Fine-Grained Cryptanalysis: Tight Conditional Bounds for Dense k-SUM and k-XOR

Itai Dinur*
Ben-Gurion University
dinuri@cs.bgu.ac.il

Nathan Keller‡
Bar-Ilan University
Nathan.Keller@biu.ac.il

Ohad Klein‡
Bar-Ilan University
ohadkel@gmail.com

Abstract

An average-case variant of the k-SUM conjecture asserts that finding k numbers that sum to 0 in a list of r random numbers, each of the order r^k, cannot be done in much less than $r^{\lceil k/2 \rceil}$ time. On the other hand, in the *dense regime* of parameters, where the list contains more numbers and many solutions exist, the complexity of finding one of them can be significantly improved by Wagner's k-tree algorithm. Such algorithms for k-SUM in the dense regime have many applications, notably in cryptanalysis.

In this paper, assuming the average-case k-SUM conjecture, we prove that known algorithms are essentially optimal for $k = 3, 4, 5$. For $k > 5$, we prove the optimality of the k-tree algorithm for a limited range of parameters. We also prove similar results for k-XOR, where the sum is replaced with exclusive or.

Our results are obtained by a self-reduction that, given an instance of k-SUM which has a few solutions, produces from it many instances in the dense regime. We solve each of these instances using the dense k-SUM oracle, and hope that a solution to a dense instance also solves the original problem. We deal with potentially malicious oracles (that repeatedly output correlated useless solutions) by an obfuscation process that adds noise to the dense instances. Using discrete Fourier analysis, we show that the obfuscation eliminates correlations among the oracle's solutions, even though its inputs are highly correlated.

*The first author was supported by the Israel Science Foundation (grants no. 573/16 and 1903/20).

‡This research was supported by the European Research Council under the ERC starting grant agreement no. 757731 (LightCrypt) and by the BIU Center for Research in Applied Cryptography and Cyber Security in conjunction with the Israel National Cyber Bureau in the Prime Ministers Office. In addition, the second author was supported by the Israel Science Foundation (grants no. 1612/17 and 2669/21).

‡The third author was supported by the Clore Scholarship Programme.

1 Introduction

1.1 Background

The k-SUM problem. For parameters $k = O(1), r$, the classical worst-case search variant of the k-SUM problem asks: Given a list of r numbers, find (with high probability) k of them whose SUM is zero, assuming such numbers exist.[1] Given that a solution exists, a simple sort-and-match (or meet-in-the-middle) algorithm finds it in time $T = \tilde{O}(r^{\lceil k/2 \rceil})$ (the notation $\tilde{O}$ hides logarithmic factors in r), and the well-known k-SUM conjecture (that generalizes the 3-SUM conjecture [17]), states that no algorithm can do substantially better in standard computational models (such as the word RAM model).

In this paper, we consider average-case variants of the k-SUM problem.

Definition 1.1 (Average-case k-SUM problem)**.** *In the (k, N, r)-SUM problem, the input consists of r elements $z_1, \ldots, z_r$, each of them chosen independently and uniformly at random from $\{-N, \ldots, N\}$. The goal is to find a k-tuple (an ordered set of distinct indices) $K = \{i_1, \ldots, i_k\}$, such that $\sum_{j \in K} z_j = 0$, where the sum is over $\mathbb{Z}$.*

In a *sparse* regime of parameters *only a few solutions exist on average*, i.e., $r^k \approx N$. It is considered folklore that the uniform distribution is a hard distribution for k-SUM under a standard model of computation (see [23] and [33] for a formulation for $k = 3$):

Conjecture 1.2 (Sparse average-case k-SUM conjecture)**.** *Any algorithm that solves the (k, N, r)-SUM problem where $r = N^{1/k}$ with probability $\Omega_k(1)$ has expected running time of at least $T = \Omega_k(r^{\lceil k/2 \rceil - o(1)})$.*

We note that for constant k and $N = \omega(r^k)$, a solution exists with probability $o(1)$, hence the problem cannot be solved with probability $\Omega_k(1)$, regardless of the running time.

In the *dense* setting where *many solutions exist* on average (namely, when $r^k \gg N$), one can do much better.

For $k = 3$, there is a simple algorithm that filters the input by keeping only numbers that are smaller than some threshold in absolute value. This gives a smaller sparse instance to which the standard algorithm is applied to solve the problem in time $T = \tilde{O}(N/r)$ (for $N^{1/3} \leq r \leq N^{1/2}$). For $k > 3$, improvements are obtained via the celebrated Wagner's *k-tree algorithm* [35] discussed below.

The k-XOR problem. The discussion above equally applies to the average-case k-XOR problem.

Definition 1.3 (Average-case k-XOR problem)**.** *In the $(k, 2^n, r)$-XOR problem, the input consists of r vectors $z_1, \ldots, z_r$, each chosen independently and uniformly at random from $\{0,1\}^n$. The goal is to find a k-tuple, $K = \{i_1, \ldots, i_k\}$, such that $\bigoplus_{j \in K} z_j = 0_n$.*

Similarly to k-SUM, the following conjecture is considered folklore.

Conjecture 1.4 (Sparse average-case k-XOR conjecture)**.** *Any algorithm that solves the $(k, N = 2^n, r)$-XOR problem where $r = N^{1/k}$ with probability $\Omega_k(1)$ has expected running time of at least $T = \Omega_k(r^{\lceil k/2 \rceil - o(1)})$.*

[1] Another variant of the k-SUM problem asks, given k lists of r/k numbers, find k numbers – one from each list – whose SUM is zero. The two problems are equivalent, up to $O_k(1)$ factors.

In the dense 3-XOR problem, the input consists of $r \gg 2^{n/3}$ uniform vectors. Similarly to 3-SUM, a simple filtering algorithm has complexity $T = \tilde{O}(N/r)$ for $N^{1/3} \leq r \leq N^{1/2}$. The dense 3-XOR problem has various applications in cryptography and cryptanalysis [9, 14, 22, 24, 31]. While mild (logarithmic in N) improvements to the simple filtering algorithm are known [22, 24, 31], any substantial (i.e., polynomial in N) improvement would be considered a breakthrough.

Wagner's k-tree algorithm. For $k > 3$, the *k-tree algorithm* of Wagner [35] allows finding a solution to k-XOR in time $T = \tilde{O}(N^{1/(1+\lfloor \log_2 k \rfloor)})$, when r is of similar size. The generalized algorithm of Minder and Sinclair [29] provides a tradeoff between r and T, for all $N^{1/k} \leq r \leq N^{1/(1+\lfloor \log_2 k \rfloor)}$. For the most basic case of $k = 4$, the tradeoff curve is $T = \tilde{O}(N/r^2)$ for $N^{1/4} \leq r \leq N^{1/3}$. As was noted in [35], the algorithm is also applicable to the modular k-SUM problem in $\mathbb{Z}_N$. Similarly, it can be easily modified to work for the average-case variant of k-SUM stated above. For the sake of completeness, we give a high-level overview of the k-tree algorithm and its generalization in Appendix A.

In the 20 years since its introduction, the k-tree algorithm (notably for small k values) has become a central tool in cryptanalysis for solving both dense k-SUM and k-XOR problems (see [22]). Specifically, it is used in breaking hash functions [28, 35], stream ciphers [26], block ciphers [15], signature schemes [7] (where the optimal value of k depends on the amount of available data), etc. Furthermore, it has found multiple applications that are not directly related to cryptanalysis. Notably, the *representation technique* [19] crucially relies on variants of the algorithm for small values of k to find one out of many representations of a solution to a problem. This technique gave rise to breakthrough algorithms for solving subset-sum [19, 30] and related problems such as decoding binary linear codes [6].

Finally, the k-tree algorithm is closely related to the Blum-Kalai-Wasserman (BKW) algorithm for solving the LPN (learning parity with noise) problem [8] and its extensions, such as Lyubashevsky's algorithm [27] (although these use $k = \omega(1)$).

In this paper we address the question: **Are the best-known algorithms for dense k-SUM and k-XOR optimal?**

1.2 Our results

We show that in some of the most basic cases, $k = 3, 4, 5$, as well as in other settings, the best known algorithms for k-SUM (resp., k-XOR) in the dense regime are optimal up to logarithmic factors in the input list size, unless the sparse average-case k-SUM (resp., k-XOR) conjecture fails.

Informal statement of the main results. Our main theorem for k-SUM is as follows.

Theorem 1.5 (Conditional dense k-SUM hardness, informal)**.** *Assume that any algorithm that solves $(k, N, N^{1/k})$-SUM with probability $\Omega_k(1)/(\log N)^2$ has expected running time of at least $T = T(N, k)$.*

Then, there is $C = C(k)$ such that for any $0 \leq \epsilon \leq 1/2$, any algorithm that solves $(k, N', (N')^{(1+\epsilon)/k})$-SUM with probability $1/2$ has expected running time of at least $C \cdot T((N')^{1+\epsilon}, k) \cdot (N')^{-\epsilon}$.

Remark 1.6. We make several related remarks about the theorem.

- While the success probability in the hardness assumption of Theorem 1.5 is slightly smaller than the constant success probability in Conjecture 1.2, disproving this stronger assump-

tion with the same time complexity, would be considered a breakthrough (e.g., for cryptanalytic applications).

- It is possible to amplify the success probability in the hardness assumption from $\Omega_k(1)/(\log N)^2$ to $1/2$ (or any constant), at the cost of increasing the number of input elements r and the expected running time of the algorithm by a factor of $(\log N)^2 \cdot O_k(1)$. The amplification is obtained by partitioning the elements into disjoint groups of size $N^{1/k}$, and running the algorithm for $(k, N, N^{1/k})$-SUM on each group independently.
- The input size of $N^{1/k}$ in the conditional hardness assumption can be adjusted to $C_1 \cdot N^{1/k}$ for any constant $C_1 > 0$. This only requires adjusting the hidden constant behind the success probability $\Omega_k(1)/(\log N)^2$.

We obtain similar results for k-XOR (with constant success probability in the hardness assumption, as in Conjecture 1.4). Our main theorem for k-XOR is as follows.

Theorem 1.7 (Conditional dense k-XOR hardness, informal)**.** *Assume that any algorithm that solves $(k, N, N^{1/k})$-XOR with probability $\Omega_k(1)$ has expected running time of at least $T = T(N, k)$.*

Then, there is $C = C(k)$ such that for any $0 \le \epsilon \le 1/2$, any algorithm that solves $(k, N', (N')^{(1+\epsilon)/k})$-XOR with probability $1/2$ has expected running time of at least $C \cdot T((N')^{1+\epsilon}, k) \cdot (N')^{-\epsilon}$.

Discussion. For $k = 3$, by a slightly stronger variant of the sparse average-case k-SUM conjecture (with the success probability in the hardness assumption adjusted to $\Omega_k(1)/(\log N)^2$), we set $N' = N$ and $T(N, 3) = \Omega(N^{2/3-o(1)})$, and deduce that any algorithm for $(3, N, r)$-SUM with $r \approx N^{(1+\epsilon)/3}$ that succeeds with probability $1/2$ has expected running time of $\Omega(N^{(2/3-o(1))(1+\epsilon)-\epsilon}) = \Omega(N^{2/3-o(1)} \cdot N^{-\epsilon/3})$. We conclude that the tradeoff $T = \tilde{O}(N/r)$ for $N^{1/3} \le r \le N^{1/2}$ obtained by the simple (filtering) sort-and-match algorithm is essentially optimal. Similar tightness holds for the 3-XOR problem.

By a similar calculation for $k = 4$, the (extended) k-tree algorithm is essentially optimal, under the sparse average-case k-SUM (resp., k-XOR) conjecture. Theorems 1.5 and 1.7 yield optimality of the best known algorithm also for $k = 5$ and for part of the range for other values of k. In particular, for even values of k, setting $N' = N$ and $T(N, k) = \Omega_k(N^{1/2-o(1)})$, we conclude that for any algorithm for (k, N, r)-SUM (resp., XOR) with $r = N^{(1/k)\cdot(1+\epsilon)}$ that succeeds with probability $1/2$ has expected running time of $\Omega_k(N^{1/2-\epsilon/2-o(1)})$. This essentially matches the extended k-tree algorithm for k values divisible by 4, in the range $0 \le \epsilon \le 1/3$.

We further note that the loss in the reduction provided by the above theorems is almost *linear*, i.e., a $O_k(1)$ factor for k-XOR and logarithmic in N for k-SUM. This means that for $k = 3, 4, 5$ any improvement of known algorithms, *even by a sufficiently large logarithmic factor in N*, can be leveraged through the theorem to obtain a similar improvement in the algorithms for the sparse average-case k-SUM (resp., k-XOR) problem.

1.3 Our methods

In the following description, we focus on k-XOR, as technical details are simpler for this problem. We then summarize the main different ingredients for k-SUM.

The reduction for k-XOR. We achieve our result by a reduction from the average-case k-XOR problem in the sparse regime, to the average-case k-XOR problem in the dense regime. The basic observation is that we can generate a dense instance with r input vectors from a sparse instance with r input vectors by truncating the n-bit input vectors to obtain shorter m-bit input vectors for $m < n$. This increases the effective input list size (relative to the vector length) and the number of solutions.

We rewrite our main result in a more convenient form based on this observation by a change of variables for the dense regime:

Theorem 1.8 (Conditional dense k-XOR hardness, informal, reformulated)**.** *Assume that any algorithm that solves $(k, N = 2^n, N^{1/k} = 2^{n/k})$-XOR with probability $\Omega_k(1)$ has expected running time of at least $T = T(N, k)$.*

Then, there is $C = C(k)$ such that for any $n/2 \leq m \leq n$, any algorithm that solves $(k, M = 2^m, 2^{n/k})$-XOR with probability $1/2$ has expected running time of at least $C \cdot T(2^m, k) \cdot 2^{m-n}$.

Observe that this is a indeed a reformulation of Theorem 1.7, obtained by setting $n' = m$ and $n = (1 + \epsilon)m$ (hence, $(N')^{-\epsilon} = 2^{-\epsilon n'} = 2^{-\epsilon m} = 2^{m-n}$).

The basic idea of the reduction is to take the sparse input of r uniform n-bit vectors, generate from it *many dense k-XOR inputs* of r uniform m-bit vectors, and solve each one using a black-box algorithm B. We then check whether each solution yields a k-XOR solution for the original n-bit vectors. If we could make the input sets of m-bit vectors *completely independent*, then $O(2^{n-m})$ calls to B would be sufficient (as the probability that a solution for m-bit vectors corresponds to a solution for n-bit vectors is 2^{m-n}), and the assertion of Theorem 1.8 would be achieved. In fact, it is easy to show that pairwise independence suffices. However, one cannot make these input sets pairwise independent and maintain their relation to the original n-bit vectors at the same time (unless $m \leq n/2$, which is not useful in our case). Thus, even though it is called about 2^{n-m} times, B could potentially repeatedly output solutions to dense k-XOR instances that reside in a small set which does not contain any solution to the sparse instance.

We overcome this obstacle by an *obfuscation* process, which applies to the input vectors two different types of noise consecutively, and allows us to achieve almost pairwise independence of *B's outputs* even though its inputs are *significantly correlated*. In the reduction, we are given a sequence $z_1, \ldots, z_r \in \{0,1\}^n$ for which we wish to solve the k-XOR problem. We apply the following procedure.

1. **Draw** a uniformly random matrix $T \in \{0,1\}^{m \times n}$ of full rank m, and a uniformly random permutation P on r elements.
2. **Let** $x_i = T(z_{P(i)})$ for all $i \in [r]$.
3. **Feed** B with $x_1, \ldots, x_r$. In case it outputs a k-tuple K with $\bigoplus_{i \in K} x_i = 0_m$, **test** whether $\mathcal{K} = P(K)$ satisfies $\bigoplus_{j \in \mathcal{K}} z_j = 0_n$, and **if** so – **output** the k-tuple $\mathcal{K}$. **Otherwise, repeat**.

We prove that after 2^{n-m} trials, with probability of $\Omega_k(1)$, the process outputs a solution of the sparse k-XOR problem. We use *discrete Fourier analysis* in order to bound the correlation between B's outputs.

Remark 1.9. If we hash down a hard sparse instance to get a dense instance, then clearly any procedure that enumerates all solutions to the dense instance (if there aren't too many) is hard as well. However, our reduction does not follow this standard paradigm, as the oracle for

the dense instance can only produce a single solution. Thus, we hash the sparse instance down in many different ways and repeatedly invoke the oracle in order to force it to produce many different potential solutions to the sparse instance.

