
    (a) Pick $s_1, \dots, s_{2^i} \in [n]$ uniformly at random and let $S$ be the sequence $a_{s_1}, \dots, a_{s_{2^i}}$.
    (b) Apply a $k$-distinctness algorithm to $S$ with failure probability $1/(8 \log_2 n)$.
    (c) If it returns a set of $k$ equal elements, set $\ell = 2^i$ and terminate the loop.
3. Set $M = \lceil K/\epsilon^2 \rceil$ for some universal constant $K$ to be determined. For $k = 1, \dots, M$:
    (a) Pick $s_1, \dots, s_\ell \in [n]$ uniformly at random and let $S$ be the sequence $a_{s_1}, \dots, a_{s_\ell}$.
    (b) Let $T$ be an empty sequence.
    (c) Repeat the following subroutine forever:
        i. Apply a $k$-distinctness algorithm to $S$ with failure probability $\epsilon^2/(8K\ell)$.
        ii. If it returns that all elements are distinct, terminate.
        iii. If it returns a $k$-tuple of indices $I = i_1, \dots, i_k$ such that $a_p = a_q$ for all $p, q \in I$,
             append $i_1, \dots, i_k$ to $T$. Remove one element of the $k$-tuple from $S$.
    (d) Let the sequence $B = b_1, \dots, b_\ell$ be defined by concatenating $T$ with the sequence
        containing one copy of each element occurring in $T$, followed by a sequence of at
        most $\ell$ arbitrary elements not in $T$.
    (e) Set $C^{(k)} := |\{U \in \binom{[\ell]}{k} : b_i = b_j \text{ for all } i, j \in U\}|$.
4. Output $\frac{n^k}{M\binom{\ell}{k}} \sum_{k=1}^{M} C^{(k)}$.
```

Algorithm 2: Computing F_k

Theorem 5. *Algorithm 2 outputs an approximation of F_k which is accurate up to relative error $1+\epsilon$ with probability at least $3/4$, using an expected number of queries which is*

$$O((n^{(1-1/k)(1-2^{k-2}/(2^k-1))}/\epsilon^2)\log(n/\epsilon)).$$

Proof. First note that, by a union bound, we can assume that all the uses of the k-distinctness algorithms succeed, except with total error probability $1/4$. We now show that it is likely that ℓ is chosen such that $An/F_k^{1/k} \le \ell \le Bn/F_k^{1/k}$, for some A and B relatively close to 1. By Markov's inequality and Lemma 4, for any $\ell \le An/F_k^{1/k}$

$$\Pr_{s_1,\dots,s_\ell}[C_k(s_1,\dots,s_\ell) \ge 1] \le \mathbb{E}_{s_1,\dots,s_\ell}[C_k(s_1,\dots,s_\ell)] = \frac{\binom{\ell}{k} F_k}{n^k} \le \frac{F_k}{n^k}\left(\frac{\ell e}{k}\right)^k.$$

Therefore, the probability that ℓ is set to be lower than $An/F_k^{1/k}$ after the first loop is at most

$$F_k \left(\frac{e}{kn}\right)^k \sum_{i=0}^{\log_2(An/F_k^{1/k})} 2^{ik} \le 2F_k \left(\frac{e}{kn}\right)^k \left(\frac{An}{F_k^{1/k}}\right)^k = 2\left(\frac{Ae}{k}\right)^k.$$

On the other hand, let $\ell = Dn/F_k^{1/k}$, for some D such that $D \ge B \ge 1$ and $\ell \ge 2$. Then from

Lemma 4,

$$\mathrm{Var}(C_k) \leq \sum_{q=k}^{2k-1} \left(\frac{\ell F_k^{1/k}}{n}\right)^q = \sum_{q=k}^{2k-1} D^q \leq k\, D^{2k-1}. \tag{1}$$

So, via Chebyshev's inequality (aka the second moment method),

$$\Pr_{s_1,\dots,s_\ell}[C_k(s_1,\dots,s_\ell)=0] \leq \frac{\mathrm{Var}(C_k)}{\mathbb{E}[C_k]^2} \leq k\,\frac{D^{2k-1}n^{2k}}{\binom{\ell}{k}^2 F_k^2} \leq k\,\frac{k^{2k}D^{2k-1}n^{2k}}{\ell^{2k}F_k^2} = \frac{k^{2k+1}}{D} \leq \frac{k^{2k+1}}{B}.$$

Fixing, for example, $A = (k/e)20^{-1/k}$, $B = 10k2^{2k+1}$, we have that $An/F_k^{1/k} \leq \ell \leq Bn/F_k^{1/k}$ except with probability at most $1/5$.

Assuming that ℓ is indeed bounded in this way, we now show that it suffices to repeat the second subroutine $O(1/\epsilon^2)$ times to estimate F_k up to relative error $1+\epsilon$. By Lemma 4 we have $\mathrm{Var}(C_k) \leq kB^{2k-1}$. Let $\overline{C} = \frac{1}{M}\sum_{k=1}^M C^{(k)}$. Then $\mathrm{Var}(\overline{C}) \leq kB^{2k-1}/M$. By Chebyshev's inequality,

$$\Pr\left[\left|\overline{C} - \frac{\binom{\ell}{k}F_k}{n^k}\right| \geq \epsilon\frac{\binom{\ell}{k}F_k}{n^k}\right] \leq \frac{k^2B^{4k-2}n^{2k}}{M\epsilon^2\binom{\ell}{k}^2F_k^2} \leq \frac{k^{2k+1}B^{4k-2}n^{2k}}{M\epsilon^2\ell^{2k}F_k^2} \leq \frac{k^{2k+1}B^{4k-2}}{A^{2k}M\epsilon^2}.$$

This implies that, in order to estimate $\mathbb{E}[C^{(k)}]$ up to relative error $1+\epsilon$ with failure probability at most $1/5$, say, it suffices to take $M = \lceil 5k^{2k+1}B^{4k-2}/(A^{2k}\epsilon^2)\rceil$.

