Exercise 1 (A streaming algorithm for the second moment of the frequencies). We are given a stream of numbers \(x_1, \ldots, x_n \in \{0, \ldots, m - 1\}\) and we want to compute the sum of the squares of the frequencies of each values 0 to \(m - 1\) in this stream: if \(f_a(x) = \#\{i : x_i = a\}\), we want to compute \(F_2(x) = \sum_{a=0}^{m-1} f_a(x)^2\).

Take a random function \(h : \{1, \ldots, m\} \rightarrow \{-1, 1\}\), i.e.: for all \(a\), \(h(a)\) is chosen independently and uniformly at random in \(\{-1, 1\}\). And do the following:

Algorithm 1 Second frequency moment random algorithm

Pick a hash function \(h : \{0, \ldots, m - 1\} \rightarrow \{-1, 1\}\) uniformly at random
Compute \(Z = h(x_1) + \cdots + h(x_n)\) while reading the stream
Output \(Z^2\)

\[\begin{align*}
\text{Question 1.1} & \quad \text{Show that } \mathbb{E}[Z^2] = F_2(x) \text{ where the expectation is taken over all the possible values for } h. \quad \text{\(\triangleright\) Hint. For } a \neq b, \text{ show that } \mathbb{E}_h[h(a)h(b)] = 0 \text{ and } \mathbb{E}_h[h(a)^2] = 1. \\
\text{Answer: } \triangleright & \quad \text{First remark that, } \mathbb{E}[h(a)^2] = \mathbb{E}[1] = 1 \text{ and } \mathbb{E}[h(a)] = \frac{1}{2} \times -1 + \frac{1}{2} \times 1 = 0. \\
& \quad \text{Now, if } a \neq b, \text{ then } h(a) \text{ and } h(b) \text{ are independent and } \mathbb{E}[h(a)h(b)] = \mathbb{E}[h(a)] \mathbb{E}[h(b)] = 0. \text{ It follows:}
\end{align*}\]

\[
\mathbb{E}[Z^2] = \mathbb{E}[(h(x_1) + \cdots h(x_n))^2] = \mathbb{E} \left[ \left( \sum_{a=0}^{m-1} f_a(x)h(a) \right)^2 \right] = \sum_{a=0}^{m-1} f_a(x)^2 \mathbb{E}[h(a)^2] + \sum_{a \neq b} f_a(x)f_b(x) \mathbb{E}[h(a)h(b)] = \sum_{a=0}^{m-1} f_a(x)^2 = F_2(x). 
\]

\[\triangleright\]

\textbf{Question 1.2) Show that } \mathbb{V}ar(Z^2) = \mathbb{E}[Z^4] - \mathbb{E}[Z^2]^2 = 2 \sum_{a \neq b} f_a(x)^2 f_b(x)^2 \leq 2F_2(x)^2. \\
\text{Answer: } \triangleright & \quad \text{As before, note that if } b, c, d \text{ are all different from } a, \text{ by independence of } h(a) \text{ from } h(b), h(c) \text{ and } h(d), \text{ we have: } \mathbb{E}[h(a)^2h(b)] = \mathbb{E}[h(a)h(b)] = 0 \text{ and}
\]
\[ \mathbb{E}[h(a)h(b)h(c)h(d)] = \mathbb{E}[h(a)] \mathbb{E}[h(b)h(c)h(d)] = 0. \]

It follows that:

\[
\mathbb{E}[Z^4] = \mathbb{E} \left[ \left( \sum_{a=0}^{m-1} f_a(x)h(a) \right)^4 \right] \\
= \sum_{a=0}^{m-1} f_a(x)^4 \mathbb{E}[h(a)^4] \\
+ 4 \sum_{a=0}^{m-1} f_a(x)^3 f_b(x) \mathbb{E}[h(a)^3 h(b)] \\
+ \frac{1}{2} \left( \sum_{a=0}^{m-1} f_a(x)^2 f_b(x)^2 \mathbb{E}[h(a)^2 h(b)] \right) + 2 \sum_{a=0}^{m-1} f_a(x)^2 f_b(x)^2 \mathbb{E}[h(a)^2 h(b) h(c)] \\
+ \sum_{a=0}^{m-1} f_a(x) f_b(x) f_c(x) f_d(x) \mathbb{E}[h(a) h(b) h(c) f(d)] \\
= \sum_{a=0}^{m-1} f_a(x)^4 + 3 \sum_{a,b \neq b} f_a(x)^2 f_b(x)^2.
\]