The reduction for k-SUM. The reduction for k-SUM follows the same general strategy (with modified obfuscation), but its proof is more involved. In particular, in addition to discrete Fourier-analytic techniques, it uses tools from Littlewood-Offord theory [16].

Concisely, the reason for further complexity in the k-SUM case, is that there are only a few group-homomorphisms from $\mathbb{Z}_N$ to $\mathbb{Z}_M$ (that may be used to obfuscate the input), while group-homomorphisms from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ are abundant. We now elaborate on this point.

Both k-XOR and k-SUM reductions employ an obfuscation procedure to the input of the sparse problem. For k-XOR we hash r elements of $\{0,1\}^n$ to r elements of $\{0,1\}^m$ ($m < n$), while for k-SUM we hash r elements of $\mathbb{Z}_N$ to r elements of $\mathbb{Z}_M$. Since we are required to pull-back linear information from the output of the process (i.e. a solution to a dense problem), to its input (i.e., solve the sparse problem), it is important that the obfuscation would be linear. However, the class of *obfuscation* functions must be rich enough to mask additional information that can be exploited by the dense algorithm to output correlated solutions. In the k-XOR case, we make use of a random, rank-m, linear map from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$. There are $2^{\Theta(mn)}$ options to choose this linear map. In the k-SUM problem, if we would insist on a (surjective) linear map from $\mathbb{Z}_N$ to $\mathbb{Z}_M$, there would be at most M such functions, which gives insufficient obfuscation (in particular, any $x \in \mathbb{Z}_N$ that is divisible by M, would be mapped to 0). Hence, we must settle for a *somewhat-linear* obfuscation. We choose a function $\phi\colon \mathbb{Z}_N \to \mathbb{Z}_M$ of the form

$$\phi(x) = \gamma\lfloor \alpha \cdot x \cdot M/N \rceil \bmod M,$$

where $\alpha \in \mathbb{Z}_N^*$, $\gamma \in \mathbb{Z}_M^*$ are chosen uniformly at random and $\lfloor\cdot\rceil$ denotes rounding to the nearest integer. This new class of non-strictly-linear obfuscation functions still enables us to pull back a solution from a dense problem to the sparse problem. Moreover, it turns out that this class is rich enough to allow obfuscation. However, proving this latter point is more involved, since the non-linearity of the obfuscation makes the use of Fourier-analytic tools more complex.

1.4 Related work

Our result is naturally related to three lines of work.

Algorithms for dense k-SUM and k-XOR. The first line of work is the quest for designing better algorithms for *generalized birthday problems*, where the goal is to find a single k-SUM or k-XOR solution out of many. The systematic analysis of this problem was initiated by Wagner [35] in his k-tree algorithm,[2] and has led to numerous refined algorithms and applications thereof [13, 22, 24, 29, 31].

In this respect, we show that – perhaps surprisingly – the best-known algorithms are *essentially optimal* for $k = 3, 4, 5$, unless a standard conjecture in computational complexity fails. Moreover, under similar conjectures, the best known algorithms for any $k > 5$ are essentially optimal for some range of parameters.

Fine-grained complexity. The second related line of work deals with complexity reductions to the k-SUM problem and its variants, that have become a flourishing field of research in the last decade, as part of the *fine-grained complexity* research direction [34]. Reductions to

[2] We note that variants of the k-tree algorithm were presented earlier, starting with Camion and Patarin [11], as is stipulated in [35, full version].

k-SUM or to k-XOR were shown for problems in computational geometry [4, 17], dynamic algorithms [1, 32], graph algorithms [21, 36], pattern matching [2] and more. In the last few years, such reductions were shown also for several cryptographic problems [5, 18, 23], as part of the emerging *fine-grained cryptography* research area [12].

Our work provides yet another reduction for a cryptography-related problem, however the context of reduction in our case is somewhat different. While previous works prove security of (mainly theoretical) *classes of cryptographic primitives*, based on well-founded hardness assumptions, our work shows *a bound on the possible effectiveness of an important class of cryptanalytic algorithms* that are widely used for breaking cryptosystems. Thus, our results also have more practical significance.

Asymptotic hardness for dense k-SUM. Finally, a very recent work related to ours is the paper [10] by Brakerski, Stephens-Davidowitz, and Vaikuntanathan, which proves asymptotic optimality of the k-tree algorithm for average-case k-SUM by reducing it from *worst-case complexity of lattice problems*. While [10] is related to our work, the results of the two papers are complementary due to several important differences which we summarize below.

First, [10] yields asymptotic bounds as $k \to \infty$, while our work concentrates on small values of k, which are those that usually appear in applications of the k-tree algorithm to certain types of cryptanalytic problems. In this respect, our result resolves an open problem stated in [10, Sec. 1.3] (although our reduction is not from a worst-case problem). Second, while the bounds of [10] are tight only up to a constant multiplicative factor *in the exponent* (and are less relevant for small values of k), our reduction is tight up to constant (or logarithmic) factors for certain parameter ranges such as $k = 3, 4, 5$. As the reduction of [10] does not give meaningful results for small values of k, it is not clear how to combine our techniques with the ones of [10] to obtain a reduction from a worst-case problem.

Third, while the reduction of [10] is from a different problem, involving lattices, our reduction is from a standard conjecture in the sparse regime to the dense regime of the same problem (i.e., a *self-reduction*). Fourth, the reduction technique is different. Both papers aim at obtaining 'sufficiently different' variants of the same input sample z. However, in [10] these are obtained by *re-randomization* (repeatedly generating almost independent inputs to the k-SUM algorithm from the same list of vectors) and their independence is proved via the *leftover hash lemma* [20]. On the other hand, our variants are obtained via the *obfuscation* method described above, in which the inputs to the k-SUM algorithm are highly correlated, and we prove the low correlation of the algorithm's outputs via *discrete Fourier analytic* methods. We note that it is possible to use the leftover hash lemma to analyze our reduction as well, but it seems to give very loose results. Finally, in addition to k-SUM, we also obtain conditional hardness results for the k-XOR problem.

1.5 Additional application and open problems

Additional application. The security proof of the hash construction $T5$, recently proposed by Dodis et al. [14], is based on dense 3-XOR and 4-XOR assumptions. Our results directly imply that the security of the construction can be based on standard sparse 3-XOR and 4-XOR assumptions instead of non-standard dense ones. In this sense, our work (in combination with the original security proof of [14]) allows to prove security for a cryptosystem, similarly to [5, 18, 23], yet this proof is obtained for a practical cryptosystem.

Open problems. The main remaining open problem is to improve our lower bound in the setting of a large k and a large number of solutions, or alternatively, to improve the k-tree

algorithm in this range.

The structure of the paper. Next, we summarize our notations and conventions. In Section 3 we prove our main result for k-XOR, while in Section 4 we prove our main result for k-SUM.

2 Notations and Conventions

In this section we introduce notations and conventions that will be used throughout the paper.

Notations.

- $x \sim S$ means that x is a random variable uniformly distributed in the set S.
- We interchangeably write $\{0,1\}$ and $\mathbb{F}_2$, where $0 = 1 \oplus 1$.
- When $z \in \mathbb{F}_2^{r \times n}$, z_{ij} denotes the (i,j)'th entry of z, and $z_i := (z_{i1}, \ldots, z_{in})$.
- For $x \in \mathbb{F}_2^{r \times m}$ and a permutation $P \in S_r$, we denote by $P(x)$ the value $y \in \mathbb{F}_2^{r \times m}$ satisfying $y_{P(i)} = x_i$ for all i.
- For a linear map $T \colon \mathbb{F}_2^n \to \mathbb{F}_2^m$ and for $z \in \mathbb{F}_2^{r \times n}$, we denote by $T(z)$ the value $x \in \mathbb{F}_2^{r \times m}$ satisfying $x_i = T(z_i)$ for all i.
- $\mathbb{Z}_L$ is the group whose elements are $\{0, 1, \ldots, L-1\}$, and whose operation is addition modulo L.
- $a \,\%\, b$ (or $a \bmod b$) stands for the single element in $(a + b\mathbb{Z}) \cap [0, b)$.
- For a real number u, the rounded value $\lfloor u \rceil$ is the unique integer in $u + (-1/2, 1/2]$.
- For functions $f, g : \mathbb{N} \to \mathbb{R}_+$ and a fixed parameter k, $f = O(g)$ means that $\forall n : f(n) \leq Cg(n)$ for an absolute constant C, $f = \tilde{O}(g)$ means that $\forall n : f(n) \leq C_1 \cdot (\log n)^{C_2} \cdot g(n)$ for absolute constants C_1, C_2, and $f = O_k(g)$ means that $\forall n : f(n) \leq h(k) \cdot g(n)$ for some function $h : \mathbb{N} \to \mathbb{R}_+$.

Conventions.

- **Operations within domains.** Throughout the paper, we consider variables in various domains. For example, when analyzing k-XOR, we consider variables in $\{0,1\}^\ell$ and $\{0,1\}^{r \times \ell}$ for different values of ℓ, r, while for k-SUM, we consider variables in $\mathbb{Z}_L$ and $\mathbb{Z}_L^r$ for different values of L, r. Whenever an operation is applied on two elements of the same domain, the result belongs to the same domain. For example, addition between two elements of $\mathbb{Z}_L$ is always performed modulo L.
- **Names of variables.** In the reductions presented in the paper, we begin with vectors that belong to a 'large' space – $\{0,1\}^{r \times n}$ for k-XOR ($\mathbb{Z}_N^r$ for k-SUM), and use them to define vectors that belong to a 'smaller' space – $\{0,1\}^{r \times m}$ for k-XOR ($\mathbb{Z}_M^r$ for k-SUM). Throughout the paper, vectors that belong to a 'large' space are denoted by z, while vectors that belong to a 'small' space are denoted by x or y. Auxiliary vectors denoted by u or v may belong to an arbitrary domain, which will be explicitly defined.
- **Inner products.** All inner products in Section 3 are between functions $f, g : \{0,1\}^{r \times m} \to \mathbb{R}$ (for a particular choice of m), and consequently, their results belong to $\mathbb{R}$. Most inner products in Section 4 are between vectors in $\mathbb{Z}_p^r$ (for a particular choice of p), and consequently, their results belong to $\mathbb{Z}_p$.

3 Hardness of Dense k-XOR

In this section we prove Theorem 1.7 (or equivalently, Theorem 1.8). The precise formulation of the main theorem is as follows.

Theorem 3.1 (Sparse to dense k-XOR reduction)**.** *Let m, n be integers such that $n/2 \leq m \leq n$. Assume there is an algorithm for $(k, 2^m, 2^{n/k})$-XOR with success probability β and expected running time $\mathcal{T}$. Then, there is an algorithm for $(k, 2^n, 2^{n/k})$-XOR with success probability $\frac{\beta^4}{(16k)^{k+2}}$, and expected running time at most $2^{n-m} \cdot (\mathcal{T} + \tilde{O}_k(2^{n/k}))$.*

We make a few remarks. First, we work in the standard word RAM computational model, where an operation on each vector (of size $O(n)$) takes unit time. However, our results do not change substantially in other standard computational models (e.g., in a model that counts the number of bit operations). Second, when applying the contrapositive of this theorem to prove hardness results for dense k-XOR with $k = O(1)$ and $\beta = 1/2$ to obtain the theorems 1.7 and 1.8, the factor $\tilde{O}_k(2^{n/k})$ in this theorem is bounded by $O_k(\mathcal{T})$ (and hence consumed by the factor $C = C(k)$ of theorems 1.7 and 1.8). Indeed, the input in the $(k, M = 2^m, 2^{n/k})$-XOR problem already contains $2^{n/k}$ vectors. Finally, the loss factor $\frac{\beta^4}{(16k)^{k+2}}$ in the success probability can be significantly improved by a more refined analysis.

The proof of the theorem is based on two reductions: the main sparse to dense k-XOR reduction and the simple (but inefficient) sparse to dense reduction which is required for parameter ranges where the main reduction is not applicable. We summarize these reductions in the two lemmas below and prove the simpler lemma first.

Lemma 3.2 (Main k-XOR reduction)**.** *Let m, n, r satisfy $n/2 \leq m \leq n \leq \log_2 \binom{r}{k}$. Assume there is an algorithm for $(k, 2^m, r)$-XOR, with success probability β and expected running time $\mathcal{T}$. Then, there is an algorithm for $(k, 2^n, r)$-XOR with success probability*

$$\frac{\beta^4}{128} \cdot \left(2^n / \binom{r}{k}\right)^2$$

and expected running time at most $2^{n-m} \cdot (\mathcal{T} + \tilde{O}(r))$.

Lemma 3.3 (Simple reduction)**.** *Let $d > 0$ by an integer. Assume there is an algorithm for $(k, 2^n, d \cdot r)$-XOR, with success probability β and expected running time $\mathcal{T}$. Then, there is an algorithm for $(k, 2^n, r)$-XOR with success probability at least $\frac{\beta}{(2d)^k}$ and expected running time at most $\mathcal{T} + \tilde{O}_k(d \cdot r)$.*

Proof of Lemma 3.3. Let B be an algorithm for $(k, 2^n, d \cdot r)$-XOR. We construct an algorithm for $(k, 2^n, r)$-XOR as follows. Given an instance $z_1, \ldots, z_r$, pick $(d-1) \cdot r$ vectors in $\{0,1\}^n$ uniformly at random and append them the original instance. Then, apply a uniform permutation to the $d \cdot r$ vectors and run B on this instance. If B succeeds to return a k-XOR, and this k-tuple is included in $z_1, \ldots, z_r$, then the algorithm returns it. Otherwise, it fails.

As the input to B is uniformly permuted, the events (1) B succeeds, and (2) the returned k-tuple is included in $z_1, \ldots, z_r$, are independent. By assumption, the probability of the first event is β. Note that we may assume $r \geq 2k$, as otherwise, a trivial algorithm for $(k, 2^n, r)$-XOR that goes over all possible k-tuples of input vectors runs in time $O_k(1) \leq \mathcal{T} + \tilde{O}_k(d \cdot r)$. Given that $r \geq 2k$, the probability of the second event is at least $((r-k)/(d \cdot r - k))^k \geq (2d)^{-k}$. Thus, the algorithm succeeds with probability at least $\beta/(2d)^k$. □

We now prove that Theorem 3.1 follows from the lemmas.

Proof of Theorem 3.1. Our goal is to devise an algorithm for $(k, 2^n, 2^{n/k})$-XOR given an algorithm B for $(k, 2^m, 2^{n/k})$-XOR with success probability β and expected running time $\mathcal{T}$.

We first construct an algorithm for $(k, 2^n, 2^{n/k} \cdot k)$-XOR and then use Lemma 3.3 to sparsify the input.

Clearly, B is also applicable to $(k, 2^m, 2^{n/k} \cdot k)$-XOR with similar success probability and complexity (by ignoring all but the first $2^{n/k}$ input vectors). Denote $r = 2^{n/k} \cdot k$ and note that as $k! > (k/e)^k$ for all k, we have

$$e^{-k}\binom{r}{k} \leq 2^n = (r/k)^k \leq \binom{r}{k}.$$

Therefore, based on algorithm B for $(k, 2^m, 2^{n/k} \cdot k)$-XOR, by Lemma 3.2, there is an algorithm B_1 for $(k, 2^n, 2^{n/k} \cdot k)$-XOR with success probability

$$\beta_1 = \frac{\beta^4}{128} \cdot \left(\frac{2^n}{\binom{r}{k}}\right)^2 \geq \frac{\beta^4}{128e^{2k}}$$

and expected running time at most $2^{n-m} \cdot (\mathcal{T} + \tilde{O}_k(2^{n/k}))$.

Using algorithm B_1 for $(k, 2^n, r = 2^{n/k} \cdot k)$-XOR, Lemma 3.3 (applied with $d = k$) implies that there is an algorithm for $(k, 2^n, 2^{n/k})$-XOR with success probability at least

$$\frac{\beta_1}{(2d)^k} \geq \frac{\beta^4}{(16k)^{k+2}}$$

and expected running time at most $2^{n-m} \cdot (\mathcal{T} + \tilde{O}_k(2^{n/k})) + \tilde{O}_k(k \cdot 2^{n/k}) = 2^{n-m} \cdot (\mathcal{T} + \tilde{O}_k(2^{n/k}))$, as claimed. □

3.1 Overview of the main reduction lemma

The proof of the main reduction lemma (Lemma 3.2) is constructive – namely, we construct an algorithm and show that it satisfies the assertion of the lemma. A natural way to solve the sparse k-XOR problem using an oracle for the dense k-XOR problem, is to **truncate** some of the bits. That is, given r vectors, $z_1, \ldots, z_r \in \{0,1\}^n$, with $2^n \approx r^k$ (so that we expect only $\Theta(1)$ solutions) we may feed the oracle with $x_1, \ldots, x_r \in \{0,1\}^m$, where x_i is obtained from z_i by truncating the last $t := n - m$ bits. In the new problem corresponding to $x_1, \ldots, x_r$ we expect to have $\Theta(2^t)$ (i.e., many) solutions, and hence the oracle is applicable. A k-tuple output by the dense k-XOR oracle has a probability of 2^{-t} to be a solution to the original z-problem (since we truncated exactly t bits, and z is uniformly distributed). Thus, it seems that if we feed the oracle $\Theta(2^t)$ times with truncated inputs, we expect $\Theta(1)$ out of the $\Theta(2^t)$ output k-tuples to solve the original z-problem, and consequently, solving the x-problem is at most 2^t times easier than solving the z-problem.