We finally compute the expected number of queries used by the algorithm. Assume that $\ell = O(n/F_k^{1/k}) = O(n^{1-1/k})$ and for conciseness write $\alpha = 1 - 2^{k-2}/(2^k-1)$. The first loop makes

$$\sum_{i=0}^{\log_2 \ell} O(2^{\alpha i}\log\log n) = O(\ell^\alpha \log\log n) = O(n^{\alpha(1-1/k)}\log\log n)$$

queries. In the second loop, the expected number of repetitions of the subroutine is upper-bounded by the expected value of $C_k(s_1,\dots,s_\ell)$, because the number of elements such that there are at least $k-1$ other elements with the same value is a lower bound on the number of k-wise collisions. For $\ell \leq Bn/F_k^{1/k}$, this expected value is $O(1)$. Therefore, the expected runtime of the subroutine is $O((n/F_k^{1/k})^\alpha \log(\ell/\epsilon^2)) = O(n^{\alpha(1-1/k)}\log(n/\epsilon))$. As there are $O(1/\epsilon^2)$ uses of the subroutine, the overall expected runtime is $O((n^{\alpha(1-1/k)}/\epsilon^2)\log(n/\epsilon))$ as claimed. □

We remark that it might be possible to improve the dependence on ϵ of this algorithm to $\widetilde{O}(1/\epsilon)$ by replacing step 3 with the use of a quantum algorithm for approximately computing the mean given a bound on the variance [35], as in Section 3.2 below. The reason that this does not seem immediate is that we only know an upper bound on the expected runtime of step 3, rather than a worst-case bound as required by this quantum algorithm.

2.3 F_∞

We observe that F_∞ is closely connected to the much-studied (and confusingly named) k-distinctness problem: determining whether a sequence S of integers contains k equal integers [4, 11]. Approximating F_∞ up to relative error less than $1/(3k)$ allows one to solve k-distinctness. The case $k=2$ (element distinctness) has a lower bound of $\Omega(n^{2/3})$ [4], implying that the same lower bound holds for computing F_∞ up to relative error $O(1)$. No stronger lower bound is known for higher k.

On the other hand, if we could solve k-distinctness for all k, we could compute F_∞ exactly using binary search. One can show by straightforward techniques (see below) that k-distinctness requires $\Omega(n)$ queries for $k = \Omega(n)$. However, to approximate F_∞ it is not necessary to solve k-distinctness exactly, but merely to solve a gapped version of k-distinctness. That is, we are given parameters k, ϵ and asked to distinguish between the following two cases:

1. S contains k equal elements;
2. S contains no sequence of $(1-\epsilon)k$ equal elements or more.

If we can approximate F_∞ up to relative error ϵ, we can clearly solve gapped k-distinctness. In addition, if we can solve gapped k-distinctness for arbitrary k, we can approximate F_∞ using binary search, at a cost of an $O(\log n)$ factor in the number of queries used.

2.4 Lower bounds

We can obtain a number of easy lower bounds on the query complexity of estimating the frequency moments via reductions from previously studied problems. We will use the following result of Nayak and Wu [37]:

Theorem 6 (Nayak and Wu [37]). *Let $f : [n] \rightarrow \{0,1\}$ either satisfy $|\{x : f(x) = 1\} = n/2$, or $|\{x : f(x) = 1\}| = (1+\epsilon)n/2$. Then any quantum algorithm that determines which is the case must make $\Omega(1/\epsilon)$ queries to f.*

Theorem 7. *The quantum query complexity of estimating F_0 up to relative error ϵ, with failure probability at most $1/3$, is at least $\Omega(\sqrt{n/\epsilon})$. For any $k > 1$, the quantum query complexity of estimating F_k up to relative error ϵ, with failure probability at most $1/3$, is at least $\Omega(n^{1/2-1/(2k)}/\epsilon)$ and also $\Omega(n^{1/3})$ for any $\epsilon < 1/4$. For $k = \infty$, the quantum query complexity is at least $\Omega(n^{2/3} + 1/\epsilon)$.*

Proof. We first deal with the F_0 lower bound, by a reduction from the threshold function Th_d on n bits, for $d \le n/2$. This function is defined by $\mathrm{Th}_d(x) = 1$ if $|x| \ge d$, and $\mathrm{Th}_d(x) = 0$ otherwise. Given an input $x \in \{0,1\}^n$, define a sequence of n integers a_i by $a_i = i$ if $x_i = 1$, and $a_i = 0$ otherwise. Then $\mathrm{Th}_d(x) = 1$ if and only if $F_0 \ge d$. To determine this, it suffices to approximate F_0 up to relative error $1/(4d)$. As the threshold function has a lower bound of $\Omega(\sqrt{dn})$ for $d \le n/2$ [8], setting $d = \lceil 1/\epsilon \rceil$ implies the claimed result.

For the first bound for $k > 1$, we reduce quantum counting to estimating F_k. Consider the problem of distinguishing two n-bit strings, one of which has Hamming weight $\ell \le n/2$, the other of which has Hamming weight $\ell + \Delta$. This requires $\Omega(\sqrt{n/\Delta} + \sqrt{\ell n}/\Delta)$ quantum queries [37]. For any bit-string $x \in \{0,1\}^n$, define a sequence of n integers a_i such that $a_i = i$ if $x_i = 0$, and $a_i = 0$ if $x_i = 1$. Consider two strings x, y such that $|x| = n^{1/k}$, and $|y| = (n(1+8\epsilon))^{1/k}$; so $\ell = n^{1/k}$, $\Delta = n^{1/k}((1+8\epsilon)^{1/k} - 1)$. Then F_k is equal to $2n - n^{1/k}$ for the first corresponding sequence, and $2n + 8\epsilon n - (n(1+8\epsilon))^{1/k}$ for the second sequence. One can verify that approximating F_k up to relative error ϵ allows these two cases to be distinguished.

For the $\Omega(n^{1/3})$ bound for $k > 1$, we use a reduction from the collision problem, which has a lower bound of $\Omega(n^{1/3})$ [1]. Let S_1 be a sequence of n numbers where each number occurs once, and let S_2 be a sequence of n numbers where each number occurs twice. Then $F_k(S_1) = n$, $F_k(S_2) = n2^{k-1}$. So estimating F_k up to multiplicative error $\epsilon < 1/4$ allows these two cases to be distinguished for any $k > 1$.

