Chapter 5

Streaming Algorithms II: Frequency Moments

We continue our study of streaming algorithms by considering another family of key problems in this model: *frequency moment* estimation.

5.1 Frequency Moments Estimation

Let us now consider a general family of problems in the streaming model. Suppose we receive a stream of elements $e_1, \ldots, e_n$ from the universe $[m]$. The **frequency vector** $f \in \mathbb{N}^m$ of the stream is defined as the m-element vector such that the i-th coordinate f_i is the number of times i appears in the stream.

A typical class of problems in streaming algorithms is that of computing some function of the frequency vector. Among such problems, an important class is that of *frequency moment* estimation problems: For any integer $p \geqslant 0$, we define the ***p*-th frequency moment** as

$$F_p = \|f\|_p^p = \sum_{i=1}^{m} f_i^p;$$

also, $F_0 := \|f\|_0$ is the number of non-zero entries of f and $F_\infty = \|f\|_\infty$ is the maximum of f.

The frequency moment estimation problem is defined as follows.

Problem 5.1. Given an integer $p \geqslant 0$ and a stream of n elements from the universe $[m]$, defining a frequency vector f, estimate F_p.

Particularly interesting instances of this problem are the following:

- F_0 is the number of distinct elements in the stream. This is what we already studied in the previous chapter.
- F_1 is simply the length of the stream, i.e., $F_1 = n$.
- $F_2 = \|f\|_2^2$. While F_2 may not have a 'direct' interpretation as F_0 and F_1, it is related to how *concentrated* (non-uniform) the frequency vector is: the larger F_2 is, the further away the frequency vector is from "uniform", that is, it has more mass on fewer elements.

We already know how to estimate F_0 from the previous chapter. We now see algorithms for F_1 and F_2. It is worth noting that when studying F_0 and F_2, for simplicity, we assume we know the value of n but that cannot be the case for F_1, as the entire goal is to estimate n itself.

5.2 First Frequency Moment: Morris Counter

The problem we would like to solve now, namely, F_1-estimation, sounds entirely trivial. We are just seeing n numbers (for an unknown n), and simply need to count how many numbers we have seen! Of course, we can do this using $O(\log n)$ bits trivially by just maintaining a counter. But can we do even better (if we allow randomization and approximation)?

A result of [Mor78], referred to as the **Morris Counter**, shows that we can indeed.

Theorem 5.2 ([Mor78])**.** *There is an algorithm for maintaining a counter of n elements that outputs a $(1 \pm \varepsilon)$-approximation to n with probability at least $1 - \delta$ using*

$$O(\log \log n + \log (1/\varepsilon) + \log (1/\delta))$$

bits of space in expectation.

The trivial solution for this problem is to store a counter X that is incremented after each element. Thus, the counter itself will have value n at the end and requires $\Theta(\log n)$ bits to store. The Morris counter also maintains a counter X. However, it only increases it for each arriving element *probabilistically*, namely, with some probability $(1+\alpha)^{-X}$ for a carefully chosen $\alpha \in (0, 1)$. This way, the larger X becomes, the less likely it is for a single arriving element to update X, meaning the value of X itself is "small" and more space-efficient to maintain. At the same time, to get a $(1 \pm \varepsilon)$-approximation for "large" values of n, we only need a few updates to happen between $(1 - \varepsilon)n$-th element arriving and n-th element arriving to "signal" their difference – after that, we return the final answer based on some carefully chosen $f(X)$ as well. Figure 5.1 illustrates this behavior.

Algorithm 5.3.

1. Let $X = 0$ and let $\alpha = \varepsilon^2 \delta$.
2. After each element arrives, increment $X \leftarrow X + 1$ with probability $(1 + \alpha)^{-X}$.
3. Return
$$A := \frac{1}{\alpha} \cdot \left((1 + \alpha)^{X_n} - 1\right)$$
as the answer.

Let us prove that Algorithm 5.3 outputs an unbiased estimator of the answer.

We need to bound the expectation and the variance of the returned answer by Algorithm 5.3.

Claim 5.4. $\mathbb{E}[A] = n$.