The **flaw** in this argument is that we cannot expect the oracle to output a newly-forged k-tuple in every application (especially if the oracle is deterministic and is fed with the similar inputs in all applications). Hence, although the expected number of solutions we find is $\Theta(1)$, it might be that we solve the z-problem only with a small probability (e.g., it might be that with a high probability the oracle outputs many identical solutions).

Therefore, we have to **trick** the oracle so that the k-tuples it outputs in the different applications will be **pairwise independent** of each other – almost as if we feed it with a fresh uniformly chosen input every time.

For this purpose, we devise a method that receives r vectors in $\{0,1\}^n$ (denoted $z_1,\ldots,z_r$) and randomly obfuscates them, returning r vectors in $\{0,1\}^m$ (denoted $x_1,\ldots,x_r$). This obfuscation meets two criteria:

1. A solution to the k-XOR x-problem gives rise to a solution to the k-XOR z-problem with good probability, (i.e., $p \approx 2^{-t}$).
2. The obfuscation is powerful enough to disguise the fact all the x's are generated from the same z, so that each application of the oracle **(pairwise) independently** has a chance to solve the z-problem.

As we show below, this obfuscation method guarantees that after applying the oracle sufficiently many times on obfuscated x-problems, with a high probability a solution of the original z-problem will be obtained.

Structure of the proof. First we present the obfuscation algorithm and the lemma which asserts that it indeed satisfies the aforementioned properties. Then we prove the main reduction lemma, assuming the obfuscation lemma. Finally, we prove the obfuscation lemma, which is the most complex part of the conditional k-XOR hardness proof.

3.2 The obfuscation algorithm

Let m,n,r satisfy $n/2 \le m \le n \le \log_2\binom{r}{k}$, and let L be a parameter to be specified below. Let B be an algorithm for $(k,2^m,r)$-XOR. The algorithm A for $(k,2^n,r)$-XOR, which receives as input an r-tuple of n-bit vectors $(z_1,\ldots,z_n) \in \{0,1\}^{r\times n}$, is defined as follows.

Algorithm 3.4.

1. **Repeat** L times:
2. **Draw** a uniformly random full-rank matrix $T \in \mathbb{F}_2^{m\times n}$ (rank $= m$) and a uniformly random permutation $P \in S_r$.
3. **Let** $x_i = T(z_{P(i)})$ for all $i \in [r]$.
4. **Feed** B with $(x_1,\ldots,x_r)$. In case it outputs a k-tuple K with $\bigoplus_{i\in K} x_i = 0_m$, **test** whether $\mathcal{K} = P(K)$ satisfies $\bigoplus_{i\in\mathcal{K}} z_i = 0_n$, and **if** it does – **output** the k-tuple $\mathcal{K}$. **Otherwise, continue**.

Thus, we try to solve A's problem, by considering many subproblems (in which we consider only m-bit vectors, instead of n-bit ones), trying to solve these using B, and in case its output solves A's problem, we output the result. Each of these trials succeeds with some probability, and only one success is required. Repeating this procedure enough times, we are expected to find a solution with reasonable probability – unless failures are correlated. This is where the obfuscation lemma comes into play – it shows the trials are sufficiently independent for A to succeed.

The obfuscation lemma. The heart of the proof is the following lemma:

Lemma 3.5. *Let $(z_1,\ldots,z_r) \in \{0,1\}^{r\times n}$ be chosen uniformly at random. Let $(x_1^{(1)},\ldots,x_r^{(1)}) \in \{0,1\}^{r\times m}$ and $(x_1^{(2)},\ldots,x_r^{(2)}) \in \{0,1\}^{r\times m}$ be obtained from it by the procedure described above (in two out of the L iterations). Let $\mathcal{K}_1,\mathcal{K}_2$ be the two corresponding $\mathcal{K}$'s obtained in the process. Then, assuming $t := n-m \le m \le n \le \log_2\binom{r}{k}$, we have*

$$\Pr[\mathcal{K}_1 = \mathcal{K}_2] \le 2^{2-2t}, \tag{1}$$

where the probability is taken over $z, x^{(1)}, x^{(2)}$, and B's randomness.

Note that cases where at least one of $\mathcal{K}_1, \mathcal{K}_2$ is not obtained (that is, when in at least one of the two iterations, Algorithm B fails to find a solution to the $(k, 2^m, r)$-XOR problem) are not counted as equality between $\mathcal{K}_1$ and $\mathcal{K}_2$.

We note that each of $(x_1^{(1)}, \ldots, x_r^{(1)})$ and $(x_1^{(2)}, \ldots, x_r^{(2)})$ is likely to have about 2^t solutions to k-XOR, but only $O(1)$ of these are common to both and typically correspond to k-XOR solutions for $(z_1, \ldots, z_r)$. Therefore, the lemma essentially asserts that B cannot do much better than output a uniform solution to each x-problem.

3.3 Proof of the main reduction lemma

We prove now that the assertion of Lemma 3.2 follows from Lemma 3.5. The proof is a rather standard probabilistic argument. Afterwards we present the considerably more complex proof of Lemma 3.5.

Proof of Lemma 3.2, assuming Lemma 3.5. Consider a slightly tweaked obfuscation process which has exactly L iterations (and may output multiple solutions). Clearly, the success probability of the tweaked obfuscation process is identical to the original one, and thus we analyze it instead. For any $1 \leq l \leq L$, let $\mathcal{K}_l$ be the k-tuple obtained in the l's iteration ($\mathcal{K}_l$ exists only when B succeeds, i.e. with probability β). Denote by S_l the event that $\mathcal{K}_l$ admits a solution to the $(k, 2^n, r)$-XOR problem. We have $\Pr[S_l] = 2^{m-n}\beta$ for each $l = 1, \ldots, L$, since z is uniformly random, and m out of the n dimensions of $\bigoplus_{i \in \mathcal{K}_l} z_i$ are known to nullify, independently of the other, beforehand-erased, $n - m$ dimensions. (In other words, we know that $\bigoplus_{i \in \mathcal{K}_l} z_i$ belongs to the kernel of a randomly chosen full-rank linear transformation $T : \mathbb{F}_2^n \to \mathbb{F}_2^m$, and hence, $\Pr[\bigoplus_{i \in \mathcal{K}_l} z_i = 0_n] = 2^{m-n}$.)

Define the random variables

$$Z' := \sum_{l=1}^{L} \mathbb{1}\{S_l\} - \sum_{1 \leq l < l' \leq L} \mathbb{1}\{\mathcal{K}_l = \mathcal{K}_{l'}\}, \qquad Z := \max(Z', 0),$$

where $\mathbb{1}\{E\}$ is the indicator of the event E. A simple inclusion-exclusion-like principle shows that Z' lower bounds the number of distinct solutions found for the $(k, 2^n, r)$-XOR problem in (tweaked) Algorithm 3.4. As the number of solutions is non-negative, Z lower bounds it as well. The Paley-Zygmund inequality, applied for the non-negative random variable Z, implies

$$\Pr[Z > 0] \geq \frac{\mathbb{E}[Z]^2}{\mathbb{E}[Z^2]}.$$

Since $\Pr[Z > 0]$ lower bounds the probability that (tweaked) Algorithm 3.4 solves the $(k, 2^n, r)$-XOR problem, our task is reduced to lower bounding $\mathbb{E}[Z]^2$, and upper bounding $\mathbb{E}[Z^2]$. The value of $\mathbb{E}[Z]$ is easily bounded as

$$\mathbb{E}[Z] \geq \mathbb{E}[Z'] = \sum_{l=1}^{L} \Pr[S_l] - \sum_{1 \leq l < l' \leq L} \Pr[\mathcal{K}_l = \mathcal{K}_{l'}].$$

Using $\mathbb{E}[S_l] = 2^{m-n}\beta$ and Lemma 3.5, we obtain $\mathbb{E}[Z] \geq L \cdot 2^{m-n}\beta - \binom{L}{2} 2^{2(1+m-n)}$. We choose $L = \beta \cdot 2^{n-m-2}$, and obtain

$$\mathbb{E}[Z] \geq \beta^2/8.$$

We henceforth upper bound $\mathbb{E}[Z^2]$. Let D be the number of distinct k-tuples $T \subseteq [r]$ with $\bigoplus_{i \in T} z_i = 0_n$ (that is, the number of actual solutions for the $(k, 2^n, r)$-XOR problem in the

set $\{z_1, \ldots, z_r\}$). Since Z is not larger than the number of solutions found in (tweaked) Algorithm 3.4, we have $Z \leq D$, and in particular, $\mathbb{E}[Z^2] \leq \mathbb{E}[D^2]$.

Note that for different k-tuples T, T', the events $\bigoplus_{i \in T} z_i = 0_n$ and $\bigoplus_{i \in T'} z_i = 0_n$ are independent, and each of them has probability 2^{-n}. Hence,

$$\mathbb{E}[D^2] = \binom{r}{k} 2^{-n} + \binom{r}{k} \left(\binom{r}{k} - 1 \right) 2^{-2n} \leq \binom{r}{k}^2 2^{1-2n},$$

where the ultimate inequality holds since $\binom{r}{k} \geq 2^n$ by assumption. Therefore, the algorithm succeeds with probability at least

$$\Pr[Z > 0] \geq \frac{\mathbb{E}[Z]^2}{\mathbb{E}[D^2]} \geq \frac{\beta^4}{128 \binom{r}{k}^2 2^{-2n}}.$$

The running time of Algorithm 3.4 (including the $\widetilde{O}(r)$ additional overhead of each iteration) is

$$L \cdot (\mathcal{T} + \widetilde{O}(r)) \leq 2^{n-m} \cdot (\mathcal{T} + \widetilde{O}(r)).$$

This completes the proof of the lemma. □

Remark 3.6. The proof crucially relies on the strength of the obfuscation – specifically, on the fact that the probability of outputting the same k-tuple in an iteration pair is $O(2^{-2t})$. This is the motivation behind using the obfuscation we propose, as weaker obfuscations, such as **truncating t randomly chosen bits**, have oracles that output the same k-tuple in an iteration pair with probability much higher than 2^{-2t}, even if we apply an invertible linear transformation after truncation.

3.4 Proof of the obfuscation lemma

In this section we prove Lemma 3.5. We start by introducing a distribution that models two independent outputs of the obfuscation process, and restate the obfuscation lemma.

Definition 3.7. *We say that a pair of random variables $(x^{(1)}, x^{(2)})$, each taking values in $\mathbb{F}_2^{r \times m}$, has an (m, r, t)-distribution, if there exist random variables z, $T^{(j)}$ for $j \in \{1, 2\}$ where:*

1. *z, $T^{(1)}$, $T^{(2)}$, are independent random variables.*
2. *$z \sim \mathbb{F}_2^{r \times (m+t)}$ is uniformly distributed.*
3. *$T^{(j)} \colon \mathbb{F}_2^{m+t} \to \mathbb{F}_2^m$ is a uniformly random, full-rank (i.e., rank $= m$), linear transformation.*
4. *$x_i^{(j)} = T^{(j)}(z_i)$.*

Theorem 3.8. *Let B be an algorithm that receives as input a list of r vectors, each of length m bits, and outputs the indices of $k > 0$ vectors among them whose XOR is 0_m (or a failure string). If (x, y) has an (m, r, t)-distribution, and $P, Q \sim S_r$ are two uniformly random and independent permutations, then*

$$\Pr[P^{-1}(B(P(x))) = Q^{-1}(B(Q(y)))] \leq 2^{-2t} + \frac{2^{m-t}}{\binom{r}{k}} + 2^{-t+1-m}, \tag{2}$$

where the probability is taken over B's randomness, x, y and P, Q (The event on the left hand side is contained in the event that both executions $B(P(x))$, $B(Q(y))$ succeed).

Notice that Lemma 3.5 immediately follows from Theorem 3.8 (compare (1) with (2)).

3.4.1 Proof outline

The proof of Theorem 3.8 uses techniques from *discrete Fourier analysis* and consists of several steps.

1. **Transformation to real-valued functions.** We show that instead of analyzing the obfuscation on a tuple-valued function, it is sufficient to analyze its action on the simpler class of real-valued functions. We utilize the fact that our obfuscation randomly permutes the input vectors, so that any oracle $B\colon \{0,1\}^{r\times m} \to \binom{[r]}{k}$ must, informally, treat all candidate output k-tuples in the same way. Hence it suffices to analyze the modified, real-valued, oracle $B'\colon \{0,1\}^{r\times m} \to [0,1]$ which indicates the probability that B outputs the specific k-tuple $K := \{1,\ldots,k\}$ when applied on its input. Specifically, our task is reduced to showing that

$$\mathbb{E}[B'(y)B'(y')] \leq O_k(2^{-2t}/r^k), \tag{3}$$

   where y, y' are two independent obfuscations of a common, random, $z \in \{0,1\}^{r\times n}$.

2. **Bounding the correlation using discrete Fourier analysis.** In order to prove (3), we consider the Fourier expansion of B', namely,

$$B' = \sum \widehat{B'}(S)\chi_S, \qquad \text{where} \qquad \chi_S(v) = (-1)^{\sum_{i\in S} v_i}.$$

   We divide the Fourier expansion into two parts – the *Cartesian* part

$$(B')^C = \sum_{\{S=U\times V : U\subseteq[r], V\subseteq[m]\}} \widehat{B'}(S)\chi_S,$$

   and the non-Cartesian part $(B')^{\perp} = B' - (B')^C$ (which is orthogonal to $(B')^C$). Informally, the contribution of the Cartesian part to the correlation corresponds to the information on *aligned XORs* of variables (such as $(z_{1,2} \oplus z_{1,3}) \oplus (z_{4,2} \oplus z_{4,3})$) preserved between the two obfuscations, while the contribution of the non-Cartesian part carries the rest of the information. Then we handle each part of the correlation separately:

   (a) *Obfuscation hides everything but aligned XORs.* We show that for any function B', the obfuscation reduces the contribution to the left hand side of (3), associated with the non-Cartesian part, to at most 2^{-2t}. This argument depends only on the obfuscation, and does not rely on the specific problem we try to solve.

   (b) *Aligned XORs do not reveal much.* We show that in the case of the k-XOR problem, the contribution of the Cartesian part is also small. Here we use the specific structure of the problem – specifically, the set $\{x : B'(x) > 0\}$ being small and admitting a nice algebraic structure (namely, $B'(x) = 0$ whenever $\bigoplus_{i=1}^{k} x_i \neq 0_m$).

   Combination of the two bounds completes the proof.

3.4.2 Transformation to real-valued functions

Lemma 3.9. *Let $B\colon \{0,1\}^{r\times m} \to \binom{[r]}{k}$ be an algorithm that outputs either a k-tuple R with $\bigoplus_{i\in R} x_i = 0_m$, or a failure string. Let $K := \{1,\ldots,k\}$ and define $B'\colon \{0,1\}^{r\times m} \to [0,1]$ by*

$$B'(x) = \mathop{\mathbb{E}}_{\substack{P\sim S_r \\ \bar{T}\sim GL_m(\mathbb{F}_2)}} \left[\mathbb{1}\{B(\bar{T}(P(x))) = P(K)\}\right], \tag{4}$$

where $P \sim S_r$ is a uniformly random permutation, and $\bar{T} \sim GL_m(\mathbb{F}_2)$ is a uniformly random invertible linear map. The expectation in (4) is taken also over B's randomness. Then,

$$\bigoplus_{i \in K} x_i \neq 0 \implies B'(x) = 0, \tag{5}$$

$$\mathop{\mathbb{E}}_{x}[B'(x)] \leq 1/\binom{r}{k}, \tag{6}$$

and if (x, y) has an (m, r, t)-distribution and $P, Q \sim S_r$ are independent, then

$$\Pr[P^{-1}(B(P(x))) = Q^{-1}(B(Q(y)))] = \binom{r}{k} \mathop{\mathbb{E}}_{x,y}[B'(x)B'(y)]. \tag{7}$$

Remark 3.10. We note that while the obfuscation algorithm uses a full-rank shrinking transformation T from $\mathbb{F}_2^{m+t}$ to $\mathbb{F}_2^m$, this transformation does not appear explicitly in Lemma 3.9. Instead, it appears implicitly via the assumption that (x, y) has an (m, r, t)-distribution (made just before (7)), and plays a central role in the proof of (7).