The $\Omega(n^{2/3})$ bound for $k = \infty$ follows from the lower bound on the quantum query complexity of element distinctness [1], which is clearly no easier. The $\Omega(1/\epsilon)$ bound follows from Theorem 6, considering sequences containing only 0's and 1's. □

We finally consider the case of computing F_k with very high accuracy.

Theorem 8. *For any $k \neq 1$, the quantum query complexity of computing F_k up to $O(1/n)$ relative error, with probability of failure at most $1/3$, is $\Theta(n)$.*

Proof. First we take $k \neq \infty$. Beame and Machmouchi [10] showed that the complexity of the problem of distinguishing between the following two cases is $\Theta(n)$: a sequence of n integers, divided into equal pairs; and a sequence of n integers, which are equal pairs apart from two distinct integers. For $k > 1$, $F_k(S_1) = 2^{k-1}n$ and $F_k(S_2) = 2^{k-1}n + 2$. (For $k = 0$, we have $n/2$ vs. $n/2 - 1$). So computing F_k up to $O(1/n)$ relative error for any $k \neq 1$ allows us to distinguish these two cases.

For $k = \infty$, observe that for any $\epsilon < 1$, computing F_∞ up to error $\epsilon/2$ can be used to distiguish between the two cases in Theorem 6, so the problem has quantum query complexity $\Omega(1/\epsilon)$. □

3 Quantum streaming complexity

We now move on to studying the quantum complexity of computing F_k in the T-pass streaming setting. The model here is defined as follows:

1. The quantum algorithm stores a quantum state $|\psi\rangle$ of S qubits, initialised to some starting state which does not depend on the input.
2. Integers in the input stream are received one-by-one until the end of the stream; as each integer a arrives, a corresponding operation U_a is performed on $|\psi\rangle$.
3. Step 2 is repeated T times.
4. At the end, a measurement is made on $|\psi\rangle$ which is supposed to reveal F_k.

We remark that, in the case $T = 1$, this model is very similar to a one-way quantum finite automaton [36], and also to variants of the streaming model studied by Gavinsky et al. [24] and Le Gall [34]. We could have (essentially equivalently) also defined this process by splitting the stored qubits into two registers, and performing the following operations for each arriving element a:

1. Apply U_a to the first register, where $U_a|x\rangle = |x + a\rangle$.
2. Apply some fixed unitary operation V to both registers.
3. Apply U_a^{-1} to the first register.

We assume that the algorithm knows the number n of elements in the stream in advance.

3.1 F_0

In order to obtain an efficient quantum algorithm for estimating F_0 in the multiple-pass streaming model, we will modify another efficient classical algorithm of Bar-Yossef et al. [7]. The basic idea is as follows. If $h : [m] \to [R]$ is a random function picked from a t-wise independent family of hash functions, for some $t = O(\log 1/\epsilon)$ and some $R = \Theta(F_0)$, then the probability (over the random choice of h) that h maps any of the elements in the stream to 1 provides a good estimate for F_0. Here, rather than sampling random functions h to estimate this quantity, we will use amplitude estimation [13]. The main claim that we need to check is that the operation of checking whether h maps any of the elements in the stream to 1 can be performed reversibly in a *space-efficient* fashion, with only a few passes over the stream. Note that standard reversible-computation techniques do not seem to immediately imply this.

Lemma 9. *Let $\mathcal{H}$, $|\mathcal{H}| = M$, be a family of functions $h_j : [m] \to [R]$ such that the map $H : (j, x) \mapsto h_j(x)$ can be implemented in time T and space S. Then there is a quantum algorithm which estimates $\Pr_j[\exists i, h_j(a_i) = 1]$ up to additive error ϵ using space $S + O(\log n + \log 1/\epsilon)$, $O(1/\epsilon)$ passes over the input, and time $O(T)$ per pass.*

Proof. Set $p = \Pr_j[\exists i, h_j(a_i) = 1]$. We will apply quantum amplitude estimation to estimate p. To do so, we need to coherently implement the function $f(j) = [\exists i, h_j(a_i) = 1]$. We will show that this can be implemented with two passes over the stream and space $S + O(\log n)$. For each element a, the map

$$U_a : |j\rangle|y\rangle \mapsto |j\rangle|y + [h_j(a) = 1]\rangle$$

can be implemented with one use of H, which uses time T and space S. After the whole stream has been read in, performing this map for each element a_i, we have effectively implemented the map

$$|j\rangle|0\rangle \mapsto |j\rangle||\{i : h_j(a_i) = 1\}|\rangle.$$

We now use an ancilla register to store whether the second register is nonzero:

$$|j\rangle|0\rangle|z\rangle \mapsto |j\rangle||\{i : h_j(a_i) = 1\}|\rangle|z + [\exists i, h_j(a_i) = 1]\rangle.$$

It remains to uncompute the contents of the second register. We can do this by reading the stream in again and performing the map

$$U_a^{-1} : |j\rangle|y\rangle \mapsto |j\rangle|y - [h_j(a) = 1]\rangle;$$

this requires only one extra qubit to remember whether we are adding or subtracting. The overall result is that we have implemented the map

$$|j\rangle|z\rangle \mapsto |j\rangle|z + [\exists i, h_j(a_i) = 1]\rangle$$

as required, with two passes over the stream and $S + O(\log n)$ space. Amplitude estimation requires $O(1/\epsilon)$ uses of this map and its inverse, and $O(\log 1/\epsilon)$ qubits of additional space, to estimate $\Pr_j[\exists i, h_j(a_i) = 1]$ up to additive error ϵ [13]. □

We now apply Lemma 9 to the framework of Bar-Yossef et al. [7]. Let $\mathcal{R}$ be the set of all functions $h : [m] \to [R]$, and set $r = \Pr_{h \in \mathcal{R}}[\exists i, h(a_i) = 1]$. The following lemma says that, if we can approximate r, we can approximate F_0.

Lemma 10 (Corollary of Bar-Yossef et al. [7]). *Fix $\epsilon \leq 1$ and assume that R satisfies $2F_0 \leq R \leq 50F_0$. Assume that $\widetilde{r}$ satisfies $|\widetilde{r} - r| \leq \epsilon/150$ and define*

$$\widetilde{F_0} = \frac{\ln(1-\widetilde{r})}{\ln(1-1/R)}.$$

Then $|\widetilde{F_0} - F_0| \leq \epsilon F_0$.