Thus, \( \text{Var}[Z^2] = \mathbb{E}[Z^4] - \mathbb{E}[Z^2] = \sum_{a=0}^{m-1} f_a(x)^4 + 3 \sum_{a,b \neq b} f_a(x)^2 f_b(x)^2 - \left( \sum_{a=0}^{m-1} f_a(x)^2 \right)^2 \]
\[
= 2 \sum_{a,b \neq b} f_a(x)^2 f_b(x)^2 \leq 2 \left( \sum_{a=0}^{m-1} f_a(x)^2 \right)^2 = 2F_2(x)^2.
\]

\(<\)

Remark that this algorithm requires a lot of memory to store \( h; O(m \log m) \) bits, almost as much as counting the frequencies independently (\( O(m \log n) \) bits). But remark that we only need the values of \( h \) to be \( 4 \)-wise independent to obtain the results above. Let us thus use the following construction for \( h \) that will require much less memory.

Consider the field \( \mathbb{F}_{2^k} \) where \( k = \lceil \log_2 m \rceil \) such that \( 2^{k-1} < m \leq 2^k \). Let us identify the elements of \( \mathbb{F}_{2^k} \) as a string of \( k \) bits and as numbers from 0 to \( 2^k - 1 \) as well. Let \( \pi: \mathbb{F}_{2^k} \to \{-1, 1\} \) be the function that associates to any number \( a \in \mathbb{F}_{2^k} \) the value \(-1\) if the first bit of \( a \) is 0 and the value \(+1\) otherwise.

For all 4-tuple \((u, v, w, t) \in (\mathbb{F}_{2^k})^4\), let \( P_{uvwt} : \mathbb{F}_{2^k} \to \mathbb{F}_{2^k} \) be the polynomial:

\[
P_{uvwt}(a) = ua^3 + va^2 + wa + t,
\]

and set \( h_{uvwt}(a) = \pi(P_{uvwt}(a)) \).

**Question 1.3)** Show that if \( u, v, w, t \) are chosen independently and uniformly at random in \( \mathbb{F}_{2^k} \), then for all fixed distinct values \( a, b, c, d \in \mathbb{F}_{2^k} \), the random 4-tuple \((P_{uvwt}(a), P_{uvwt}(b), P_{uvwt}(c), P_{uvwt}(d)) \) is uniform in \((\mathbb{F}_{2^k})^4\).
Conclude that when for the event least \( \text{dom} \) \( v \) \( w \) \( t \) \( f \) \( u \) \( v \) \( w \) \( t \) \( f \) \( f \), indeed, the solution \( (u, v, w, t) \) is unique since the matrix is a van der Mond matrix which is inversible as soon as \( a, b, c \) and \( d \) distinct. It follows that all the values in \( (F_{2k})^4 \) are equally probable for the \( 4 \)-tuple \( F_{\text{uwvt}}(a), F_{\text{uwvt}}(b), F_{\text{uwvt}}(c), F_{\text{uwvt}}(d) \), it is thus uniform. \( \Box \)

**Question 1.4** Conclude that when \( u, v, w, t \) are chosen independently and uniformly at random in \( F_{2k} \), the values \( h_{\text{uwvt}}(0), \ldots, h_{\text{uwvt}}(m - 1) \) are \( 4 \)-wise independent uniform random variables with values in \( \{-1, 1\} \).