Proof. The idea of the proof is to show that each arriving element, *in expectation*, leads to the increment of one in the final estimate A. Let X_t denote the variable X after t-th element arrives and $Y_t = (1 + \alpha)^{X_t}$ for any $t \in [n]$. We have

$$\mathbb{E}[Y_n] = 1 + \sum_{t=1}^{n} \mathbb{E}[Y_t - Y_{t-1}] \tag{5.1}$$

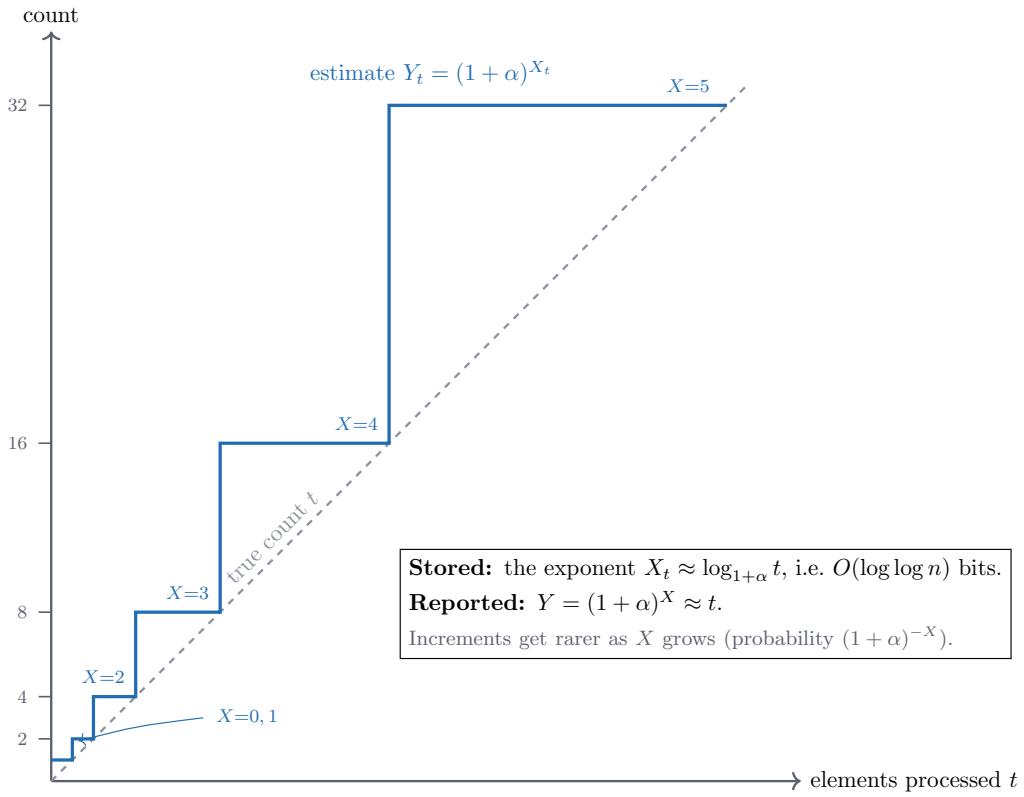

Figure 5.1: The Morris counter (drawn for $1+\alpha = 2$). The estimate $Y_t = (1+\alpha)^{X_t}$ (blue) climbs in geometric jumps and stays within a factor $(1+\alpha)$ of the true count t (dashed diagonal), while the algorithm stores only the *exponent* $X_t \approx \log_{1+\alpha} t$—about $\log\log n$ bits—and reports $(1+\alpha)^X$.

by a telescoping sum and defining $Y_0 = 1$. For any $y > 0$, we have

$$\mathbb{E}\left[Y_t - Y_{t-1} \mid Y_{t-1} = y\right] = \frac{1}{y} \cdot ((1+\alpha)\cdot y - y) = \alpha;$$

the first equality holds because with probability $1/y$ the counter X_t is incremented by one, leading to increasing Y_t by a factor of $(1+\alpha)$ compared to Y_{t-1}. The RHS of above, regardless of y, is always α which, together with the law of conditional expectation, implies that

$$\mathbb{E}\left[Y_t - Y_{t-1}\right] = \mathop{\mathbb{E}}_{y\sim Y_{t-1}} \mathbb{E}\left[Y_t - Y_{t-1} \mid Y_{t-1} = y\right] = \alpha.$$

Plugging in this bound in Eq (5.1) gives us

$$\mathbb{E}\left[Y_n\right] = 1 + n\cdot\alpha. \tag{5.2}$$

Given the definition of $Y_n = (1+\alpha)^{X_n}$, we have

$$\mathbb{E}\left[A\right] = \mathbb{E}\left[(1+\alpha)^{X_n} - 1\right] \cdot \frac{1}{\alpha} = n,$$

as desired. □

The next step is to bound the variance of our estimator.