In addition, if we replace $\mathcal{R}$ with a t-wise independent family of hash functions for large enough t, the probability (over the random choice of hash function h) that there exists an i such that $h(a_i) = 1$ is not substantially affected.

Lemma 11 (Corollary of Bar-Yossef et al. [7]). *Let $\mathcal{H}$ be a t-wise independent family of hash functions $h_j : [m] \rightarrow [R]$, where $t = \lceil \ln(300/\epsilon)/\ln 5 \rceil$. Set $p = \Pr_j[\exists i, h_j(a_i) = 1]$. Then $|p - r| \leq \epsilon/300$.*

We now have all the ingredients we need for the F_0 estimation algorithm, which is formally described as Algorithm 3.

Set $t = \lceil \ln(300/\epsilon)/\ln 5 \rceil$.

1. Use the algorithm of [3] to obtain an estimate R such that $2F_0 \leq R \leq 50F_0$ using one pass over the stream and space $O(\log m)$, with probability at least $3/5$.
2. Let $\mathcal{H}$ be a family of t-wise independent hash functions $h_j : [m] \rightarrow [R]$.
3. Using the algorithm of Lemma 9, estimate $p = \Pr_j[\exists i, h_j(a_i) = 1]$ up to additive error $\epsilon/300$. Call the estimate $\widetilde{p}$.
4. Output $\ln(1-\widetilde{p})/\ln(1-1/R)$.

Algorithm 3: Streaming estimation of F_0, based on [7]

Theorem 12. *Algorithm 3 computes F_0 up to relative error ϵ, with failure probability at most $1/3$, using space $O(\log m \log(1/\epsilon))$ and $O(1/\epsilon)$ passes over the input.*

Proof. The claim follows from Lemmas 9, 10, and 11. In somewhat more detail: by Lemma 11, after step 3 of the algorithm, assuming that step 1 has succeeded, $|\widetilde{p} - p| \leq \epsilon/300$ and $|p - r| \leq \epsilon/300$, so $|\widetilde{p} - r| \leq \epsilon/150$. By Lemma 10, the output of the algorithm differs from F_0 by relative error ϵ. Lemma 9 states that approximating p to the required level of accuracy can be done using $O(1/\epsilon)$ passes over the input. The space usage is dominated by the space required to describe the hash functions, which is $O(t \log m) = O(\log m \log(1/\epsilon))$ qubits. □

3.2 F_2 and F_k, $k > 2$

To compute F_2 in the streaming model, we will apply the following result to ideas from the F_2 approximation algorithm of Alon, Matias and Szegedy [3]:

Theorem 13 (Quantum approximation with a bound on the relative variance [35]). *Let $v(\mathcal{A})$ be the distribution on the outputs of a quantum algorithm $\mathcal{A}$ such that $\mathbb{E}[v(\mathcal{A})^2]/\mathbb{E}[v(\mathcal{A})]^2 \leq B$, for some $B \geq 1$, and $\mathcal{A}$ uses S qubits of space. Then there is a quantum algorithm which approximates μ up to additive error $\epsilon\,\mathbb{E}[v(\mathcal{A})]$, with probability at least $2/3$, and uses $\mathcal{A}$ and $\mathcal{A}^{-1}$ $O((B/\epsilon)\log^{3/2}(B/\epsilon)\log\log(B/\epsilon))$ times. The algorithm uses $O(S + \log(B/\epsilon))$ qubits of space.*

The algorithm of [3] uses a set of $M = O(m^2)$ 4-wise independent hash functions $h_i : [m] \to \{\pm 1\}$, and approximately computes the expected value of the function $f(i) = \left(\sum_{j=1}^m h_i(j) n_j\right)^2$ over the random choice of hash function h_i. This is sufficient to approximate F_2:

Claim 14 (Alon, Matias and Szegedy [3])**.** *If $i \in [M]$ is picked uniformly at random, $\mathbb{E}_i[f(i)] = F_2$ and $\mathrm{Var}(f) \le 2F_2^2$.*

Here we would like to apply the algorithm of Theorem 13 to accelerate this procedure. To do so, we need to compute f reversibly and space-efficiently. For each hash function h, we can compute

$$\sum_{i=1}^n h(a_i) = \sum_{j=1}^m h(j) n_j$$

using one pass over the stream. Further, we can compute $f(i)$ reversibly for any i using two passes and space $O(\log m + \log n) = O(\log n)$. We first perform the map

$$|i\rangle|x\rangle|y\rangle \mapsto |i\rangle|x + \sum_{j=1}^n h_i(a_j)\rangle|y + f(i)\rangle,$$

using one pass over the stream. We then use a second pass over the stream to subtract $h_i(a_j)$ for each j, so the state of the second register is effectively unchanged and we have performed the map $|i\rangle|y\rangle \mapsto |i\rangle|y + f(i)\rangle$. To carry out the inverse operation, we do the same thing in reverse.

We can therefore apply Theorem 13 to f, and obtain the following result:

Theorem 15. *Algorithm 4 computes F_2 up to relative error ϵ, with failure probability at most $1/3$, using space $O(\log n + \log(1/\epsilon))$ and*

$$O((1/\epsilon)\log^{3/2}(1/\epsilon)\log\log(1/\epsilon)) = \widetilde{O}(1/\epsilon)$$

passes over the input.

1. Let $\mathcal{H}$ be a family of $O(m^2)$ 4-wise independent hash functions $h_j : [m] \to \{\pm 1\}$.
2. Apply the algorithm of Theorem 13 with accuracy bound ϵ to the following subroutine:
   - (a) Pick $h \in \mathcal{H}$ uniformly at random.
   - (b) Output $(\sum_{i=1}^n h(a_i))^2$.