Proof. To show (5) note that if $\bigoplus_{i \in K} x_i \neq 0$ then B cannot output $P(K)$ on the input $\bar{T}(P(x))$, by our assumption on B, and $\bar{T}$ being invertible. Hence $B'(x) = 0$ in such a case.

To verify (6), denote $x' = \bar{T}(P(x))$ and observe that when $x \sim \{0,1\}^{r \times m}$, we have $x' \sim \{0,1\}^{r \times m}$ independently of P. Hence, by interchanging order of summation,

$$\begin{aligned}\mathop{\mathbb{E}}_{x}[B'(x)] &= \mathop{\mathbb{E}}_{P,\bar{T}}[\mathop{\mathbb{E}}_{x}[\mathbb{1}\{B(\bar{T}(P(x))) = P(K)\}]] = \mathop{\mathbb{E}}_{P,\bar{T}}[\mathop{\mathbb{E}}_{x'}[\mathbb{1}\{B(x') = P(K)\}]] \\ &= \mathop{\mathbb{E}}_{x'}[\mathop{\mathbb{E}}_{P}[\mathbb{1}\{B(x') = P(K)\}]] \leq 1/\binom{r}{k},\end{aligned}$$

where the latter inequality holds because for any fixed x', $P(K)$ attains the value of $B(x')$ with probability at most $1/\binom{r}{k}$.

In order to prove (7), we reason about $\mathbb{E}_{x,y}[B'(x)B'(y)]$. Observe that for any $K' \subseteq [r]$ with $|K'| = k$, the function $B'_{K'}$ defined by $B'_{K'}(x) = \mathbb{E}_{P,T}[\mathbb{1}\{B(\bar{T}(P(x))) = P(K')\}]$ satisfies

$$\mathop{\mathbb{E}}_{x,y}[B'_{K'}(x)B'_{K'}(y)] = \mathop{\mathbb{E}}_{x,y}[B'(x)B'(y)]. \tag{8}$$

Indeed, let $R \in S_r$ be such that $R(K) = K'$. As $(R(x), R(y))$ has the same distribution as (x, y), we have

$$\begin{aligned}\mathbb{E}_{x,y}[B'_{K'}(x)B'_{K'}(y)] &= \mathbb{E}_{x,y}[B'_{K'}(R(x))B'_{K'}(R(y))] \\ &= \mathop{\mathbb{E}}_{x,y}\Big[\mathop{\mathbb{E}}_{P',\bar{T}'}[\mathbb{1}\{B(\bar{T}'(P'R(x))) = P'(K')\}] \mathop{\mathbb{E}}_{P'',\bar{T}''}[\mathbb{1}\{B(\bar{T}''(P''R(y))) = P''(K')\}]\Big] \\ &= \mathop{\mathbb{E}}_{x,y}\Big[\mathop{\mathbb{E}}_{P',\bar{T}'}[\mathbb{1}\{B(\bar{T}'(P'(x))) = P'R^{-1}(K')\}] \mathop{\mathbb{E}}_{P'',\bar{T}''}[\mathbb{1}\{B(\bar{T}''(P''(y))) = P''R^{-1}(K')\}]\Big] \\ &= \mathop{\mathbb{E}}_{x,y}[B'(x)B'(y)].\end{aligned}$$

Notice that if (x, y) has an (m, r, t)-distribution, and $\bar{T}', \bar{T}'' \sim GL_m(\mathbb{F}_2)$ are uniformly random invertible linear maps independent of all other variables, then $(\bar{T}'(x), \bar{T}''(y))$ has an (m, r, t)-

distribution as well. We verify (7):

$$\begin{aligned}
&\Pr_{x,y,P,Q}\left[P^{-1}(B(P(x)))Q^{-1}(B(Q(y)))\right] \\
&\quad= \sum_{K'} \mathop{\mathbb{E}}_{x,y,P,Q}\left[\mathbb{1}\{P^{-1}(B(P(x))) = K'\}\mathbb{1}\{Q^{-1}(B(Q(y))) = K'\}\right] \\
&\quad= \sum_{K'} \mathop{\mathbb{E}}_{x,y,P,Q,\bar{T}',\bar{T}''}\left[\mathbb{1}\{P^{-1}(B(P(\bar{T}'(x)))) = K'\}\mathbb{1}\{Q^{-1}(B(Q(\bar{T}''(y)))) = K'\}\right] \\
&\quad= \sum_{K'} \mathop{\mathbb{E}}_{x,y}\left[B'_{K'}(x)B'_{K'}(y)\right] \\
&\quad= \binom{r}{k} \mathop{\mathbb{E}}_{x,y}\left[B'(x)B'(y)\right],
\end{aligned}$$

where the penultimate equality holds since P (resp. Q) commutes with $\bar{T}'$ (resp. $\bar{T}''$), and the ultimate equality uses (8). □

3.4.3 Obfuscation hides everything but aligned XORs

We begin with the standard definition of the Fourier-Walsh expansion of functions on the discrete cube, adapted to our setting.

Definition 3.11 (Fourier expansion)**.** *Given $S \subseteq [r]\times[m]$, define $\chi_S\colon \{0,1\}^{r\times m} \to \{-1,1\}$ by $\chi_S(x) = (-1)^{\sum_{(i,j)\in S} x_{i,j}}$. The set $\{\chi_S\}_{S\subseteq[r]\times[m]}$ is an orthonormal basis for the set of functions $\left\{f \,\middle|\, f\colon \{0,1\}^{r\times m} \to \mathbb{R}\right\}$, with respect to the standard inner product $\langle f, g\rangle = \mathbb{E}_{x\sim\{0,1\}^{r\times m}}[f(x)g(x)]$. Hence each $f\colon \{0,1\}^{r\times m} \to \mathbb{R}$ can be decomposed to*

$$f = \sum_{S\subseteq[r]\times[m]} \widehat{f}(S)\chi_S,$$

where $\widehat{f}(S) = \langle f, \chi_S\rangle$, and in particular, $\widehat{f}(\emptyset) = \mathbb{E}[f]$.

Definition 3.12 (Cartesian decomposition)**.** *Given $S \subseteq [r]\times[m]$, we call S a* Cartesian product *if there exist $U \subseteq [r]$ and $V \subseteq [m]$ such that $S = U \times V$.*

The Fourier expansion allows decomposing any function $f\colon \{0,1\}^{r\times m} \to \mathbb{R}$ into Cartesian and non-Cartesian parts:

$$f = f^C + f^{\perp} = \left(\sum_{S \text{ Cartesian product}} \widehat{f}(S)\chi_S\right) + \left(\sum_{S \text{ non Cartesian product}} \widehat{f}(S)\chi_S\right),$$

where $\langle f^C, f^{\perp}\rangle = 0$.

Definition 3.13 (Cartesian functions)**.** *A function $f\colon \{0,1\}^{r\times m} \to \mathbb{R}$ is called a* Cartesian function *if $f = f^C$.*

Lemma 3.14. *Let $f\colon \{0,1\}^{r\times m} \to \mathbb{R}$. Suppose (x, y) has an (m, r, t)-distribution. Then*

$$\mathrm{Cov}(f(x), f(y)) \le 2^{-t}\left\|f^C\right\|_2^2 + 2^{-2t}\left\|f^{\perp}\right\|_2^2.$$

Recall that the goal of the obfuscation process is to reduce the correlation between different obfuscations of the same input z (which correspond to different iterations of Algorithm 3.4 described above) to $O_k(1) \cdot 2^{-2t}$. In this respect, the lemma asserts that the obfuscation hides the non-Cartesian part, which corresponds to everything except for *aligned XORs* (that is, expressions of the form $\bigoplus_{i \in I} \bigoplus_{j \in J} x_{i,j}$).

Proof. Write $f = \sum_S \widehat{f}(S)\chi_S$. We have

$$\begin{aligned} \mathrm{Cov}(f(x), f(y)) &= \mathop{\mathbb{E}}_{(x,y)} \big[(\sum_S \widehat{f}(S)\chi_S(x))(\sum_{S'} \widehat{f}(S')\chi_{S'}(y)) \big] - \mathbb{E}[f]^2 \\ &= \mathbb{E}\big[\sum_{S,S'} \widehat{f}(S)\widehat{f}(S')\chi_S(x)\chi_{S'}(y) \big] - \mathbb{E}[f]^2 \\ &\le \sum_S \widehat{f}(S)^2 \Big(\sum_{S'} \big| \mathbb{E}\left[\chi_S(x)\chi_{S'}(y)\right] \big| \Big) - \widehat{f}(\emptyset)^2, \end{aligned} \tag{9}$$

where the last step uses the inequality $\widehat{f}(S)\widehat{f}(S') \le (\widehat{f}(S)^2 + \widehat{f}(S')^2)/2$, applied for all S, S'.

In order to analyze $\mathbb{E}\left[\chi_S(x)\chi_{S'}(y)\right]$, let us recall how (x, y) is distributed according to Definition 3.7. We draw a uniformly random $z \sim \mathbb{F}_2^{r \times (m+t)}$ and two uniformly random rank-m linear maps $T_1, T_2 \colon \mathbb{F}_2^{m+t} \to \mathbb{F}_2^m$ and define $(x, y) = (T_1(z), T_2(z))$.

Observe that there exist linear maps $T_1^*, T_2^* : (\mathbb{F}_2^m)^* \to (\mathbb{F}_2^{m+t})^*$ such that for each $S = (S_1, \ldots, S_r) \subseteq [r] \times [m]$, we have

$$\chi_S(x) = \chi_S(T_1(z)) = \chi_{T_1^* S}(z) \qquad \text{and} \qquad \chi_S(y) = \chi_S(T_2(z)) = \chi_{T_2^* S}(z),$$

where the S_i's are regarded as elements of $(\mathbb{F}_2^m)^* \cong \mathbb{F}_2^m$.

Formally, consider the dual linear maps $T_1^*, T_2^* : (\mathbb{F}_2^m)^* \to (\mathbb{F}_2^{m+t})^*$, defined by

$$T_l^*(f)(a) := f(T_l(a)), \qquad l = 1, 2$$

where $f \colon \mathbb{F}_2^m \to \mathbb{F}_2$ is a linear functional, and $a \in \mathbb{F}_2^{m+t}$ is a vector. (Matrix-wise, the representing matrix of T_l^* according to the (dual) standard basis, is the transpose of the representing matrix of T_l in the standard basis.) Note that each $S \subseteq [r] \times [m]$ naturally corresponds to an r-tuple of linear functionals $(S_1, \ldots, S_r)$, where $S_i(b) = \bigoplus_{j=1}^m (b_j \cdot \mathbb{1}\{(i, j) \in S\})$ for any $b \in \mathbb{F}_2^m$. Thus, we may slightly abuse notation and write $S = (S_1, \ldots, S_r)$, and subsequently, define $T_l^*(S) = (T_l^*(S_1), \ldots, T_l^*(S_r))$ and regard its outputs as elements of $[r] \times [m + t]$.

Since for each $S = (S_1, \ldots, S_r)$, we have $\chi_S(x) = \chi_S(T_1(x)) = \chi_{T_1^* S}(z)$ and $\chi_{S'}(y) = \chi_{T_2^* S'}(z)$ as was written above, and since $\mathbb{E}[\chi_A(z)\chi_B(z)] = \mathbb{1}\{A = B\}$, we can write (9) as

$$\begin{aligned} \mathrm{Cov}(f(x), f(y)) &\le \sum_S \widehat{f}(S)^2 \Big(\sum_{S'} \big| \mathop{\mathbb{E}}_{T_1, T_2} \mathop{\mathbb{E}}_{z} \left[\chi_{T_1^* S}(z)\chi_{T_2^* S'}(z)\right] \big| \Big) - \widehat{f}(\emptyset)^2 \\ &\le \sum_S \widehat{f}(S)^2 \Big(\sum_{S'} \Pr\left[T_1^*(S) = T_2^*(S')\right] \Big) - \widehat{f}(\emptyset)^2. \end{aligned}$$

Noting that T_2^* is injective (as T_2 is of full rank, and duality preserves rank), we conclude

$$\mathrm{Cov}(f(x), f(y)) \le \sum_{S \neq \emptyset} \widehat{f}(S)^2 \Pr\left[T_1^*(S) \in \mathrm{Im}(T_2^*)\right] = \sum_{S \neq \emptyset} \widehat{f}(S)^2 \Pr\Big[\bigwedge_{i=1}^r \left[T_1^*(S_i) \in \mathrm{Im}(T_2^*)\right] \Big].$$

To see that $\Pr[T_1^*(S) \in \mathrm{Im}(T_2^*)] \le 2^{-t}$ for all $S \neq \emptyset$, choose a nonempty row S_i in S, and observe that the probability $\Pr[T_1^*(S_i) \in \mathrm{Im}(T_2^*)]$ is the same as the probability that a specific non-zero vector in $\mathbb{F}_2^{m+t}$ is inside a random subspace of $\mathbb{F}_2^{m+t}$ of dimension m. This probability is $\frac{2^m-1}{2^{m+t}-1} \le 2^{-t}$.

We furthermore claim that $\Pr[T_1^*(S) \in \mathrm{Im}(T_2^*)] \le 2^{-2t}$ whenever S is not a Cartesian product. Indeed, if we choose two different nonempty 'rows' S_i, S_j of S (which is possible as S is not a Cartesian product), the probability $\Pr[T_1^*(S) \in \mathrm{Im}(T_2^*)]$ is upper bounded by the probability that both $T_1^*(S_i) \in \mathrm{Im}(T_2^*)$ and $T_1^*(S_j) \in \mathrm{Im}(T_2^*)$. Since $T_1^*(S_i)$, $T_1^*(S_j)$ are distinct, the aforementioned probability is $\frac{2^m-1}{2^{m+t}-1} \cdot \frac{2^m-2}{2^{m+t}-2} \le 2^{-2t}$. Overall,

$$\begin{aligned}\mathrm{Cov}(f(x), f(y)) &\le 2^{-t} \sum_{S \text{ Cartesian}} \widehat{f}(S)^2 + 2^{-2t} \sum_{S \text{ not Cartesian}} \widehat{f}(S)^2 \\ &= 2^{-t} \left\| f^C \right\|_2^2 + 2^{-2t} \left\| f^\perp \right\|_2^2,\end{aligned}$$

where the ultimate equality uses Parseval's identity. This completes the proof. □

3.4.4 Aligned XORs do not reveal much

Lemma 3.15. *Let $B' \colon \{0,1\}^{r \times m} \to [0,1]$ have $\mathbb{E}_x[B'(x)] = \mu$. Suppose $B'(x) = 0$ whenever $\bigoplus_{i=1}^k x_i \neq 0_m$ ($k > 0$). Then*

$$\left\| (B')^C \right\|_2^2 \le 2^m \mu^2 + 2^{1-m} \mu.$$

Proof. Write $f(x) = \mathbb{1}\{\bigoplus_{i=1}^k x_i = 0\}$ and $B'(x) = f(x) \cdot g(x)$, where $g(x)$ does not depend on x_1. This is possible since we may restrict our attention to those x with $\bigoplus_{i=1}^k x_i = 0$ (B' and f are zero elsewhere), and for such values of x, the value of x_1 can be inferred from $x_2, \ldots, x_k$.

Consider the expansion $B' = \sum_{S \subseteq [r] \times [m]} \alpha_S \chi_S$. One can easily verify that the Fourier expansion of f is

$$f = \sum_U \beta_U \chi_U, \qquad \text{where} \qquad \beta_U = \begin{cases} 2^{-m}, & U = [k] \times U', U' \subseteq [m] \\ 0, & \textit{otherwise.} \end{cases}$$

Denote the Fourier expansion of g by $g = \sum_{V \subseteq ([r] \setminus \{1\}) \times [m]} \gamma_V \chi_V$.