Claim 5.5. $\mathrm{Var}\left[A\right] \leqslant \alpha \cdot n^2$.

Proof. Define X_t and Y_t for any $t \in [n]$ as in Claim 5.4. Since $A = \frac{1}{\alpha}(Y_n - 1)$ and $\mathbb{E}[A] = n$ by Claim 5.4, we have,

$$\mathrm{Var}[A] = \mathbb{E}\left[A^2\right] - \mathbb{E}[A]^2 = \frac{1}{\alpha^2} \cdot \mathbb{E}\left[(Y_n - 1)^2\right] - n^2 = \frac{1}{\alpha^2}\left(\mathbb{E}\left[Y_n^2\right] - 2\,\mathbb{E}[Y_n] + 1\right) - n^2. \quad (5.3)$$

It thus remains to compute $\mathbb{E}\left[Y_n^2\right]$, which we do similar to Claim 5.4: conditioned on $Y_{n-1} = y$, with probability $1/y$ the counter is incremented and $Y_n = (1+\alpha) \cdot y$; otherwise $Y_n = y$. Hence,

$$\begin{aligned}
\mathbb{E}\left[Y_n^2\right] &= \mathop{\mathbb{E}}_{y \sim Y_{n-1}} \mathbb{E}\left[Y_n^2 \mid Y_{n-1} = y\right] && \text{(by the law of conditional expectation)} \\
&= \mathop{\mathbb{E}}_{y \sim Y_{n-1}} \left[\frac{1}{y} \cdot (1+\alpha)^2 \cdot y^2 + \left(1 - \frac{1}{y}\right) \cdot y^2\right] && \text{(by the argument above)} \\
&= \mathop{\mathbb{E}}_{y \sim Y_{n-1}} \left[y^2 + (\alpha^2 + 2\alpha) \cdot y\right] && \text{(by expanding the terms)} \\
&= \mathbb{E}\left[Y_{n-1}^2\right] + (\alpha^2 + 2\alpha) \cdot \mathbb{E}[Y_{n-1}] && \text{(by linearity of expectation)} \\
&= \mathbb{E}\left[Y_{n-1}^2\right] + (\alpha^2 + 2\alpha) \cdot (1 + \alpha(n-1)). && \text{(as } \mathbb{E}[Y_t] = 1 + \alpha \cdot t \text{ by Eq (5.2))}
\end{aligned}$$

Applying this recurrence repeatedly and using $\mathbb{E}\left[Y_0^2\right] = 1$, we obtain

$$\begin{aligned}
\mathbb{E}\left[Y_n^2\right] &= \mathbb{E}\left[Y_0^2\right] + \sum_{t=1}^{n} (\alpha^2 + 2\alpha) \cdot (1 + \alpha \cdot (t-1)) \\
&= 1 + (\alpha^2 + 2\alpha)\left(n + \alpha \cdot \frac{n \cdot (n-1)}{2}\right). && \text{(as } \textstyle\sum_{t=1}^{n}(t-1) = \frac{n \cdot (n-1)}{2}\text{)}
\end{aligned}$$

Since $\mathbb{E}[Y_n] = 1 + \alpha \cdot n$ by Eq (5.2), a direct calculation gives

$$\mathbb{E}\left[Y_n^2\right] - 2\,\mathbb{E}[Y_n] + 1 = (\alpha^2 + 2\alpha)\left(n + \alpha \cdot \tfrac{n(n-1)}{2}\right) - 2\alpha n = \alpha^2 n + (\alpha^3 + 2\alpha^2) \cdot \frac{n(n-1)}{2}.$$

Plugging this into Eq (5.3), the factor $1/\alpha^2$ cancels and

$$\mathrm{Var}[A] = n + (\alpha + 2) \cdot \frac{n(n-1)}{2} - n^2 = \alpha \cdot \frac{n(n-1)}{2} \leqslant \alpha \cdot n^2,$$

where the second equality uses $n + 2 \cdot \frac{n(n-1)}{2} = n^2$. This concludes the proof. □

We are now ready to conclude the proof of Theorem 5.2.