Algorithm 4: Streaming estimation of F_2, based on [3]

In the case of other moments F_k, for fixed $k > 2$, we can do something similar (but less efficient), using a different estimator described by Alon, Matias and Szegedy [3]. For $i \in [n]$, let $N(i) = |\{j : j \ge i, a_i = a_j\}|$, and let $N_k(i) = n(N(i)^k - (N(i)-1)^k)$. Then the following lemma holds:

Lemma 16 (Alon, Matias and Szegedy [3])**.** *If $i \in [n]$ is picked uniformly at random, then*

$$\mathbb{E}_i[N_k(i)] = F_k, \quad \mathrm{Var}(N_k) \le k m^{1-1/k} F_k^2.$$

It is clear that $N_k(i)$ can be computed reversibly using two passes over the stream (one to compute $N(i)$ and one to uncompute it), using space $O(\log n)$. Using Theorem 13, we can approximate F_k up to additive error ϵF_k using

$$O((m^{1-1/k}/\epsilon)\log^{3/2}(m^{1-1/k}/\epsilon)\log\log(m^{1-1/k}/\epsilon)) = \widetilde{O}(m^{1-1/k}/\epsilon)$$

passes and space $O(\log n + \log(m^{1-1/k}/\epsilon))$. This is sometimes superior to the best classical algorithms [14], but only for very small $\epsilon \leq m^{-1/k}$.

3.3 F_∞

The quantum streaming algorithm for computing F_∞ is straightforward, based on a streaming implementation of the maximum-finding algorithm of Dürr and Høyer [22]. Using one pass over the stream, we can implement the map

$$|j\rangle|x\rangle \mapsto |j\rangle|x \pm n_j\rangle$$

for any j simply by adding (or subtracting) 1 to x each time we see an element with value j. We can use this as an oracle within the quantum algorithm of Dürr and Høyer, which outputs the maximum of N integers, using quantum space $O(\log^2 N)$ and $O(\sqrt{N})$ queries [22]. This immediately gives the following result:

Theorem 17. *There is a quantum algorithm which computes F_∞ exactly, with failure probability at most $1/3$, using space $O(\log^2 m)$ and $O(\sqrt{n})$ passes over the input.*

3.4 Lower bounds

Just as in the classical world, lower bounds on quantum communication complexity (see e.g. [15] for an introduction) can be used to lower-bound the quantum complexity of computing functions in the streaming model. Alice and Bob divide the input $a_1, \dots, a_n, b_1, \dots, b_n$ into two; Alice gets the first half $a_1, \dots, a_n$, Bob the second half $b_1, \dots, b_n$. If there is a streaming algorithm which computes F_k using T passes over the input and stores S qubits, by simulating this algorithm Alice and Bob obtain a two-way quantum communication protocol which communicates $O(TS)$ qubits in total, has no prior shared randomness or entanglement, and computes F_k. If there is a single-pass streaming algorithm, this gives a one-way quantum communication protocol.

We first observe that a bound of Alon, Matias and Szegedy [3] extends to give a general $\Omega(\log n)$ space lower bound in the quantum setting, and an $\Omega(\sqrt{n})$ bound for $k = \infty$. Recall that we assume that $m \geq 2n$.

Theorem 18. *Assume there exists a protocol in the multi-pass quantum streaming model which stores S qubits and uses T passes to compute F_k for a stream of n elements up to relative error $1/8$, with failure probability $1/3$. Then:*

- *if $k \neq 1$, $TS = \Omega(\log n)$;*
- *if $k = \infty$, $TS = \Omega(\sqrt{n})$.*

Proof. In the proof we use $\circ$ to denote the concatenation operation on integer sequences.

- ($k \neq 1$): Alice and Bob will embed the equality function on $\Theta(n)$ bits in their inputs. Choose a family of $2^{\Omega(n)}$ subsets of $[n]$ of size $n/4$ such that each pair of subsets has at most $n/8$ elements in common. Then, if Alice receives input x, Bob receives input y, they encode these as subsets S_x, S_y. If their strings are equal, $F_k(S_x \circ S_y) = n2^{k-2}$; otherwise, $F_0(S_x \circ S_y) \geq 3n/8$ and $F_k(S_x \circ S_y) \leq n/4 + n2^{k-3}$. For any $k \neq 1$, there is at least a constant factor gap between the values of F_k in these two different cases. In particular, approximating F_k up to relative error $1/8$ allows equality of x and y to be tested. This has an $\Omega(\log n)$ quantum communication complexity lower bound [33].

- ($k = \infty$): Alon, Matias and Szegedy [3] give a reduction from Disjointness, which we repeat here. Given sets S_a, S_b, Alice and Bob simply apply the streaming algorithm to the concatenation $S_a \circ S_b$. If there are any elements in common, $F_\infty(S_a \circ S_b) \geq 2$; otherwise, $F_\infty(S_a \circ S_b) = 1$. So computing F_∞ up to relative error ϵ, for any $\epsilon < 1/3$, allows Alice and Bob to determine whether their sets are disjoint. This has a quantum communication complexity lower bound of $\Omega(\sqrt{n})$ [38].

□

There is also a straightforward bound on the complexity of exact computation.