**Answer.** Remark that \( \pi \) maps half the elements in \( F_{2k} \) to \(-1\) and the other half to \(1\). Thus, the image by \( \pi \) of a uniform random variable in \( F_{2k} \) is a uniform random variable in \( \{-1, 1\} \). Formally, for all \( (\alpha, \beta, \gamma, \delta) \in \{-1, 1\}^4 \) and distincts \( a, b, c, d \in F_{2k} \),

\[
\Pr_{u,v,w,t} \{ (\pi(F_{\text{uwvt}}(a)), \pi(F_{\text{uwvt}}(b)), \pi(F_{\text{uwvt}}(c)), \pi(F_{\text{uwvt}}(d))) = (\alpha, \beta, \gamma, \delta) \} = \frac{1}{(2k)^4}.
\]

which implies that \( \pi(F_{\text{uwvt}}(0)), \ldots, \pi(F_{\text{uwvt}}(m - 1)) \) are \( 4 \)-wise independent uniform random variables in \( \{-1, 1\} \). \( \Box \)

**Question 1.5** Conclude that there is a \( (\varepsilon, \delta) \)-estimator computing \( F_2(x) \) using \( O(\log m + \log n) \) bits of memory. Describe it and explain the bound on the memory needed as a function of \( \delta \) and \( \varepsilon \).

**Answer.** Consider the following algorithm and let us prove that it is a \( (\varepsilon, \delta) \)-estimator:

Recall that \( \text{Var}(\mu_i) = \text{Var}(Z)/B \leq 2F_2(x)^2/B \). By Chebychev inequality, for all \( i = 1, \ldots, A \),

\[
\Pr[|\mu_i - F_2(x)| \geq \varepsilon F_2(x)] \leq \frac{\text{Var}(\mu_i)}{\varepsilon^2 F_2(x)^2} \leq \frac{2}{B \varepsilon^2} \leq \frac{1}{4}.
\]

Furthermore, if the median of the values \( \mu_1, \ldots, \mu_A \) lies outside \( (1 \pm \varepsilon)F_2(x) \), then at least \( A/2 \) of the values lie outside as well. Then, if \( Y_1 \) denotes the indicator random variable for the event \( \mu_i \not\in (1 \pm \varepsilon)F_2(x) \) (note that \( E[Y_1] \leq 1/4 \)), then by Hoeffding inequality,

\[
\Pr[|\text{output} - F_2(x)| \geq \varepsilon F_2(x)] \leq \Pr[Y_1 + \cdots + Y_A \geq A/2] \\
\leq \Pr[Y_1 + \cdots + Y_A - E[Y_1 + \cdots + Y_A] \geq A/4] \\
\leq \exp \left( -\frac{2(A/4)^2}{A} \right) \leq \delta.
\]
Algorithm 2 Memory efficient second frequency moment \((\varepsilon, \delta)\)-estimator

Let \(k = \lceil \log_2 m \rceil\), \(A = \lceil 8 \ln(1/\delta) \rceil\) and \(B = \lceil 8/\varepsilon^2 \rceil\)

for \(i = 1..A\) and \(j = 1..B\)

- Pick \(u_{ij}, v_{ij}, w_{ij}, t_{ij}\) independently and uniformly at random in the field \(\mathbb{F}_{2^k}\)
- Let \(h_{ij}\) be the hash function: \(h_{ij}(a) = \pi(u_{ij}a^3 + v_{ij}a^2 + w_{ij}a + t_{ij})\)
- Compute \(Z_{ij} = h_{ij}(x_1) + \cdots + h_{ij}(x_n)\) for all \(i = 1..A\) and \(j = 1..B\) simultaneously while reading the stream

for \(i = 1..A\)

- Compute the average \(\mu_i = \frac{(Z_{i1})^2 + \cdots + (Z_{iB})^2}{B}\)

return the median of the values \(\mu_1, \ldots, \mu_A\)

Now, the algorithm is memory efficient since it uses: \(4AB\) variables of \(k\) bits each (the \(u_{ij}, v_{ij}, w_{ij}, t_{ij}\)) and \(AB + A\) variables of at most \(2 \log n\) bits (the \(Z_{ij}\) and \(\mu_i\)).

The total number of bits of memory used by the \((\varepsilon, \delta)\)-estimator for \(F_2(x)\) is thus: \(O\left(\frac{\ln(1/\delta)}{\varepsilon^2}(\log m + \log n)\right)\).