Since $B' = f \cdot g$, for any S we have $\alpha_S = \sum_{U \triangle V = S} \beta_U \gamma_V$. We claim that for any S, this sum consists of a single term, that is, $\alpha_S = \beta_{U_0} \gamma_{V_0}$ for some unique pair (U_0, V_0) with $S = U_0 \triangle V_0$. To see this, assume $U \triangle V = S$ and $\beta_U, \gamma_V \neq 0$. Since g does not depend on x_1, V does not contain elements of the form $(1, i)$. Hence, given S we may decode U as $U = [k] \times U'$, where $U' = \{i : (1,i) \in S\}$ (recall U is Cartesian, as otherwise $\beta_U = 0$). Since we unambiguously determine U, we uniquely determine $V = S \triangle U$.

Let us now compute $\left\| (B')^C \right\|_2^2 = \sum_{S \text{ Cartesian}} \alpha_S^2$. Write each α_S as $\beta_U \gamma_V$. We split the total contribution of the terms $\alpha_S^2 = (\beta_U \gamma_V)^2$ to $\sum \alpha_S^2$ into two cases, depending on V.

Case 1: $V = \emptyset$. Observe that

$$\gamma_V = \mathbb{E}[g] = 2^m \mu.$$

To see this, fix a value $x_2^0, x_3^0, \ldots, x_r^0$ and let x_1^0 be such that $\bigoplus_{i=1}^k x_i^0 = 0_m$. Since g does not depend on x_1, then for any x_1, we have

$$g(x_1, x_2^0, \ldots, x_r^0) = g(x_1^0, x_2^0, \ldots, x_r^0) = B'(x_1^0, x_2^0, \ldots, x_r^0).$$

On the other hand, $B'(x_1, x_2^0, \ldots, x_r^0) = 0$ for any $x_1 \neq x_1^0$ (as $B'(x) = 0$ for every x s.t. $\bigoplus_{i=1}^k x_i \neq 0_m$). Hence, each value x with $B'(x) \neq 0$ corresponds to exactly 2^m values x' s.t. $g(x') = B(x)$, obtained from x by changing only the first coordinate. Consequently, $\mathbb{E}[g] = 2^m\mu$.

Thus, the total contribution of terms with $V = \emptyset$ is at most

$$\sum_U \beta_U^2 \cdot \gamma_V^2 = 2^m\mu^2,$$

using the fact that $\sum_U \beta_U^2 = \mathbb{E}[f^2] = \mathbb{E}[f] = 2^{-m}$ by Parseval's identity.

Case 2: $V \neq \emptyset$. We claim that in this case, for any V, there are at most two U's for which $V \triangle U$ is Cartesian. Indeed, let $i \in [r]$ be such that $V_i := V \cap (\{i\} \times [m]) \neq \emptyset$. Note that $V \cap (\{1\} \times [m]) = \emptyset$. Hence, if $U = ([k] \times U') \neq \emptyset$, then $(V \triangle U) \cap (\{1\} \times [m]) = U'$. On the other hand,

$$(V \triangle U) \cap (\{i\} \times [m]) = \begin{cases} V_i \triangle U', & i \leq k, \\ V_i, & i > k. \end{cases}$$

As $V \triangle U$ is Cartesian, we have $(V \triangle U) \cap (\{i\} \times [m]) = (V \triangle U) \cap (\{1\} \times [m]) = U'$. As $V_i \neq \emptyset$, this is possible only if $i > k$ and $V_i = U'$, that is, $U = [k] \times V_i$. In addition, $U = \emptyset$ is possible if V is Cartesian.

Thus, the total contribution of terms $\alpha_S^2 = (\beta_U\gamma_V)^2$ with $V \neq \emptyset$, is at most (recall $|\beta_U| \leq 2^{-m}$)

$$2 \cdot (2^{-m})^2 \sum_V \gamma_V^2 = 2^{1-2m}\,\mathbb{E}[g^2] \leq 2^{1-2m}\,\mathbb{E}[g] = 2^{1-m}\mu.$$

We conclude

$$\left\|(B')^C\right\|_2^2 = \sum_{S \text{ Cartesian}} \alpha_S^2 \leq 2^m\mu^2 + 2^{1-m}\mu.$$

This completes the proof. □

3.4.5 Wrapping up the proof of the obfuscation lemma

Proof of Theorem 3.8. Combining lemmas 3.14 and 3.15, and using $\mathbb{E}[B'] \leq 1/\binom{r}{k}$, we get

$$\mathrm{Cov}(B'(x), B'(y)) \leq 2^{-2t}/\binom{r}{k} + 2^{-t} \cdot 2^m/\binom{r}{k}^2 + 2^{-t} \cdot 2^{1-m}/\binom{r}{k}.$$

Using Lemma 3.9, we deduce

$$\Pr[P^{-1}(B(P(x))) = Q^{-1}(B(Q(y)))] = \binom{r}{k} \mathop{\mathbb{E}}_{x,y}[B'(x)B'(y)] \leq 2^{-2t} + \frac{2^{m-t}}{\binom{r}{k}} + 2^{-t+1-m}.$$

This completes the proof. □

4 Hardness of Dense k-SUM

In this section we prove Theorem 1.5. The precise formulation of the theorem is as follows.

Theorem 4.1 (Sparse to dense k-SUM reduction)**.** *Let M, N be integers such that $\sqrt{N} \leq M \leq N$. Assume there is an algorithm for $(k, M, N^{1/k})$-SUM with success probability β and expected running time $\mathcal{T}$. Then, there is an algorithm for $(k, N, N^{1/k})$-SUM with success probability at least $\frac{\beta^4}{(\log M)^2 \cdot k^{O(k)}}$ and expected running time at most $N/M \cdot (\mathcal{T} + \tilde{O}_k(N^{1/k}))$.*

The proof is based on the modular k-SUM problem, which we call k-MSUM.

Definition 4.2 (Average-case k-MSUM problem)**.** *In the (k, N, r)-MSUM problem, the input consists of r elements $z_1, \ldots, z_r$, each chosen independently and uniformly at random from $\mathbb{Z}_N$. The goal is to find a k-tuple $K = \{i_1, \ldots, i_k\}$, such that $\sum_{j \in K} z_j \bmod N = 0$.*

Informally, the proof consists of three stages.

1. **Reduction to k-MSUM.** In Section 4.1, we show that for obtaining the reduction from the $(k, N, N^{1/k})$-SUM problem to the $(k, M, N^{1/k})$-SUM problem (i.e. proving Theorem 4.1), it is sufficient to devise a reduction from $(k, pq, r = (pq)^{1/k})$-MSUM to (k, p, r)-MSUM for a pair of primes p, q that satisfy $pq \approx N$ and $p \approx M$.

2. **Obfuscation process.** In Section 4.2, we introduce an obfuscation process that transforms r-tuples of vectors in $\mathbb{Z}_{pq}$ to r-tuples of vectors in $\mathbb{Z}_p$, similarly to the reduction for the k-XOR problem presented in Section 3. Specifically, we show that it is sufficient to prove an *obfuscation lemma* which asserts that the outputs of the (k, p, r)-MSUM oracle are sufficiently independent when it is applied to the 'obfuscated' inputs.

3. **Proof of the obfuscation lemma.** In Section 4.3 we prove the obfuscation lemma by employing techniques from discrete Fourier analysis and combinatorics. It is the most involved part of the proof of the main theorem.

4.1 Reduction to modular k-SUM and proof of Theorem 4.1

The main ingredient in the proof of Theorem 4.1 is the following lemma, which provides reduction from k-MSUM modulo pq to k-MSUM modulo p, for prime numbers p, q with $p \geq q$.

Lemma 4.3 (Main k-MSUM reduction)**.** *Let p, q, r be positive integers such that $p > q$ are prime numbers and $pq \leq \binom{r}{k}$. Assume there is an algorithm for (k, p, r)-MSUM with success probability β and expected running time $\mathcal{T}$.*

Then, there is an algorithm for (k, pq, r)-MSUM with success probability

$$\Omega\left(\left(\frac{\beta^2 \cdot pq}{k \log(q) \cdot \binom{r}{k}}\right)^2\right)$$

and expected running time at most $q \cdot (\mathcal{T} + \tilde{O}(r))$.

In this subsection we prove that Lemma 4.3 implies Theorem 4.1. The (more complex) proof of Lemma 4.3 spans the following subsections.

The derivation of Theorem 4.1 from Lemma 4.3 relies on two additional reductions.

Lemma 4.4 (Simple reduction)**.** *Let r, r' be positive integers such that $r \geq r'$. Assume there is an algorithm for (k, N, r)-SUM (resp., MSUM), with success probability β and expected running time $\mathcal{T}$. Then, there is an algorithm for (k, N, r')-SUM (resp., MSUM) with success probability at least $\frac{\beta}{(2r/r')^k}$ and expected running time at most $\mathcal{T} + \tilde{O}_k(r)$.*

The proof of this lemma is essentially the same as the proof of the analogous Lemma 3.3 for k-XOR, and we omit it.

Lemma 4.5 (k-SUM to k-MSUM reduction)**.** *Let M, N be positive integers such that $M \leq 2N+1$. Assume there is an algorithm for (k, M, r)-MSUM with success probability β and expected running time $\mathcal{T}$. Then, there is an algorithm for (k, N, r)-SUM with success probability $\Omega\left(\frac{\beta}{k(8N \cdot k/M)^k}\right)$ and expected running time at most $\mathcal{T} + \tilde{O}_k(r)$.*

Proof. Denote $r' = \lceil r \cdot M/4N \rceil$. Given an algorithm for (k, M, r)-MSUM with success probability β and running time $\mathcal{T}$, by Lemma 4.4, there is an algorithm B for (k, M, r')-MSUM with success probability at least $\frac{\beta}{(8N/M)^k}$ and expected running time at most $\mathcal{T} + \tilde{O}_k(r)$. We use B to devise an algorithm A for (k, N, r)-SUM.

We first assume that M is odd. On input that consists of r integers $z_1, \ldots, z_r$ uniform in $\{-N, \ldots, N\}$, A performs the following steps.

1. Discard all z_i such that $z_i \notin \{-(M-1)/2, \ldots, (M-1)/2\}$. Denote the number of remaining elements by r_1. If $r_1 < r'$, then return failure. Otherwise, take the first r' remaining elements and denote them by $u_1, \ldots, u_{r'}$.

2. Define the mapping $u_i \mapsto y_i$ (onto $\mathbb{Z}_M$) by $y_i = u_i \bmod M$. Note that $y_1, \ldots, y_{r'}$ is a (k, M, r')-MSUM instance.

3. Pick k vectors $v'_1, \ldots, v'_k$ in $\mathbb{Z}_M$ uniformly at random, conditioned on $\sum_{j \in [k]} v'_j \bmod M = 0$. Then, for each $i \in [r']$, pick $j \in [k]$ uniformly at random and define $v_i = v'_j$ and $x_i = y_i + v_i \bmod M$.

4. Run B on the input $x_1, \ldots, x_{r'}$ and assume it returns a k-tuple K' such that $\sum_{i \in K'} x_i \bmod M = 0$. Trace K' back to the corresponding k-tuple K for $z_1, \ldots, z_r$ and if $\sum_{i \in K} z_i = 0$, return K. Otherwise, return failure.

Note that we do not run B directly on $y_1, \ldots, y_{r'}$ since it may be malicious and return k-tuples that sum to 0 modulo M, but never give a k-SUM over the integers for $z_1, \ldots, z_r$.

Analysis. Clearly, the algorithm returns a correct output if it succeeds and its complexity is as claimed. To analyze the success probability, we consider the following events:

1. $r_1 \geq r'$.

2. B returns a k-tuple K' such that $\sum_{i \in K'} x_i \bmod M = 0$.

3. $\{v_i\}_{i \in K'} = \{v'_j\}_{j \in [k]}$ (as possible multi-sets). Note that this implies $\sum_{i \in K'} v_i \bmod M = \sum_{j \in [k]} v'_j \bmod M = 0$ and therefore,

$$\sum_{i \in K'} u_i \bmod M = \sum_{i \in K'} y_i \bmod M = \sum_{i \in K'} (x_i - v_i) \bmod M = 0.$$

4. $\sum_{i \in K'} u_i = 0$.

Observe that if the fourth event occurs, then $\sum_{i \in K} z_i = \sum_{i \in K'} u_i = 0$ and A succeeds. In the following, we lower bound the probability of these events.

First, $\mathbb{E}[r_1] = r \cdot M/(2N+1)$ and a simple tail bound gives $\Pr[r_1 \geq r'] \geq 1/4$ for r sufficiently large (i.e., larger than some constant value). Conditioned on the first event, the second event occurs with probability at least $\frac{\beta}{(8N/M)^k}$. The third event occurs with probability at least $1/(k!) > k^{-k}$. Note that since $v_1, \ldots, v_{r'}$ are picked independently of $y_1, \ldots, y_{r'}$, then

the third event is independent of the second. Thus, the first three events occur with probability $\Omega\left(\frac{\beta}{(8N\cdot k/M)^k}\right)$.

Finally, recall that $v'_1,\ldots,v'_k$ are picked uniformly at random from $\mathbb{Z}_M$, conditioned on $\sum_{j\in[k]} v'_j \bmod M = 0$. Thus, conditioning on $\sum_{i\in K'} x_i \bmod M = 0$, and on the event that $\{v_i\}_{i\in K'} = \{v'_j\}_{j\in[k]}$ (but not on the individual values of each v'_j), the k-tuple $\{u_i\}_{i\in K'}$ is uniformly distributed in $\{-(M-1)/2,\ldots,(M-1)/2\}^k$, conditioned on $\sum_{i\in K'} u_i \bmod M = 0$. Given this distribution of $\{u_i\}_{i\in K'}$, it remains to lower bound the probability that $\sum_{i\in K'} u_i = 0$ by $1/k$.

Write $U = \sum_{i\in K'} u_i$, as a sum of k uniform integers in $\{-(M-1)/2,\ldots,(M-1)/2\}$. Note that for any t,

$$\begin{aligned} &\Pr[U=t] \le \Pr[U=t+1] \quad \text{if} \quad t+1 \le \mathbb{E}[U], \\ &\Pr[U=t] \ge \Pr[U=t+1] \quad \text{if} \quad t \ge \mathbb{E}[U]. \end{aligned} \tag{10}$$

To see this, observe that the function $t \mapsto \Pr[U=t]$ is log-concave, as a convolution of (discrete) log-concave functions (see for example [3, Proposition 10*(vii)*]), and is symmetric around $t = \mathbb{E}[U]$.

Since in our case $\mathbb{E}[U] = 0$, then for any t, we have $\Pr[U=0] \ge \Pr[U=t]$. Hence,

$$\begin{aligned} \Pr\left[U=0 \,|\, U \bmod M = 0\right] &= \frac{\Pr[U=0]}{\Pr[U \bmod M = 0]} \ge \frac{\Pr[U=t]}{\Pr[U \bmod M = 0]} \\ &\ge \Pr\left[U=t \,|\, U \bmod M = 0\right]. \end{aligned}$$

As $U \in \{-k(M-1)/2,\ldots,k(M-1)/2\}$, given that $U \bmod M = 0$, U can only attain k possible values, implying that $\Pr\left[U=0 \,|\, U \bmod M = 0\right] \ge 1/k$.

Finally, if M is even, we change the algorithm to remove $z_i \notin \{-M/2,\ldots,M/2-1\}$. The analysis is similar, but we have $\mathbb{E}[U] = -k/2$. Nevertheless, the final result is unchanged since $\Pr[U=0] \ge \Pr[U=t]$ for every t such that $t \bmod M = 0$ (assuming M is larger than some constant; we indeed may assume M, r are larger than any fixed constant, hidden by the Ω notation in the assertion). □

We now derive Theorem 4.1 from Lemmas 4.3, 4.4 and 4.5.

Proof of Theorem 4.1. Let M, N be such that $\sqrt{N} \le M \le N$. Our goal is to devise an algorithm for $(k, N, N^{1/k})$-SUM, given an algorithm B for $(k, M, N^{1/k})$-SUM with success probability β and expected running time $\mathcal{T}$.

Clearly, B can be applied to solve $(k, M, N^{1/k}\cdot 2k)$-SUM with the same success probability and complexity.

Let p be a prime number that satisfies $M \le p < 2M$.[3] By Lemma 4.4, based on B, there is an algorithm B_1 for $(k, (p-1)/2, N^{1/k}\cdot 2k)$-SUM with success probability at least $\beta_1 = \frac{\beta}{k^{O(k)}}$ and expected running time at most $\mathcal{T} + \tilde{O}_k(N^{1/k})$. B_1 immediately gives an algorithm for $(k, p, N^{1/k}\cdot 2k)$-MSUM with the same parameters (as a K-tuple that sums to 0 over the integers sums to zero mod p).