Theorem 19. *Assume there exists a protocol in the multi-pass quantum streaming model which stores S qubits and uses T passes to compute F_k exactly for a stream of n elements, with failure probability $1/3$. Then, if $k \notin \{0, 1, \infty\}$, $TS = \Omega(n)$.*

Proof. We reduce from the problem of computing the Hamming distance between two bit-strings $x, y \in \{0,1\}^n$, which has a quantum communication complexity lower bound of $\Omega(n)$ [27]. Given x, Alice produces an n-element sequence by setting the i'th element to be $i + nx_i$; Bob does something similar with y. Then for any $k \notin \{0, 1, \infty\}$, F_k of the concatenated sequence is precisely $n2^k - (2^k - 1)d(x,y)$, so determining F_k exactly enables the Hamming distance between x and y to be computed. □

We now prove a general quantum lower bound on approximating F_k in the streaming model, based on a sequence of reductions, all following from previously known results. The key point is that good approximations to F_k are known to give efficient protocols, in the setting of two-way communication complexity, for a problem known as Gap-Hamming-Distance [28, 39, 18]. Let $\mathrm{GHD}_{n,t}$ be the partial function defined on a subset of $\{0,1\}^n \times \{0,1\}^n$ as follows:

$$\mathrm{GHD}_{n,t}(x,y) = \begin{cases} 0 & \text{if } d(x,y) \leq t - \sqrt{n} \\ 1 & \text{if } d(x,y) \geq t + \sqrt{n} \end{cases}$$

Then we have the following sequence of claims:

Claim 20 (from Indyk and Woodruff [28], Woodruff [39])**.** *Fix $k \neq \{1, \infty\}$ and $\epsilon \geq 1/\sqrt{n}$. Assume there is a protocol in the multi-pass quantum (resp. classical) streaming model which stores S qubits and uses T passes to approximate F_k up to relative error ϵ, with failure probability p, for a stream of n elements. Then there is $\ell = \Theta(1/\epsilon^2)$ such that there is a quantum (resp. classical) protocol in the 2-way communication model for $GHD_{\ell,\ell/2}$, which has failure probability p and uses $O(TS)$ qubits of communication.*

Claim 21 (Chakrabarti and Regev [18]). *Fix $\alpha \in (0, 1/2]$. Then, if there is a quantum (resp. classical) communication protocol for $GHD_{n,n/2}$ using c qubits (resp. bits) of communication with failure probability p, there is a quantum (resp. classical) communication protocol for $GHD_{n,\alpha n}$ using $O(c)$ qubits (resp. bits) of communication with failure probability p.*

Claim 22 (Razborov [38], see [32] for version here). *Fix a Q-qubit quantum communication protocol on n-bit inputs x, y, with acceptance probabilities $P(x, y)$. Write $P(i) = \mathbb{E}_{|x|=|y|=n/4,|x\wedge y|=i}[P(x, y)]$. Then, for every $d \le n/4$, there exists a degree-d polynomial q such that $|P(i) - q(i)| \le 2^{-d/4+2Q}$ for all $i \in \{0, \dots, n/4\}$.*

Claim 23 (Nayak and Wu [37]). *Let $q : \{0, \dots, n\} \to [-1/3, 4/3]$ be a degree-d polynomial such that $q(a) \le 1/3$, $q(b) \ge 2/3$ for some $a, b \in \{0, \dots, n\}$. Let $c \in \{a, b\}$ be such that $|n/2 - c|$ is maximised, and let $\Delta = |a - b|$. Then $d = \Omega(\sqrt{n/\Delta} + \sqrt{c(n - c)}/\Delta)$.*

Theorem 24. *Assume there exists a protocol in the multi-pass quantum streaming model which stores S qubits and uses T passes to compute F_k up to relative error ϵ for a stream of n elements, with failure probability $1/4$, for $k \neq 1$. Then $TS = \Omega(1/\epsilon)$.*

Proof. The theorem follows from Claims 20 to 23. Assume there exists a protocol which stores S qubits and uses T passes to approximate F_k up to relative error ϵ, with success probability $3/4$. Then by Claim 20 there is a protocol for $\mathrm{GHD}_{\ell,\ell/2}$, where $\ell = \Theta(1/\epsilon^2)$, with the same success probability, using $O(TS)$ qubits of communication. By Claim 21, there is similarly a protocol for $\mathrm{GHD}_{\ell,\ell/8}$ using $Q = O(TS)$ qubits of communication. Now consider the special case of this problem where the inputs x, y are restricted to Hamming weight $\ell/4$. Then $d(x, y) = \ell/2 - 2|x \wedge y|$, so in the notation of Claim 22, $P(i) \le 1/4$ for $i \ge \ell/8 + \sqrt{\ell}/2$, and $P(i) \ge 3/4$ for $i \le \ell/8 - \sqrt{\ell}/2$. Taking $d = O(Q)$, there is a degree-d polynomial q such that $q(i) \le 1/3$ for $i \ge \ell/8 + \sqrt{\ell}/2$, $q(i) \ge 2/3$ for $i \le \ell/8 - \sqrt{\ell}/2$. By Claim 23, we must have $d = \Omega(\sqrt{\ell})$. Thus $TS = \Omega(\sqrt{\ell}) = \Omega(1/\epsilon)$, completing the proof. □

4 Outlook

We have initiated the study of the quantum complexity of approximately computing the frequency moments. However, there are still many open questions to be resolved. In particular:

- What is the quantum query complexity of approximately computing F_∞? As discussed in Section 2.3, this seems to be closely connected to a "gapped" version of the well-studied k-distinctness problem, whose precise complexity is still unknown. For each $k > 2$, improved bounds on k-distinctness would also improve our algorithm for computing F_k.
- What is the quantum streaming complexity of approximating F_k for $k > 2$? Just as the efficient quantum algorithm for F_2 is based on the streaming algorithm of Alon, Matias and Szegedy [3], it would be interesting to determine whether more recent streaming algorithms for F_k [28, 12, 14] could be used to obtain efficient quantum algorithms.

It would also be interesting to find a practically relevant problem demonstrating a separation between quantum and classical streaming complexity in the one-way model. This could involve proving a separation between one-way quantum and classical communication complexity for a total function, which is a major open problem.

Acknowledgements

I would like to thank Raphaël Clifford for suggesting the question which inspired this paper, and Alexander Belovs for sending me a copy of [5]. This work was supported by EPSRC Early Career Fellowship EP/L021005/1.

A Proofs of combinatorial bounds

Lemma 4 (restated). *Fix ℓ such that $1 \le \ell \le n$. Let $s_1, \dots, s_\ell \in [n]$ be picked uniformly at random and define*

$$C_k(s_1, \dots, s_\ell) := |\{T \in \binom{[\ell]}{k} : a_{s_i} = a_{s_j} \text{ for all } i, j \in T\}|.$$

Then

$$\mathbb{E}_{s_1,\dots,s_\ell}[C_k(s_1,\dots,s_\ell)] = \frac{\binom{\ell}{k} F_k}{n^k}$$

and

$$\mathrm{Var}(C_k) \le \sum_{q=k}^{2k-1} \left(\frac{\ell F_k^{1/k}}{n} \right)^q.$$

Proof. First observe that, for any set $T \in \binom{[\ell]}{k}$,

$$\Pr_{s_1,\dots,s_\ell}\left[a_{s_i} = a_{s_j} \text{ for all } i,j \in T\right] = \frac{1}{n^k} \sum_{p_1,\dots,p_k=1}^{n} [a_{p_1} = \dots = a_{p_k}] = \frac{F_k}{n^k}. \quad (2)$$

For the expectation, we compute

$$\begin{aligned}
\mathbb{E}_{s_1,\dots,s_\ell}[C_k(s_1,\dots,s_\ell)] &= \mathbb{E}_{s_1,\dots,s_\ell}[|\{T \in \binom{[\ell]}{k} : a_{s_i} = a_{s_j} \text{ for all } i,j \in T\}|] \\
&= \sum_{T \in \binom{[\ell]}{k}} \Pr_{s_1,\dots,s_\ell}\left[a_{s_i} = a_{s_j} \text{ for all } i,j \in T\right] \\
&= \frac{\binom{\ell}{k} F_k}{n^k}.