Proof of Theorem 5.2. Firstly, by Chebyshev's inequality (Proposition 2.5),

$$\Pr\left(|A - \mathbb{E}[A]| \geqslant \varepsilon \cdot \mathbb{E}[A]\right) \leqslant \frac{\mathrm{Var}[A]}{\varepsilon^2 \cdot \mathbb{E}[A]^2} \leqslant \frac{\alpha \cdot n^2}{\varepsilon^2 \cdot n^2} = \delta,$$

using the bounds in Claim 5.4 and Claim 5.5 and by the choice of $\alpha = \varepsilon^2\delta$. Given $\mathbb{E}[A] = n$, this proves the correctness of the algorithm.

We now analyze the space complexity of the algorithm which is $O(\log X)$ bits for storing the counter X. Define $Y = (1+\alpha)^X$ and thus $X = \log_{1+\alpha} Y$. We first state the following highly useful inequality, called the Jensen's inequality.

Fact 5.6 (Jensen's Inequality)**.** *For any concave function $f : \mathbb{R} \to \mathbb{R}$ and random variable Z, $\mathbb{E}[f(Z)] \leqslant f(\mathbb{E}[Z])$ (recall that f is concave iff its second derivative is always non-positive).*

We have,

$$\begin{aligned}
\mathbb{E}[\log X] &\leqslant \log(\mathbb{E}[X]) \qquad \text{(by Jensen's inequality (Fact 5.6) and concavity of log-function)} \\
&= \log\left(\mathbb{E}\left[\log_{(1+\alpha)} Y\right]\right) = \log(\mathbb{E}[\log Y] / \log(1+\alpha)) \\
&\qquad \text{(by the definition of } X \text{ and since } \log_y x = \log x / \log y) \\
&= \log(\mathbb{E}[\log Y]) - \log\log(1+\alpha) \qquad \text{(as } \log(x/y) = \log x - \log y) \\
&\leqslant \log\log \mathbb{E}[Y] - \log(\alpha/2) \\
&\qquad \text{(another application of Jensen's inequality and since } (1+\alpha) > 2^{\alpha/2} \text{ for } \alpha \in (0,1)) \\
&\leqslant \log\log n + \log\left(\frac{4}{\varepsilon^2 \cdot \delta}\right) \qquad \text{(by Eq (5.2), } \mathbb{E}[Y] \leqslant n \text{ and by the choice of } \alpha) \\
&= O(\log\log n + \log(1/\varepsilon) + \log(1/\delta)),
\end{aligned}$$

concluding the proof. □

Remark. Morris counter was first introduced in [Mor78] for a special case when $\alpha = 1$ to achieve a 2-approximation. It was subsequently analyzed by [Fla85] for values of $\alpha \in (0,1)$ and the approach presented here is similar to this proof. More recently, [NY22] proved that in fact, Morris counter with parameter $\alpha = O(\varepsilon^2/\log(1/\delta))$ can already solve the problem using space $O(\log\log n + \log(1/\varepsilon) + \log\log(1/\delta))$, that is, an exponentially better space dependence on the parameter δ compared to Theorem 5.2 ([NY22] also proves that this space complexity is optimal for any streaming algorithm for estimating F_1).

5.3 Second Frequency Moment: AMS Sketch

We now turn our attention to F_2 estimation. Let $f^{(i)}$ be the frequency vector after i elements have been received. The trivial algorithm for this problem is to maintain the entire $f^{(i)}$ and update it at each step, which requires $O(m \log n)$ bits. What if we only maintain a small "sketch" of $f^{(i)}$, that uses fewer bits, and that allows us to recover some aggregate information about $f^{(i)}$? To realize this idea, we use the notion of *linear sketching*.