Let q be a prime number such that $N/2p \le q < N/p$. Note that we have

$$pq < N < \binom{N^{1/k}\cdot 2k}{k} < N\cdot k^k \qquad \text{and} \qquad q < N/p \le N/M \le M \le p.$$

[3] Such a prime can clearly be found efficiently since by a quantitative version of the prime number theorem, the number of primes between M and $2M$ is $\Omega(M/\log M)$. Therefore, one may pick such a prime by random sampling and using a standard primarily test algorithm.

Hence, we can apply Lemma 4.3 based on B_1 to deduce that there is an algorithm B_2 for $(k, pq, N^{1/k} \cdot 2k)$-MSUM with success probability at least

$$\beta_2 = \Omega\left(\frac{\beta_1^4 (pq)^2}{k^2 \log(q)^2 \binom{N^{1/k} \cdot 2k}{k}^2}\right) \geq \frac{\beta^4}{(\log M)^2 \cdot k^{O(k)}}$$

and expected running time at most $q \cdot (\mathcal{T} + \tilde{O}_k(N^{1/k})) \leq N/M \cdot (\mathcal{T} + \tilde{O}_k(N^{1/k}))$.

Noting that $N/2 \leq pq \leq N$, we invoke Lemma 4.5 based on B_2, and conclude that there is an algorithm B_3 for $(k, N, N^{1/k} \cdot 2k)$-SUM with success probability at least $\beta_3 = \frac{\beta_2}{k^{O(k)}} = \frac{\beta^4}{(\log M)^2 \cdot k^{O(k)}}$ and expected running time at most $N/M \cdot (\mathcal{T} + \tilde{O}_k(N^{1/k}))$.

Finally, we apply Lemma 4.4 based on B_3 and deduce that there is an algorithm for $(k, N, N^{1/k})$-SUM with success probability at least $\frac{\beta_3}{k^{O(k)}} = \frac{\beta^4}{(\log M)^2 \cdot k^{O(k)}}$ and expected running time at most $N/M \cdot (\mathcal{T} + \tilde{O}_k(N^{1/k}))$. □

4.2 The obfuscation process

Our goal in the rest of this section is to prove Lemma 4.3. The proof strategy is similar to the proof of Lemma 3.2 in the k-XOR case presented in Section 3. Namely, we devise an algorithm that receives r vectors in $\mathbb{Z}_{pq}$, denoted by $z_1, \ldots, z_r$, and randomly obfuscates them, returning r vectors in $\mathbb{Z}_p$, denoted by $y_1, \ldots, y_r$. The main properties of the obfuscation are that a solution to the k-MSUM y-problem gives rise to a solution of the k-MSUM z-problem with a good probability (i.e., $\approx 1/q$), and that the applications of the oracle are sufficiently independent to yield a solution of the z-problem with the desired probability.

In this subsection we present the obfuscation algorithm, state the main obfuscation lemma which asserts that our algorithm achieves its goals, and derive Lemma 4.3 from the obfuscation lemma. The proof of the obfuscation lemma is presented in the next subsection.

4.2.1 The obfuscation algorithm and the obfuscation lemma

Let p, q, r be positive integers such that p, q are prime numbers, $p \geq q$, and $pq \leq \binom{r}{k}$. Let B be an algorithm for (k, p, r)-MSUM. Let L be a parameter to be specified below. We define the algorithm A for (k, pq, r)-MSUM, which receives as an input an r-tuple $(z_1, \ldots, z_r) \in \mathbb{Z}_{pq}^r$ of elements in $\mathbb{Z}_{pq}$, as follows.

Algorithm 4.6.

1. **Repeat** L times:
2. **Draw** uniformly random invertible $\alpha \sim \mathbb{Z}_{p \cdot q}^*$ and $\gamma \sim \mathbb{Z}_p^*$ and a uniformly random permutation $P \in S_r$.
3. **Let** $y_i = \gamma \cdot \lfloor (\alpha \cdot z_{P(i)} \,\%\, pq)/q \rceil \,\%\, p$ for all $i \in [r]$.
4. **Feed** B with $(y_1, \ldots, y_r)$. In case it outputs a k-tuple K with $\sum_{i \in K} y_i \% p = 0$, **test** whether $\mathcal{K} = P(K)$ satisfies $\sum_{i \in \mathcal{K}} z_i \% pq = 0$, and **if** it does – **output** the k-tuple $\mathcal{K}$. **Otherwise, continue**.

The obfuscation lemma in the arithmetic case is as follows.

Lemma 4.7. *Let p, q, r be positive integers such that $p \geq q$ are prime numbers and $pq \leq \binom{r}{k}$.*

Let $(z_1, \ldots, z_r) \in \mathbb{Z}_{pq}^r$ be chosen uniformly at random. Let the r-tuples $(y_1^{(1)}, \ldots, y_r^{(1)}) \in \mathbb{Z}_p^r$ and $(y_1^{(2)}, \ldots, y_r^{(2)}) \in \mathbb{Z}_p^r$ be obtained from it by the procedure described above (in two out of the L iterations). Let $\mathcal{K}_1, \mathcal{K}_2$ be the two corresponding $\mathcal{K}$'s obtained in the process. Then,

$$\Pr[\mathcal{K}_1 = \mathcal{K}_2] \leq O(\log(q)/q^2), \tag{11}$$

where the probability is taken over $z, y^{(1)}, y^{(2)}$, and B's randomness.

Note that cases where at least one of $\mathcal{K}_1, \mathcal{K}_2$ is not obtained (that is, when in at least one of the two iterations, Algorithm B fails to find a solution to the (k, p, r)-MSUM problem) are not counted as equality between $\mathcal{K}_1$ and $\mathcal{K}_2$.

4.2.2 Proof of the main reduction lemma

We now deduce Lemma 4.3 from the obfuscation lemma (Lemma 4.7).

Proof of Lemma 4.3. As in the corresponding proof of Lemma 3.2 for k-XOR, we analyze a tweaked version of the algorithm (with the same success probability) in which all L iterations are performed. For any $1 \leq l \leq L$, let $\mathcal{K}_l$ be the $\mathcal{K}$ obtained in the l's iteration ($\mathcal{K}_l$ exists only when B succeeds, i.e., with probability β). Denote by S_l the event that $\mathcal{K}_l$ admits a solution to the (k, pq, r)-MSUM problem. We claim that for all $l = 1, \ldots, L$,

$$\Pr[S_l] \geq \beta \cdot \Omega(1/(\sqrt{k}q)).$$

Indeed, S_l occurs if the B oracle succeeds, and in addition, $\sum_{i \in \mathcal{K}} z_i \,\%\, pq = 0$ holds, given that $\sum_{i \in K} y_i \,\%\, p = 0$.

The probability of the first event is β, as $y_1, \ldots, y_r$ are uniformly and independently distributed in $\mathbb{Z}_p$. (Indeed, conditioning on the variables α, P (but not on $z_{P(i)}$), the variables $\widetilde{y}_i = \lfloor (\alpha \cdot z_{P(i)} \,\%\, pq)/q \rceil \,\%\, p$ are uniformly and independently distributed in $\mathbb{Z}_p$, and so are $y_i = \gamma \cdot \widetilde{y}_i$.)

To see that the probability of the second event is $\Omega(1/(\sqrt{k}q))$, notice that $\alpha z_{P(i)} \,\%\, pq = (q\gamma^{-1} y_i + \sigma_i) \,\%\, pq$, with $\sigma_i \in \{(1-q)/2, \ldots, (q-1)/2\}$. Conditioning on α, γ, P, y_i (and not on $z_{P(i)}$), σ_i is uniformly distributed in this set. Observe that given $\sum_{i \in K} y_i \,\%\, p = 0$, the event $\sum_{i \in K} \alpha z_{P(i)} \,\%\, pq = 0$ is equivalent to $\sum_{i \in K} \sigma_i \,\%\, pq = 0$. The probability of this latter event is $\Omega(1/(\sqrt{k}q))$, for example by (10) and Chebyshev's inequality with the standard deviation of $\sum_{i \in K} \sigma_i$ equal to $\Theta(\sqrt{k}q)$.

Define the random variables

$$Z' := \sum_{l=1}^{L} \mathbb{1}\{S_l\} - \sum_{1 \leq l < l' \leq L} \mathbb{1}\{\mathcal{K}_l = \mathcal{K}_{l'}\}, \qquad Z := \max(Z', 0).$$

Similarly to the proof of Lemma 3.2 in Section 3.3, it is easy to verify that (tweaked) Algorithm 4.6 succeeds to solve the (k, pq, r)-MSUM problem with probability at least

$$\Pr[Z > 0] \geq \frac{\mathbb{E}[Z]^2}{\mathbb{E}[Z^2]}.$$

To bound $\mathbb{E}[Z]$ from below, note that

$$\mathbb{E}[Z] \geq \mathbb{E}[Z'] = \sum_{l=1}^{L} \Pr[S_l] - \sum_{1 \leq l < l' \leq L} \Pr[\mathcal{K}_l = \mathcal{K}_{l'}].$$

Using $\Pr[S_l] \geq \beta \cdot \Omega(1/(\sqrt{k}q))$ and Lemma 4.7, we get

$$\mathbb{E}[Z] \geq L \cdot \Omega(\beta/(\sqrt{k}q)) - \binom{L}{2} O(\log(q)/q^2).$$

We choose $L = c \cdot \beta q/(\sqrt{k}\log(q))$ for a sufficiently small constant c, and obtain

$$\mathbb{E}[Z] \geq \Omega\Big(\frac{\beta^2}{k\log(q)}\Big).$$

To bound $\mathbb{E}[Z^2]$ from above, note that similarly to the proof of Lemma 3.2, we have

$$\mathbb{E}[Z^2] \leq \frac{2}{(pq)^2}\binom{r}{k}^2.$$

Therefore, (tweaked) Algorithm 4.6 succeeds with probability at least

$$\Pr[Z > 0] \geq \Omega(\beta^4) \cdot \Big(\frac{pq}{\binom{r}{k} k\log(q)}\Big)^2.$$

The running time of the algorithm is

$$L \cdot (\mathcal{T} + \tilde{O}(r)) \leq q \cdot (\mathcal{T} + \tilde{O}(r)).$$

This completes the proof of the lemma. □

4.3 Proof of the obfuscation lemma

In this section we prove Lemma 4.7. We start by introducing a distribution that models two independent outputs of the obfuscation process, and restate the obfuscation lemma.

Definition 4.8. *Let p, q be prime numbers. We say that a pair of random variables $(x^{(1)}, x^{(2)})$, each taking values in $\mathbb{Z}_p^r$, has a (p, q, r)-arithmetic-distribution, if there exist random variables z, $\alpha^{(j)}$, $\gamma^{(j)}$, $j = 1, 2$, with:*

1. *z, $\alpha^{(1)}$, $\alpha^{(2)}$, $\gamma^{(1)}$, $\gamma^{(2)}$, are independent random variables.*
2. *$z \sim \mathbb{Z}_{p\cdot q}^r$ is uniformly distributed.*
3. *$\alpha^{(j)} \sim \mathbb{Z}_{p\cdot q}^*$ is a uniformly random invertible element of $\mathbb{Z}_{p\cdot q}$.*
4. *$\gamma^{(j)} \sim \mathbb{Z}_p^*$ is a uniformly random nonzero residue modulo p.*
5. *For all $i = 1 \ldots r$, $j = 1, 2$, we have*

$$x_i^{(j)} = \gamma^{(j)} \cdot \lfloor \alpha^{(j)} \cdot z_i/q \rceil \,\%\, p.$$

Lemma 4.9. *Let A be an algorithm receiving as input a list of r integers in $\mathbb{Z}_p$, and outputs the indices of $k > 0$ numbers among them whose SUM is 0 (modulo p). If (x, y) has a (p, q, r)-arithmetic-distribution with $p \geq q$, and $P, Q \sim S_r$ are two uniformly random and independent permutations, then*

$$\Pr[P^{-1}(A(P(x))) = Q^{-1}(A(Q(y)))] \leq O\Big(\frac{\log(q)}{q^2} + \frac{p}{q\binom{r}{k}}\Big), \tag{12}$$

where the probability is taken over A's randomness, x, y and P, Q. (The event on the left hand side is contained in the event that both executions $B(P(x))$, $B(Q(y))$ succeed).

Notice that Lemma 4.7 immediately follows from Lemma 4.9 (compare (11) with (12)).

4.3.1 Proof outline

The proof of Lemma 4.9 uses techniques from *discrete Fourier analysis* and combinatorial techniques. Technically, the proof is more complicated than for k-XOR since the k-SUM analog of Lemma 3.14 does not hold. Instead, it is replaced by the sequence of lemmas 4.16–4.21. The proof consists of three steps presented in the order of appearance in the paper.

1. **Transformation to real-valued functions.** Similarly to the k-XOR obfuscation, we show that instead of analyzing the obfuscation on a tuple-valued function, it is sufficient to analyze its action on the simpler class of real-valued functions. We utilize the fact that our obfuscation randomly permutes the input vectors, so that any oracle $B : \mathbb{Z}_p^r \to \binom{[r]}{k}$ must, informally, treat all candidate output k-tuples in the same way. Hence it suffices to analyze the modified, real-valued, oracle $B' : \mathbb{Z}_p^r \to [0,1]$ which essentially indicates the probability that B outputs the specific k-tuple $K := \{1, \ldots, k\}$ when applied on its input. Specifically, our task is reduced to showing that

$$\mathbb{E}[B'(x)B'(y)] \leq O\Big(\frac{\log(q)}{q^2} + \frac{p}{q\binom{r}{k}}\Big)/r^k, \tag{13}$$

   where x, y are two independent obfuscations of a common, random, $z \in \mathbb{Z}_{pq}^r$.

2. **Representing the correlation in terms of the Fourier expansion.** In order to prove (13), we consider the Fourier expansion of B', namely,

$$B' = \sum_{S \in \mathbb{Z}_p^r} \widehat{B'}(S)\chi_S, \qquad \text{where} \qquad \chi_S(v) = \exp\left(\frac{2\pi i}{p}\langle S, v\rangle\right), \qquad \widehat{B'}(S) \in \mathbb{C}.$$

   It turns out that the correlation between two obfuscations (which appear in different iterations of Algorithm A described above) is a weighted sum of the squared Fourier coefficients $\widehat{B'}(S)^2$

$$\mathrm{Cov}(B'(x), B'(y)) = \sum_{S \neq 0} (p-1) M_{p,q,r}(S) \widehat{B'}(S)^2, \tag{14}$$

   where $M_{p,q,r}(S)$ roughly serve as the 'weight' for $\widehat{B'}(S)^2$, and are defined as $M_{p,q,r}(S) = \mathbb{E}[\chi_S(x)\overline{\chi_S(y)}]$.

3. **Bounding the correlation using discrete Fourier analysis.** We bound the correlation, the right hand side of (14), in two steps:

   (a) *Partitioning into 2-dimensional subspaces.* We use the structure of solutions to the k-SUM problem to show that it is sufficient to bound $\sum_{S \in U} M_{p,q,r}(S)$ over two-dimensional subspaces $U = \{aS_1 + bS_2 : a, b \in \mathbb{Z}_p\} \subseteq \mathbb{Z}_p^r$ of a certain kind.

   (b) *Bounding $\sum_S M_{p,q,r}(S)$ over 2-dimensional subspaces.* We bound $\sum_{S \in U} M_{p,q,r}(S)$ over two-dimensional subspaces U, using a combinatorial approach. Specifically, we represent this sum as the bias introduced in an event, caused by the dependence between x, y. We show that this bias is related to a Littlewood-Offord-type problem [25], and use an antichain argument (which is a classical tool in Littlewood-Offord theory – see [16]) along with simple number theoretic estimates, to bound it.

   Combination of the two steps completes the proof.

4.3.2 Transformation to real-valued functions

Lemma 4.10. *Let $B\colon \mathbb{Z}_p^r \to \binom{[r]}{k}$ be an algorithm that outputs either a k-tuple R with $\sum_{i\in R} x_i \% p = 0$, or a failure message. Let $K := \{1,\ldots,k\}$ and define $B'\colon \mathbb{Z}_p^r \to [0,1]$ by*

$$B'(x) = \Pr_{P,\gamma}[B(P(\gamma\cdot x)) = P(\{1,2,\ldots,k\})], \tag{15}$$

where $P \sim S_r$ is a uniformly random permutation, and $\gamma \sim \mathbb{Z}_p^*$ is a uniformly random invertible element of $\mathbb{Z}_p$. Then,

$$\sum_{i\in K} x_i \neq 0 \implies B'(x) = 0, \tag{16}$$

$$\mu := \mathop{\mathbb{E}}_{x}[B'(x)] \leq 1/\binom{r}{k}, \tag{17}$$

and if (x,y) has a (p,q,r)-arithmetic distribution and $P,Q \sim S_r$ are independent, then

$$\Pr[P^{-1}(B(P(x))) = Q^{-1}(B(Q(y)))] = \binom{r}{k} \mathop{\mathbb{E}}_{x,y}[B'(x)B'(y)]. \tag{18}$$

We note that the probability in (15) is also taken over B's randomness. The proof of the lemma is omitted, being similar to the proof of Lemma 3.9.