\end{aligned}$$

We now bound the variance. We have

$$\begin{aligned}
&\mathbb{E}_{s_1,\dots,s_\ell}[C_k(s_1,\dots,s_\ell)^2] \\
= \;& \mathbb{E}_{s_1,\dots,s_\ell}\left[\left(\sum_{T \in \binom{[\ell]}{k}} [a_{s_i} = a_{s_j} \text{ for all } i,j \in T] \right)^2 \right] \\
= \;& \sum_{T,U \in \binom{[\ell]}{k}} \Pr_{s_1,\dots,s_\ell}[a_{s_i} = a_{s_j} \text{ for all } i,j \in T, \text{ and } a_{s_p} = a_{s_q} \text{ for all } p,q \in U]
\end{aligned}$$

$$\begin{aligned}
&= \sum_{T,U \in \binom{[\ell]}{k}, T\cap U \neq \emptyset} \Pr_{s_1,\dots,s_\ell}[a_{s_i} = a_{s_j} \text{ for all } i,j \in T, \text{ and } a_{s_p} = a_{s_q} \text{ for all } p,q \in U] \\
&\quad + \sum_{T,U \in \binom{[\ell]}{k}, T\cap U = \emptyset} \Pr_{s_1,\dots,s_\ell}[a_{s_i} = a_{s_j} \text{ for all } i,j \in T, \text{ and } a_{s_p} = a_{s_q} \text{ for all } p,q \in U] \\
&= \sum_{T,U \in \binom{[\ell]}{k}, T\cap U \neq \emptyset} \Pr_{s_1,\dots,s_\ell}[a_{s_i} = a_{s_j} \text{ for all } i,j \in T \cup U] \\
&\quad + \sum_{T,U \in \binom{[\ell]}{k}, T\cap U = \emptyset} \Pr_{s_1,\dots,s_\ell}[a_{s_i} = a_{s_j} \text{ for all } i,j \in T] \Pr_{s_1,\dots,s_\ell}[a_{s_i} = a_{s_j} \text{ for all } i,j \in U] \\
&= \sum_{q=k}^{2k-1} |\{T,U \in \binom{[\ell]}{k} : |T \cup U| = q\}| \frac{F_q}{n^q} + |\{T,U \in \binom{[\ell]}{k} : |T\cup U| = 2k\}| \left(\frac{F_k}{n^k}\right)^2,
\end{aligned}$$

where the final equality is (2). We have the rough bound that

$$|\{T,U \in \binom{[\ell]}{k} : |T \cup U| = q\}| \leq \ell^{2k-q}\ell^{2(q-k)} = \ell^q,$$

because we can specify T and U by picking the $|T \cap U| = 2k - q$ elements in their intersection, then the $q-k$ elements in each set $(T\cup U)\backslash T$, $(T \cup U)\backslash U$. We also have

$$F_q^{1/q} = \left(\sum_j n_j^q\right)^{1/q} \leq \left(\sum_j n_j^k\right)^{1/k} = F_k^{1/k}$$

for any $q \geq k$, by monotonicity of ℓ_p norms. Combining these two bounds,

$$\mathbb{E}_{s_1,\dots,s_\ell}[C_k(s_1,\dots,s_\ell)^2] \leq \sum_{q=k}^{2k-1} \left(\frac{\ell F_k^{1/k}}{n}\right)^q + \binom{\ell}{k}\binom{\ell-k}{k}\left(\frac{F_k}{n^k}\right)^2.$$

Therefore

$$\begin{aligned}
\mathrm{Var}(C_k) &= \mathbb{E}_{s_1,\dots,s_\ell}[C_k(s_1,\dots,s_\ell)^2] - (\mathbb{E}_{s_1,\dots,s_\ell}[C_k(s_1,\dots,s_\ell)])^2 \\
&\leq \sum_{q=k}^{2k-1} \left(\frac{\ell F_k^{1/k}}{n}\right)^q - \binom{\ell}{k}\left(\frac{F_k}{n^k}\right)^2 \left(\binom{\ell}{k} - \binom{\ell-k}{k}\right) \\
&\leq \sum_{q=k}^{2k-1} \left(\frac{\ell F_k^{1/k}}{n}\right)^q
\end{aligned}$$

as claimed. □

References

[1] S. Aaronson and Y. Shi. Quantum lower bounds for the collision and the element distinctness problems. *J. ACM*, 51(4):595–605, 2004.

[2] N. Alon, P. Gibbons, Y. Matias, and M. Szegedy. Tracking join and self-join sizes in limited storage. In *Proc. eighteenth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '99)*, pages 10–20, 1999.

[3] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. In *Proc. 28th Annual ACM Symp. Theory of Computing*, pages 20–29, 1996.

[4] A. Ambainis. Quantum walk algorithm for element distinctness. In *Proc. 45th Annual Symp. Foundations of Computer Science*, pages 22–31, 2004. `quant-ph/0311001`.

[5] A. Ambainis, A. Belovs, O. Regev, and R. de Wolf. Efficient quantum algorithms for (gapped) group testing and junta testing, 2015. Personal communication.

[6] Z. Bar-Yossef. *The Complexity of Massive Data Set Computations.* PhD thesis, University of California at Berkeley, 2002.

[7] Z. Bar-Yossef, T. S. Jayram, S. R. Kumar, D. Sivakumar, and L. Trevisan. Counting distinct elements in a data stream. In *Proc. RANDOM'02*, pages 1–10, 2002.

[8] R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. *J. ACM*, 48(4):778–797, 2001. `quant-ph/9802049`.