Definition 5.7. For any time $i \in [n]$, we define a **linear sketch** of the $f^{(i)}$ as a vector $A \cdot f^{(i)}$, where A is a $t \times m$ matrix chosen *independently* of f (and almost always randomly).

Linear sketches are easy to maintain in the streaming model: since the frequency vector is updated linearly as $f^{(i)} = f^{(i-1)} + \mathbf{1}_{e_i}$ (here, $\mathbf{1}_j$ is the m-coordinate vector that has 1 in the j-th coordinate and 0 everywhere else), we can update $A \cdot f^{(i)} = A \cdot f^{(i-1)} + A \cdot \mathbf{1}_{e_i}$ by linearity, hence, at the end of the stream have $A \cdot f$, namely, the linear sketch of the entire frequency vector.

In this approach, the number of rows t of A controls the tradeoff between the number of bits required by the sketch and the "power" of $A \cdot f$ in recovering information about f – the smaller the t, the less bits are required, but (presumably) the less information we will recover from f. Finally, since A often consists entirely of random bits, we typically store A *implicitly* using limited-independence hash functions (as was done in the previous chapter). Figure 5.2 gives an illustration of the linear sketch we will use and how it is updated in the stream.

We use linear sketching to prove the following theorem due to [AMS96], often referred to as the **AMS sketch**.

Theorem 5.8 ([AMS96]). *There is a (linear sketching) streaming algorithm that outputs a $(1 \pm \varepsilon)$-estimation of F_2 with probability at least $1 - \delta$, using $O(\frac{1}{\varepsilon^2} \cdot (\log m + \log n) \cdot \log(1/\delta))$ bits of space.*

We are now going to design a linear sketch for the F_2 estimation problem.

Algorithm 5.9.

1. Let $t := 40/\varepsilon^2$ and $\mathcal{H}$ be a 4-wise independent family of hash functions $[m] \to \{-1, 1\}$.
2. For each $i \in [t]$, define the i-th row of the matrix $A \in \{-1, 1\}^{t \times m}$ as follows:
   - Pick $h_i \in \mathcal{H}$ at random (independently of other rows) and let
   $$A_i := [h_i(1)\ h_i(2) \cdots h_i(m)],$$
   be the i-th row of A.
3. Let $s = \mathbf{0}^t$, namely, the all-zero vector of dimension t as the sketch. For each arriving element e_j in the stream, update $s \leftarrow s + A \cdot \mathbf{1}_{e_j}$ (as in Definition 5.7).
4. At the end of the stream, $Y := \frac{1}{t} \cdot \sum_{i=1}^{t} s_i^2$.

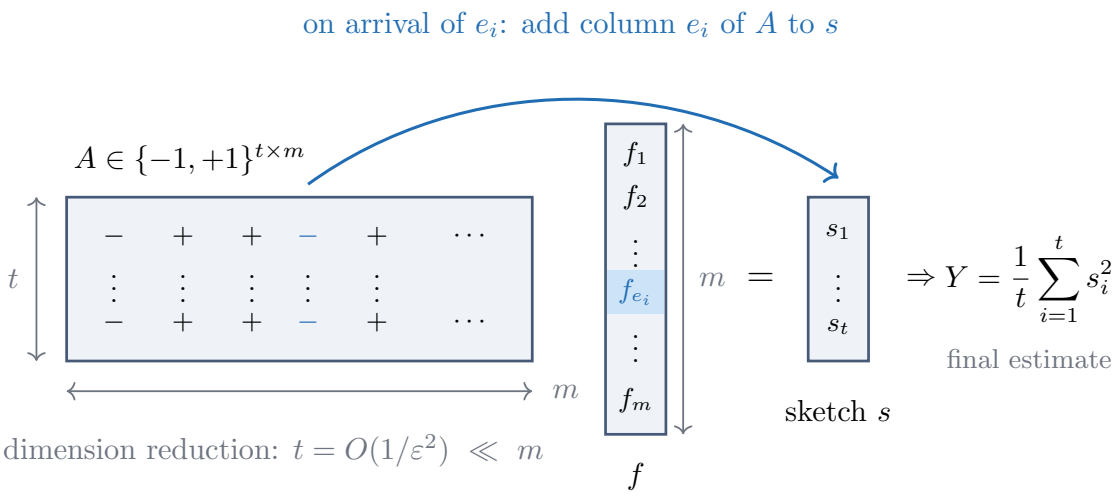

Figure 5.2: Linear sketching for F_2. The sketch is the matrix–vector product $s = A \cdot f$, where $A \in \{-1, +1\}^{t \times m}$ is a random sign matrix with $t = O(1/\varepsilon^2) \ll m$ and f is the frequency vector. In the stream, each arriving element e_i simply adds the e_i-th *column* of A to the running sketch s; the final estimate is $Y = \frac{1}{t} \sum_{i=1}^{t} s_i^2$.