Due to the structure of the k-SUM problem, the function B' has several properties that will be crucially used in the sequel. Before stating these properties, let us introduce Fourier expansion over $\mathbb{Z}_p$.

Definition 4.11 (Fourier expansion)**.** *Given $S \in \mathbb{Z}_p^r$, define $\chi_S\colon \mathbb{Z}_p^r \to \mathbb{C}$ by $\chi_S(x) = e_p(\langle S,x\rangle)$, where*

$$e_p(a) := \exp\Big(\frac{2\pi i a}{p}\Big).$$

The set $\{\chi_S\}_{S\in\mathbb{Z}_p^r}$ is an orthonormal basis for the set of functions $\big\{f \,\big|\, f\colon \mathbb{Z}_p^r \to \mathbb{C}\big\}$, with respect to the standard inner product $\langle f,g\rangle = \mathbb{E}_{x\sim\mathbb{Z}_p^r}[f(x)\overline{g(x)}]$. Hence, each $f\colon \mathbb{Z}_p^r \to \mathbb{C}$ can uniquely be decomposed as

$$f = \sum_{S\in\mathbb{Z}_p^r} \widehat{f}(S)\chi_S, \qquad \text{with} \qquad \widehat{f}(S) \in \mathbb{C}.$$

Claim 4.12. *Let B' be defined as in Lemma 4.10. Then:*

1. *For any $x \in \mathbb{Z}_p^r$ and any $\gamma \in \mathbb{Z}_p^*$, we have $B'(x) = B'(\gamma\cdot x)$.*
2. *B' can be written in the form $B'(x) = I_k(x)\cdot g(x)$, where $I_k(x) = \mathbb{1}\{\sum_{i=1}^k x_i = 0\}$.*
3. *Let $S' \in \mathbb{Z}_p^r$ be defined by $S'_j = \mathbb{1}\{j \in [k]\}$. For any $S \in \mathbb{Z}_p^r$ we have $\widehat{B'}(S) = \widehat{B'}(S+S')$.*

Proof. The first assertion holds trivially, by the definition of B'. The second holds since $B(x) = 0$ whenever $I_k(x) = 0$, with $I_k(x) = \mathbb{1}\{\sum_{i=1}^k x_i = 0\}$. Finally, the third holds as the Fourier expansion of I_k is given by $I_k = \frac{1}{p}\sum_{\nu\in\mathbb{Z}_p} \chi_{\nu\cdot S'}$ with $S'_j = \mathbb{1}\{j \in [k]\}$. □

Following Claim 4.12, we shall study and exploit properties of functions $f : \mathbb{Z}_p^r \to [0,1]$ that satisfy $f(\gamma x) = f(x)$ and $\widehat{f}(S+S') = \widehat{f}(S)$ for S' as defined in Claim 4.12 and all x,γ,S.

4.3.3 Representing the correlation in terms of the Fourier expansion

In this subsection we present a sequence of lemmas that shall be used in our proof. These lemmas allow us to represent the correlation between different obfuscations in terms of the Fourier expansion, and will be helpful in bounding the correlation using discrete Fourier analysis in the following subsections. Note that all inner products from now on are between elements of $\mathbb{Z}_p^r$, and consequently, their results lie in $\mathbb{Z}_p$.

An alternative representation of the obfuscation. In our proof, we shall frequently use the following alternative view of the obfuscation.

Claim 4.13. *A pair (x, y) taking a (p, q, r)-arithmetic-distribution may be sampled by drawing $x \sim \mathbb{Z}_p^r$ uniformly at random, choosing $v \sim \{(1-q)/2, (3-q)/2, \ldots, (q-1)/2\}^r$ uniformly, along with $\alpha \sim \mathbb{Z}_{p\cdot q}^*$ and $\gamma, \gamma' \sim \mathbb{Z}_p^*$, and setting*

$$\forall i \in [r]\colon \quad y_i = (\gamma x_i + \gamma' \lfloor \alpha v_i / q \rceil) \,\%\, p.$$

Proof. Assume (x, y) has a (p, q, r)-arithmetic distribution, that is, $x_i = \gamma_1 \cdot \lfloor \alpha_1 \cdot z_i/q \rceil \% p$ and $y_i = \gamma_2 \cdot \lfloor \alpha_2 \cdot z_i/q \rceil \% p$, where $z, \alpha_1, \alpha_2, \gamma_1, \gamma_2$ are as in Definition 4.8. Write $\alpha_1 z_i \%(pq) = q(\gamma_1^{-1} x_i \% p) + v_i$ with $v_i \in \{(1-q)/2, \ldots, (q-1)/2\}$. Note that, under the fixing of any (invertible) γ_1, α_1, each pair of $(x_i, v_i) \in \mathbb{Z}_p \times \{(1-q)/2, \ldots, (q-1)/2\}$ arises from exactly one $z_i \in \mathbb{Z}_{pq}$. Since z_i is uniformly distributed and independent of γ_i, α_i, we see that x_i, v_i are uniformly distributed (as stated in the claim) and independent of each other and of γ_i, α_i.

Finally,

$$y_i = \gamma_2 \cdot \lfloor \alpha_2 \cdot \alpha_1^{-1} (q(\gamma_1^{-1} x_i \% p) + v_i)/q \rceil \% p,$$

meaning that

$$y_i = ((\gamma_2 \alpha_2 \alpha_1^{-1} \gamma_1^{-1} \% p) x_i + \gamma_2 \lfloor \alpha_2 \alpha_1^{-1} v_i / q \rceil) \% p.$$

Denoting $\gamma := (\gamma_2 \alpha_2 \alpha_1^{-1} \gamma_1^{-1} \% p)$, $\gamma' := \gamma_2$ and $\alpha := \alpha_2 \alpha_1^{-1}$, we have $y_i = (\gamma x_i + \gamma' \lfloor \alpha v_i / q \rceil) \,\%\, p$. Note that γ, γ', α have the asserted distribution and are independent of x, v. □

A quantity representing the contribution of $\widehat{B}'(S)$ to the correlation. We now formally introduce the notion $M_{p,q,r}(S)$ that will play a central role in the proof. The relevance of the notion to the correlation we study is shown in Lemma 4.16 below.

Definition 4.14. *Let $S \in \mathbb{Z}_p^r$, and let (x, y) be a pair that has a (p, q, r)-arithmetic distribution. Define the magnitude of S as (the real number)*

$$M_{p,q,r}(S) := \mathbb{E}[\chi_S(x) \overline{\chi_S(y)}].$$

Lemma 4.15 (Orthogonality)**.** *Let $S, S' \in \mathbb{Z}_p^r$, and assume (x, y) has a (p, q, r)-arithmetic distribution. If $S' = \gamma' S$ with $\gamma' \in \mathbb{Z}_p^*$, then we have*

$$\mathbb{E}[\chi_S(x) \overline{\chi_{S'}(y)}] = M_{p,q,r}(S). \tag{19}$$

Otherwise (if $S' \neq \gamma' S$ for all $\gamma' \in \mathbb{Z}_p^*$),

$$\mathbb{E}[\chi_S(x) \overline{\chi_{S'}(y)}] = 0. \tag{20}$$

Proof. To verify (19), assume $S' = \gamma' S$ with $\gamma' \in \mathbb{Z}_p^*$. Note that by Definition 4.8, if (x, y) has a (p, q, r)-arithmetic-distribution, then $(x, \gamma'^{-1} y)$ admits a (p, q, r)-arithmetic-distribution as well. Hence,

$$\mathbb{E}[\chi_S(x)\overline{\chi_{S'}(y)}] = \mathbb{E}[\chi_S(x)\overline{\chi_{S'}(\gamma'^{-1}y)}] = \mathbb{E}[\chi_S(x)\overline{\chi_S(y)}] = M_{p,q,r}(S).$$

To verify (20), recall that Claim 4.13 shows that $x \sim \mathbb{Z}_p^r$ and $y_i = \gamma x_i + \gamma' u_i$, where $\gamma, \gamma' \sim \mathbb{Z}_p^*$, and u is independent of x (its distribution is irrelevant for the current proof). Condition on the values of γ, γ', u, so that

$$\mathbb{E}[\chi_S(x)\overline{\chi_{S'}(y)}] = \mathop{\mathbb{E}}_{\gamma,\gamma',u}[\mathop{\mathbb{E}}_{x}\left[e_p(\langle S, x\rangle - \langle S', y\rangle)\,\middle|\,\gamma, \gamma', u\right]].$$

Notice that given γ, γ', and u, the expression $\langle S, x\rangle - \langle S', y\rangle$ is linear in x, and is **non-constant** (for all γ, γ', u), since S, S' are non-proportional. Hence, $\mathbb{E}[e_p(\langle S, x\rangle - \langle S', y\rangle)] = 0$, since $x \sim \mathbb{Z}_p^r$, and the expected value of $e_p(x')$ when $x' \sim \mathbb{Z}_p$ is uniformly distributed is 0. □

Notation. Given two vectors $S, S' \in \mathbb{Z}_p^r$ and two scalars $\alpha, \beta \in \mathbb{Z}_p$, we denote by $\alpha S + \beta S' \in \mathbb{Z}_p^r$ the vector S'' having for all $i \in [r]$,

$$S''_i = (\alpha S_i + \beta S'_i) \,\%\, p.$$

Representing the correlation in terms of $M_{p,q,r}$ and the Fourier expansion. The following lemma shows how $M_{p,q,r}$ can be used to estimate the correlation $\mathrm{Cov}(B'(x), B'(y))$ we aim at bounding, thus establishing (14).

Lemma 4.16. *Let $f\colon \mathbb{Z}_p^r \to \mathbb{C}$ have $f(\gamma x) = f(x)$ for all $x \in \mathbb{Z}_p^r$, and $\gamma \in \mathbb{Z}_p^*$. Suppose (x, y) has a (p, q, r)-arithmetic-distribution. Then*

$$\mathrm{Cov}(f(x), f(y)) = \sum_{\substack{S \in \mathbb{Z}_p^r \\ S \neq 0_r}} (p-1) M_{p,q,r}(S) \widehat{f}(S)^2.$$

Proof. Using the expansion $f(x) = \sum_{S \in \mathbb{Z}_p^r} \widehat{f}(S)\chi_S(x)$, we find

$$\mathrm{Cov}(f(x), f(y)) = \mathop{\mathbb{E}}_{x,y}[(f(x) - \widehat{f}(0))\overline{(f(y) - \widehat{f}(0))}] = \sum_{S, S' \in \mathbb{Z}_p^r \setminus \{0_r\}} \widehat{f}(S)\widehat{f}(S') \mathop{\mathbb{E}}_{x,y}[\chi_S(x)\overline{\chi_{S'}(y)}].$$

By comparing coefficients, the assumption that $f(\gamma x) = f(x)$ (for all x) implies $\widehat{f}(S) = \widehat{f}(\gamma S)$. Combining with Lemma 4.15 we get

$$\mathrm{Cov}(f(x), f(y)) = \sum_{S \neq 0_r} \sum_{\gamma \in \mathbb{Z}_p^*} \widehat{f}(S)\widehat{f}(\gamma S) M_{p,q,r}(S) = (p-1) \sum_{S \neq 0_r} \widehat{f}(S)^2 M_{p,q,r}(S).$$

□

Bounding $M_{p,q,r}(S)$. The following two lemmas allows us to bound $M_{p,q,r}(S)$.

Lemma 4.17. *Suppose (x, y) has a (p, q, r)-arithmetic-distribution. For any $r \in \mathbb{N}$, any primes $p, q > 0$, and any non-zero $T \in \mathbb{Z}_p^r$, we have*

$$M_{p,q,r}(T) = \frac{p^2 \Pr[\langle T, x\rangle = \langle T, y\rangle = 0] - 1}{(p-1)^2}. \tag{21}$$

Proof. Assume (x, y) has a (p, q, r)-arithmetic-distribution, and let $T \in \mathbb{Z}_p^r$ be any vector. We show

$$\begin{aligned} M_{p,q,r}(T) = 1 \cdot \Pr[\langle T, x\rangle = \langle T, y\rangle = 0] \\ - \frac{1}{p-1} \Pr[\langle T, x\rangle = 0 \text{ XOR } \langle T, y\rangle = 0] \\ + \frac{1}{(p-1)^2} \Pr[\langle T, x\rangle \neq 0 \wedge \langle T, y\rangle \neq 0], \end{aligned} \tag{22}$$

where $(\langle T, x\rangle = 0 \text{ XOR } \langle T, y\rangle = 0)$ denotes the event that exactly one of $\langle T, x\rangle = 0$, $\langle T, y\rangle = 0$ holds. To verify (22), note that if (x, y) has a (p, q, r)-arithmetic distribution, then so does $(\gamma_1 x, \gamma_2 y)$, for any $\gamma_1, \gamma_2 \in \mathbb{Z}_p^*$. Hence,

$$\begin{aligned} \mathop{\mathbb{E}}_{x,y}[\chi_T(x)\overline{\chi_T(y)}] &= \mathop{\mathbb{E}}_{x,y}[e_p(\langle T, x\rangle - \langle T, y\rangle)] \\ &= \mathop{\mathbb{E}}_{x,y}[e_p(\gamma_1 \cdot \langle T, x\rangle - \gamma_2 \cdot \langle T, y\rangle)]. \end{aligned}$$

Letting γ_1, γ_2 be uniformly distributed in $\mathbb{Z}_p^*$ (independently of (x, y)), one can verify that for any fixed x, y we have

$$\begin{aligned} \mathop{\mathbb{E}}_{\gamma_1,\gamma_2}[e_p(\gamma_1 \cdot \langle T, x\rangle - \gamma_2 \cdot \langle T, y\rangle)] = \mathop{\mathbb{E}}_{\gamma_1}[e_p(\gamma_1 \cdot \langle T, x\rangle)] \mathop{\mathbb{E}}_{\gamma_2}[e_p(\gamma_2 \cdot \langle T, y\rangle)] \\ = \Big(\mathbb{1}\{\langle T, x\rangle = 0\} - \frac{\mathbb{1}\{\langle T, x\rangle \neq 0\}}{p-1}\Big) \cdot \Big(\mathbb{1}\{\langle T, y\rangle = 0\} - \frac{\mathbb{1}\{\langle T, y\rangle \neq 0\}}{p-1}\Big). \end{aligned} \tag{23}$$

Equation (22) then follows by averaging (23) over (x, y).

Denote the three probabilities in (22) by A, B, C, that is:

$$\begin{aligned} A &:= \Pr[\langle T, x\rangle = \langle T, y\rangle = 0], \\ B &:= \Pr[\langle T, x\rangle = 0 \text{ XOR } \langle T, y\rangle = 0], \\ C &:= \Pr[\langle T, x\rangle \neq 0 \wedge \langle T, y\rangle \neq 0]. \end{aligned}$$

Note that $A + B + C = 1$, and that for any $T \neq 0$, we have $2A + B = 2 \cdot \Pr[\langle T, x\rangle = 0] = 2/p$, since each of x, y is uniformly distributed in $\mathbb{Z}_p^r$. Substituting into (22) and simplifying, we obtain

$$M_{p,q,r}(T) = \frac{p^2 \Pr[\langle T, x\rangle = \langle T, y\rangle = 0] - 1}{(p-1)^2},$$

as asserted. □

Lemma 4.18. *Let $r \in \mathbb{N}$, let $p \geq q > 0$ be prime numbers, and let $S \in \mathbb{Z}_p^r$ be a nonzero vector. If (x, y) has a (p, q, r)-arithmetic-distribution, then*

$$\Pr_{x,y}[\langle S, x\rangle = \langle S, y\rangle = 0] \leq O(1/(pq) + 1/p^2) \leq O(1/(pq)). \tag{24}$$

Proof. Recall the distribution of x, y given by Claim 4.13, namely $y_i = (\gamma x_i + \gamma' u_i) \,\%\, p$ with $u_i = \lfloor \alpha v_i / q \rceil \,\%\, p$, where $x, v, \gamma, \gamma', \alpha$ are independent random variables. By the independence of x, u, we get

$$\Pr_{x,y}[\langle S, x\rangle = \langle S, y\rangle = 0] = \Pr_x[\langle S, x\rangle = 0] \Pr_u[\langle S, u\rangle = 0]. \tag{25}$$

Since x is uniformly random and S is nonzero, $\Pr_x[\langle S, x\rangle = 0] = 1/p$. It thus remains to show

$$\Pr_u[\langle S, u\rangle = 0] \leq O(1/q + 1/p). \tag{26}$$

To prove (26), we note that it is a Littlewood-Offord-type statement: the entries u_i are independent random variables, and $\langle S, u\rangle$ is their weighted sum, and we are concerned with the probability it attains a specific value.