[9] P. Beame, R. Clifford, and W. Machmouchi. Element distinctness, frequency moments, and sliding windows. In *Proc. 54th Annual Symp. Foundations of Computer Science*, pages 290–299, 2013. `arXiv:1309.3690`.

[10] P. Beame and W. Machmouchi. The quantum query complexity of AC0. *Quantum Inf. Comput.*, 12(7&8):670–676, 2012. `arXiv:1008.2422`.

[11] A. Belovs. Learning-graph-based quantum algorithm for k-distinctness. In *Proc. 53rd Annual Symp. Foundations of Computer Science*, pages 207–216, 2012. `arXiv:1205.1534`.

[12] L. Bhuvanagiri, S. Ganguly, D. Kesh, and C. Seha. Simpler algorithm for estimating frequency moments of data streams. In *Proc. 17th ACM-SIAM Symp. Discrete Algorithms*, pages 708–713, 2006.

[13] G. Brassard, P. Høyer, M. Mosca, and A. Tapp. Quantum amplitude amplification and estimation. *Quantum Computation and Quantum Information: A Millennium Volume*, pages 53–74, 2002. `quant-ph/0005055`.

[14] V. Braverman, J. Katzman, C. Seidell, and G. Vorsanger. An optimal algorithm for large frequency moments using $O(n^{1-2/k})$ bits. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*, pages 531–544, 2014. `http://drops.dagstuhl.de/opus/volltexte/2014/4721`.

[15] H. Buhrman, R. Cleve, S. Massar, and R. de Wolf. Non-locality and communication complexity. *Rev. Mod. Phys.*, 82(1):665–698, 2010. `arXiv:0907.3584`.

[16] H. Buhrman and R. de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288:21–43, 2002.

[17] H. Buhrman, C. Dürr, M. Heiligman, P. Høyer, F. Magniez, M. Santha, and R. de Wolf. Quantum algorithms for element distinctness. *SIAM J. Comput.*, 34(6):1324–1330, 2005. `quant-ph/0007016`.

[18] A. Chakrabarti and O. Regev. An optimal lower bound on the communication complexity of Gap-Hamming-Distance. *SIAM J. Comput.*, 41(5):1299–1317, 2012. `arXiv:1009.3460`.

[19] M. Charikar, S. Chaudhuri, R. Motwani, and V. Narasayya. Towards estimation error guarantees for distinct values. In *Proc. PODS'00*, pages 268–279, 2000.

[20] M. Coffey and Z. Prezkuta. A quantum algorithm for finding the modal value. *Quantum Information Processing*, 7(1):51–54, 2008.

[21] C. Dürr, M. Heiligman, P. Høyer, and M. Mhalla. Quantum query complexity of some graph problems. In *Proc. 31st International Conference on Automata, Languages and Programming (ICALP'04)*, pages 481–493, 2004. `quant-ph/0401091`.

[22] C. Dürr and P. Høyer. A quantum algorithm for finding the minimum, 1996. `quant-ph/9607014`.

[23] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *J. Comput. Syst. Sci.*, 31:182–209, 1985.

[24] D. Gavinsky, J. Kempe, I. Kerenidis, R. Raz, and R. de Wolf. Exponential separation for one-way quantum communication complexity, with applications to cryptography. *SIAM J. Comput.*, 38(5):1695–1708, 2009. `quant-ph/0611209`.

[25] O. Goldreich and D. Ron. On testing expansion in bounded-degree graphs. Technical Report TR00-020, Electronic Colloquium on Computational Complexity, 2000.

[26] P. Høyer and R. Špalek. Lower bounds on quantum query complexity. *Bulletin of the European Association for Theoretical Computer Science*, 87:78–103, 2005. `quant-ph/0509153`.

[27] W. Huang, Y. Shi, S. Zhang, and Y. Zhu. The communication complexity of the Hamming distance problem. *Information Processing Letters*, 99(4):149–153, 2006. `quant-ph/0509181`.

[28] P. Indyk and D. Woodruff. Tight lower bounds for the distinct elements problem. In *Proc. 44th Annual Symp. Foundations of Computer Science*, pages 283–288, 2003.

[29] P. Indyk and D. Woodruff. Optimal approximations of the frequency moments of data streams. In *Proc. 37th Annual ACM Symp. Theory of Computing*, pages 202–208, 2005.

[30] D. Kane, J. Nelson, and D. Woodruff. An optimal algorithm for the distinct elements problem. In *Proc. twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems (PODS'10)*, pages 41–52, 2010.

[31] A. Kara. A quantum algorithm for finding an ϵ-approximate mode. Master's thesis, University of Waterloo, 2005.

[32] H. Klauck, R. Špalek, and R. de Wolf. Quantum and classical strong direct product theorems and optimal time-space tradeoffs. *SIAM J. Comput.*, 36(5):1472–1493, 2007.

[33] I. Kremer. Quantum communication. Master's thesis, Hebrew University, 1995.

[34] F. Le Gall. Exponential separation of quantum and classical online space complexity. In *Proc. 18th ACM SPAA*, pages 67–73, 2006. `quant-ph/0606066`.

[35] A. Montanaro. Quantum speedup of Monte Carlo methods, 2015. `arXiv:1504.06987`.

[36] C. Moore and J. Crutchfield. Quantum automata and quantum grammars. *Theoretical Computer Science*, 237(1–2):275–306, 2000. `quant-ph/9707031`.

[37] A. Nayak and F. Wu. The quantum query complexity of approximating the median and related statistics. In *Proc. 31st Annual ACM Symp. Theory of Computing*, pages 384–393, 1999. `quant-ph/9804066`.

[38] A. A. Razborov. Quantum communication complexity of symmetric predicates. *Izvestiya of the Russian Academy of Science*, 67:145–159, 2003. `quant-ph/0204025`.

[39] D. Woodruff. Optimal space lower bounds for all frequency moments. In *Proc. 15th ACM-SIAM Symp. Discrete Algorithms*, pages 167–175, 2004.

[40] D. Woodruff and Q. Zhang. Tight bounds for distributed functional monitoring. In *Proc. 44th Annual ACM Symp. Theory of Computing*, pages 941–960, 2012. `arXiv:1112.5153`.