The space complexity of this algorithm is $O(t \cdot \log m)$ bits for storing A using t independent 4-wise independent hash functions (each $O(\log m)$ bits by Proposition 4.8) and another $O(t \cdot \log n)$ bits for storing the t-dimensional vector s (i.e., the sketch). Hence, the space complexity is

$$O(t \cdot (\log n + \log m)) = O(\frac{1}{\varepsilon^2} \cdot (\log n + \log m)) \text{ bits.}$$

We now prove the correctness of the algorithm in the following lemma.

Lemma 5.10. *For the random variable Y of Algorithm 5.9,*

$$\Pr\left(|Y - F_2| \geqslant \varepsilon \cdot F_2\right) \leqslant \frac{1}{10}.$$

We prove this by computing expected value and bounding variance of each s_i^2 (which are distributed identically), extending these bounds to Y (as an average of t such random variables), and using Chebyshev's inequality to conclude the proof. We start with the expected values.

Claim 5.11. *Let s be distributed as any s_i in Algorithm 5.9; then, $\mathbb{E}\left[s^2\right] = F_2$.*

Proof. For brevity, in the following, we use $a = [a_1, \ldots, a_m]$ to denote a random row of the matrix A such that $s = a \cdot f$. Note that by definition,

$$s^2 = \left(\sum_{i=1}^{m} a_i \cdot f_i\right)^2 = \sum_{i=1}^{m}\sum_{j=1}^{m} a_i \cdot a_j \cdot f_i \cdot f_j.$$

Using this, we can write,

$$\begin{aligned}
\mathbb{E}\left[s^2\right] &= \sum_{i=1}^{m}\sum_{j=1}^{m} \mathbb{E}\left[a_i \cdot a_j\right] \cdot f_i \cdot f_j && \text{(by linearity of expectation)} \\
&= \sum_{i=1}^{m} \mathbb{E}\left[a_i^2\right] \cdot f_i^2 + \sum_{i \neq j} \mathbb{E}\left[a_i \cdot a_j\right] \cdot f_i \cdot f_j.
\end{aligned}$$

Now note that $\mathbb{E}\left[a_i \cdot a_j\right] = \mathbb{E}\left[a_i\right] \cdot \mathbb{E}\left[a_j\right]$ for $i \neq j$ because a-values are 4-wise independent and thus a_i, a_j are independent (for this part of the argument, pairwise independence suffices and we only need 4-wise independence for later in the proof). We can thus simplify the above equation to,

$$\begin{aligned}
\mathbb{E}\left[s^2\right] &= \sum_{i=1}^{m} \mathbb{E}\left[a_i^2\right] \cdot f_i^2 + \sum_{i \neq j} \mathbb{E}\left[a_i\right] \cdot \mathbb{E}\left[a_j\right] \cdot f_i \cdot f_j && \text{(as } a_i \perp a_j \text{ for } i \neq j\text{)} \\
&= \sum_{i=1}^{m} f_i^2 = F_2 && \text{(as } \mathbb{E}\left[a_i^2\right] = 1 \text{ and } \mathbb{E}\left[a_i\right] = \mathbb{E}\left[a_j\right] = 0\text{)}
\end{aligned}$$

as desired. □

Claim 5.12. *Let s be as in Claim 5.11; then, $\mathrm{Var}\left[s^2\right] \leqslant 4 \cdot F_2^2$.*

Proof. As before, we expand $s^4 = \left(\sum_{i=1}^{m} a_i \cdot f_i\right)^4$ and use linearity of expectation:

$$\mathrm{Var}\left[s^2\right] \leqslant \mathbb{E}\left[s^4\right] = \sum_{i,j,k,\ell} \mathbb{E}\left[a_i \cdot a_j \cdot a_k \cdot a_\ell\right] \cdot f_i \cdot f_j \cdot f_k \cdot f_\ell.$$

Because the a-values are 4-wise independent with $\mathbb{E}\left[a_i\right] = \mathbb{E}\left[a_i^3\right] = 0$, a term $\mathbb{E}\left[a_i a_j a_k a_\ell\right]$ is nonzero only when *every* index appearing in it does so an even number of times; any term in which some index appears an odd number of times vanishes. Only two patterns survive: all four indices are equal, or the indices form two distinct pairs. Moreover, we also have that $\mathbb{E}\left[a_i^2\right] = \mathbb{E}\left[a_i^4\right] = 1$. As such:

- the all-equal terms contribute $\sum_i \mathbb{E}\left[a_i^4\right]\ f_i^4 = \sum_i f_i^4$;
- for the two-pair terms, there are exactly 3 ways to split the four factors into two pairs (namely $a_i a_j \,|\, a_k a_\ell$, $a_i a_k \,|\, a_j a_\ell$, and $a_i a_\ell \,|\, a_j a_k$), and each split contributes

$$\sum_{i \neq j} \mathbb{E}\left[a_i^2\right] \mathbb{E}\left[a_j^2\right]\ f_i^2 f_j^2 = \sum_{i \neq j} f_i^2 f_j^2.$$

Therefore,

$$\begin{aligned}\operatorname{Var}\left[s^2\right] \leqslant \mathbb{E}\left[s^4\right] &= \sum_{i=1}^{m} f_i^4 + 3\sum_{i\neq j} f_i^2 \cdot f_j^2 \\ &\leqslant \left(\sum_{i=1}^{m} f_i^2\right)^2 + 3 \cdot \left(\sum_{i=1}^{m} f_i^2\right)^2 = 4F_2^2,\end{aligned}$$

as desired. □

We can now conclude the proof of Lemma 5.10.

Proof of Lemma 5.10. By linearity of expectation and Claim 5.11, $\mathbb{E}[Y] = F_2$. Moreover, since each row of A is chosen independently, by Proposition 4.9,

$$\operatorname{Var}[Y] = \operatorname{Var}\left[\frac{1}{t} \cdot \sum_{i=1}^{t} s_i^2\right] = \frac{1}{t^2} \cdot t \cdot \operatorname{Var}\left[s^2\right] = \frac{1}{t} \cdot \operatorname{Var}\left[s^2\right] \leqslant \frac{1}{t} \cdot 4F_2^2,$$

by Claim 5.12. Thus, by Chebyshev's inequality (Proposition 2.5),

$$\begin{aligned}\Pr\left(|Y - F_2| \geqslant \varepsilon \cdot F_2\right) &= \Pr\left(|Y - \mathbb{E}[Y]| \geqslant \varepsilon \cdot F_2\right) \\ &\leqslant \frac{4F_2^2}{t \cdot \varepsilon^2 \cdot F_2^2} = \frac{1}{10},\end{aligned}$$

by the choice of $t = 40/\varepsilon^2$, finalizing the proof. □

As in Problem 2.11, we can always boost the probability of success of this algorithm by running it $O(\ln(1/\delta))$ times in parallel and taking the median answer. In terms of linear sketching, this corresponds to using a sketching matrix $A' \in \{-1, 1\}^{t' \times m}$ where $t' = O(\ln(1/\delta)) \cdot t$; each block of t consecutive rows form one independent instance of the sketching matrix A in this section, and there are $O(\ln(1/\delta))$ such blocks. For recovery, we first recover the answer to each of A-sub-matrices using the algorithm of this section and then take a median (so in a sense, the final answer is a proper combination of median and average of squares of entities of the sketching vector).

This concludes the proof of Theorem 5.8.

Remark. Linear sketching is a powerful and versatile algorithmic technique that is used frequently in different domains. We already saw an application to streaming algorithms in this chapter and later in the book, we will see further applications to compressed sensing and designing fast algorithms for different problems. The general theme is that linearity of the sketch allows one to separately create a sketch of different parts of the input and then simply add them to obtain a sketch of the combined input as well.