We tackle the problem by using a standard antichain argument. Roughly, choose an i with $S_i \neq 0$, and condition on the values of u_j for all $j \neq i$. Then, there is at most one value of u_i that would make $\langle S, u\rangle = S_i u_i + \sum_{j:j\neq i} S_j u_j \,\%\, p = 0$. Recall $u_i = \lfloor \alpha v_i/q \rceil \,\%\, p$, where v_i is uniformly distributed in $\{(1-q)/2, \ldots, (q-1)/2\}$. Hence, assuming that the map $v_i \mapsto u_i$ is injective, we have that $\langle S, u\rangle = 0$ with probability $\leq 1/q$, as required. This last assumption is not strictly correct, however it can be corrected as follows.

Depending on α and on $\{u_j : j \neq i\}$, we let $Z = \{\tau \in \{(1-q)/2, \ldots, (q-1)/2\} : (S_i \lfloor \alpha\tau/q \rceil + \sum_{j:j\neq i} S_j u_j) \,\%\, p = 0\}$, and note that according to the above discussion, $\Pr[\langle S, u\rangle = 0] = \mathbb{E}[|Z|]/q$. Hence, (26) is reduced to showing that $\mathbb{E}[|Z|] \leq O(1 + q/p)$. Write $Z = \{\tau_1, \ldots, \tau_k\}$ with $\tau_1 < \ldots < \tau_k$, and $Z' = \{\tau_2 - \tau_1, \tau_3 - \tau_1, \ldots, \tau_k - \tau_1\}$. Note that $|Z| = |Z'| + 1$, and that every $\sigma \in Z'$ satisfies $\alpha\sigma \in pq\mathbb{Z} + (-q, q)$ (i.e., the residue $(\alpha\sigma) \,\%\, (pq)$ is either smaller than q, or larger than $pq - q$). Hence,

$$\mathbb{E}[|Z|] \leq 1 + \mathbb{E}[|Z'|] \leq 1 + \mathop{\mathbb{E}}_{\alpha}[\sum_{\sigma=1}^{q-1} \mathbb{1}\{\alpha\sigma \in pq\mathbb{Z} + (-q, q)\}] = O(1 + q/p),$$

where the last bound follows since for any fixed σ, $\alpha\sigma \,\%\, (pq)$ is uniformly distributed in $\mathbb{Z}^*_{pq}$ (recall $\sigma < q \leq p$), and the probability it is in $pq\mathbb{Z} + (-q, q)$ is $O(1/p)$.

□

Lemmas 4.17 and 4.18 yield the following corollary, that upper bounds $M_{p,q,r}(S)$.

Corollary 4.19. *Let $r \in \mathbb{N}$, let $p \geq q > 0$ be prime numbers, and let $S \in \mathbb{Z}_p^r$ be a nonzero vector. Then*

$$|M_{p,q,r}(S)| \leq O(1/(pq)).$$

Proof. By (21) we have

$$M_{p,q,r}(S) = \frac{p^2 \Pr[\langle S, x\rangle = \langle S, y\rangle = 0] - 1}{(p-1)^2}.$$

By (24) we have

$$\Pr[\langle S, x\rangle = \langle S, y\rangle = 0] \leq O(1/(pq)).$$

These two estimates yield the desirable $|M_{p,q,r}(S)| = O(1/(pq) + 1/p^2) = O(1/(pq))$. □

4.3.4 Partitioning into 2-dimensional subspaces

While Corollary 4.19, in conjunction with Lemma 4.16, allows us bounding $\mathrm{Cov}(B'(x), B'(y))$ from above (which is the main task we are tackling), the obtained upper bound is not sufficiently tight for our purposes. To achieve a stronger bound, we use the special structure of B' observed in Claim 4.12 – namely, that it satisfies $B'(x) = B'(\gamma x)$ for all $\gamma \in \mathbb{Z}_p^*$, and that its Fourier expansion satisfies $\widehat{B'}(S) = \widehat{B'}(S + S')$ for all S and the specific S' defined in Claim 4.12 – to show that it is sufficient to bound the sums $\sum_{S\in U} M_{p,q,r}(S)$ over certain 2-dimensional subspaces U.

Lemma 4.20. *Let $r \in \mathbb{N}$, and let $p \geq q$ be prime numbers. Let $f\colon \mathbb{Z}_p^r \to [0,1]$ satisfy $f(x) = f(\gamma x)$ for all $x \in \mathbb{Z}_p^r$ and $\gamma \in \mathbb{Z}_p^*$. Furthermore, assume there exists a particular nonzero vector $S' \in \mathbb{Z}_p^r$ such that $\widehat{f}(S+S') = \widehat{f}(S)$ for all $S \in \mathbb{Z}_p^r$.*

Let C be a constant such that for all S satisfying $\forall \nu \in \mathbb{Z}_p\colon S \neq \nu S'$,

$$\sum_{\substack{\eta \in \mathbb{Z}_p^* \\ \nu \in \mathbb{Z}_p}} M_{p,q,r}(\eta S + \nu S') \leq C.$$

If $\mu = \mathbb{E}[f]$, and (x,y) has a (p,q,r)-arithmetic-distribution, then

$$\mathrm{Cov}(f(x), f(y)) \leq O\Big(\frac{p}{q}\mu^2 + \frac{C}{p}\mu\Big).$$

Proof. By Lemma 4.16, we have

$$\mathrm{Cov}(f(x), f(y)) = (p-1) \sum_{S \neq 0_r} M_{p,q,r}(S) \widehat{f}(S)^2$$

There are two kinds of contributions to the right hand side, corresponding to elements S with $S = \nu S'$, and to other elements.

Case 1: The contribution of each S with $S = \nu S'$ for $\nu \in \mathbb{Z}_p^*$, is $M_{p,q,r}(S') \cdot \widehat{f}(S')^2$. (Recall that by Lemma 4.15, for such an S, we have $M_{p,q,r}(S') = M_{p,q,r}(S)$, and since $f(x) = f(\gamma x)$, we have $\widehat{f}(S) = \widehat{f}(S')$.) We note that $|\widehat{f}(S)| = |\mathbb{E}_x[f(x)\overline{\chi_S(x)}]| \leq \mathbb{E}\,|f(x)| = \mu$. Using the bound on $M_{p,q,r}(S)$ from Corollary 4.19, and the fact that there are only $p-1$ such S's, we get that the total contribution in this case is

$$(p-1) \sum_{S = \nu S'} M_{p,q,r}(S) \widehat{f}(S)^2 \leq (p-1)^2 M_{p,q,r}(S') \mu^2 \leq p^2 O(1/(pq)) \mu^2 \leq O\Big(\frac{p}{q}\mu^2\Big).$$

Case 2: The contribution of elements S with $S \neq \nu S'$ for all $\nu \in \mathbb{Z}_p^*$ (we denote this family of elements by $\mathcal{S}$), can be analyzed using the assumption that $\widehat{f}(S + \nu S') = \widehat{f}(S)$ for all $\nu \in \mathbb{Z}_p$. It follows that

$$(p-1) \sum_{S \in \mathcal{S}} M_{p,q,r}(S) \widehat{f}(S)^2 = \frac{p-1}{p(p-1)} \sum_{S \in \mathcal{S}} \widehat{f}(S)^2 \cdot \sum_{\substack{\eta \in \mathbb{Z}_p^* \\ \nu \in \mathbb{Z}_p}} M_{p,q,r}(\eta S + \nu S') \leq \frac{C\mu}{p}.$$

Here, the first equality holds since each summand on the left hand side appears $p(p-1)$ times on the right hand side. The final inequality is obtained by the assumption regarding C and the estimate $\sum_S \widehat{f}(S)^2 = \mathbb{E}[f^2] \leq \mu$.

Overall: Combining the above two contributions we get

$$\mathrm{Cov}(f(x), f(y)) \leq O\Big(\frac{p}{q}\Big)\mu^2 + \frac{C}{p}\mu,$$

as asserted. □

4.3.5 Bounding $\sum_S M_{p,q,r}(S)$ over 2-dimensional subspaces

In this subsection we present the most complex step of the proof – bounding $\sum_{S \in \mathcal{U}} M_{p,q,r}(S)$ over 2-dimensional subspaces $\mathcal{U}$, which will allow us to complete the proof in conjunction with Lemma 4.20. The proof is quite technical. Its core element is the representation of a sub-problem as a Littlewood-Offord type problem and the use of an antichain technique for handling it.

Lemma 4.21. *Let $S, S' \in \mathbb{Z}_p^r$ be nonzero vectors with $S \neq \nu S'$ for all $\nu \in \mathbb{Z}_p^*$. Suppose $q \leq p$, then*

$$\Big| \sum_{\substack{\mu \in \mathbb{Z}_p^* \\ \nu \in \mathbb{Z}_p}} M_{p,q,r}(\mu S + \nu S') \Big| \leq O(p/q^2 + \log(q)/q + 1/p) \leq O(p \log(q)/q^2). \tag{27}$$

Proof.

Step 1. We express the left hand side of (27), using Lemma 4.17 (with $T = \mu S + \nu S'$), as

$$D := \sum_{\substack{\mu \in \mathbb{Z}_p^* \\ \nu \in \mathbb{Z}_p}} M_{p,q,r}(\mu S + \nu S') = \frac{p^2}{(p-1)^2} \sum_{\substack{\mu \in \mathbb{Z}_p^* \\ \nu \in \mathbb{Z}_p}} \Pr[\langle \mu S + \nu S', x \rangle = \langle \mu S + \nu S', y \rangle = 0] - \frac{p}{p-1}, \tag{28}$$

where the probability is taken over pairs (x, y) distributed according to a (p, q, r)-arithmetic-distribution. Adding and subtracting all pairs of the form $(\mu, \nu) = (0, \nu)$ to the sum in (28), we get

$$\frac{(p-1)^2}{p^2} D = \sum_{\substack{\mu,\nu \in \mathbb{Z}_p \\ (\mu,\nu) \neq (0,0)}} \Pr[\langle \mu S + \nu S', x \rangle = \langle \mu S + \nu S', y \rangle = 0] - (p-1) \Pr_{x,y}[\langle S', x \rangle = \langle S', y \rangle = 0] - \frac{p-1}{p}.$$

Note that given x, y, the number of solutions (μ, ν) of the equation system $(\langle \mu S + \nu S', x \rangle = 0) \wedge (\langle \mu S + \nu S', y \rangle = 0)$, is equal to the number of solutions of the system $(\mu \langle S, x \rangle + \nu \langle S, y \rangle = 0) \wedge (\mu \langle S', x \rangle + \nu \langle S', y \rangle = 0)$. Indeed, they are equal to the sizes of the left kernel and the right kernel of the matrix

$$\begin{pmatrix} \langle S, x \rangle & \langle S, y \rangle \\ \langle S', x \rangle & \langle S', y \rangle \end{pmatrix},$$

which are known to be equal. The latter linear system may succinctly be written as $\mu V(x) + \nu V(y) = 0$ where

$$V(x) := \begin{pmatrix} \langle S, x \rangle \\ \langle S', x \rangle \end{pmatrix} \in \mathbb{Z}_p^2, \qquad V(y) := \begin{pmatrix} \langle S, y \rangle \\ \langle S', y \rangle \end{pmatrix} \in \mathbb{Z}_p^2.$$

Using this equality, we obtain

$$\frac{(p-1)^2}{p^2} D = \sum_{\substack{\mu,\nu \in \mathbb{Z}_p \\ (\mu,\nu) \neq (0,0)}} \Pr_{x,y}[\mu V(x) + \nu V(y) = 0] - (p-1) \Pr_{x,y}[\langle S', x \rangle = \langle S', y \rangle = 0] - \frac{p-1}{p}.$$

Note that (27) is equivalent to $D \leq O(p/q^2 + \log(q)/q + 1/p)$, which follows from the bounds:

$$\Pr_{x,y} \big[\langle S', x \rangle = \langle S', y \rangle = 0 \big] \leq O(1/(pq) + 1/p^2), \tag{29}$$

$$\Big| \sum_{\substack{\mu,\nu \in \mathbb{Z}_p \\ (\mu,\nu) \neq (0,0)}} \Pr_{x,y}[\mu V(x) + \nu V(y) = 0] - \frac{p-1}{p} \Big| \leq O(p/q^2 + \log(q)/q + 1/p). \tag{30}$$

As (29) follows from Lemma 4.18, we are left with the task of verifying (30).

Step 2. We verify (30). Note that $\frac{p^2-1}{p^2} - \frac{p-1}{p} < 1/p$, thus (30) follows from

$$\left| \sum_{(\mu,\nu)\neq(0,0)} \Pr_{x,y}[\mu V(x) + \nu V(y) = 0] - \frac{p^2-1}{p^2} \right| \leq O(p/q^2 + \log(q)/q + 1/p). \tag{31}$$

In case of either $\mu = 0$ or $\nu = 0$, we have $\Pr_{x,y}[\mu V(x) + \nu V(y) = 0] = 1/p^2$, since both x, y are uniformly distributed in $\mathbb{Z}_p^r$ and S, S' are linearly independent vectors. We must hence verify

$$\left| \sum_{\mu\in\mathbb{Z}_p^*} \Pr_{x,y}[\mu V(x) = V(y)] - \frac{p-1}{p^2} \right| \leq O(1/q^2 + \log(q)/(pq) + 1/p^2). \tag{32}$$

We now reason about the left hand side of (32). Specifically, we consider the three sums

$$Q_1 := \sum_{\mu\in\mathbb{Z}_p^*} \Pr_{x,y}\Big[\mu V(x) = V(y) \wedge \Big(\{\langle S, x\rangle, \langle S, y\rangle\} = \{0\}\Big)\Big],$$

$$Q_2 := \sum_{\mu\in\mathbb{Z}_p^*} \Pr_{x,y}\Big[\mu V(x) = V(y) \wedge \Big(0 \in \{\langle S, x\rangle, \langle S, y\rangle\} \neq \{0\}\Big)\Big],$$

$$Q_3 := \sum_{\mu\in\mathbb{Z}_p^*} \Pr_{x,y}\Big[\mu V(x) = V(y) \wedge \Big(0 \notin \{\langle S, x\rangle, \langle S, y\rangle\}\Big)\Big],$$

and show the following estimates, which together imply (32):

$$|Q_1| \leq O(1/(pq) + 1/p^2), \tag{33}$$

$$Q_2 = 0, \tag{34}$$

$$\left|Q_3 - \frac{p-1}{p^2}\right| \leq O(1/q^2 + \log(q)/(pq) + 1/p^2). \tag{35}$$

In order to obtain (34), notice that if one of $\langle S, x\rangle, \langle S, y\rangle$ is zero, and the other is not, then there cannot be a $\mu \in \mathbb{Z}_p^*$ which is the quotient of them.

Step 3. We prove (33). First, we observe

$$Q_1 \leq \Pr_{x,y}\big[\{\langle S, x\rangle, \langle S, y\rangle\} = \{0\}\big] + (p-2)\Pr_{x,y}\big[V(x) = V(y) = 0\big].$$

This is because upon fixing x, y, whenever $\langle S', x\rangle \neq 0$ or $\langle S', y\rangle \neq 0$, there is at most one value of $\mu \in \mathbb{Z}_p^*$ for which $\mu V(x) = V(y)$. Lemma 4.18 implies

$$\Pr_{x,y}\big[\{\langle S, x\rangle, \langle S, y\rangle\} = \{0\}\big] \leq O(1/(pq) + 1/p^2).$$

Hence, it remains to show

$$\Pr_{x,y}\big[V(x) = V(y) = 0\big] \leq O(1/(p^2q) + 1/p^3). \tag{36}$$

Since $x \sim \mathbb{Z}_p^r$ is uniformly distributed, and S, S' are two independent vectors, then we have $\Pr[V(x) = 0] = 1/p^2$. Moreover, similarly to the reasoning in Lemma 4.18 ((25) in particular),

$$\Pr\left[V(y) = 0 \,|\, V(x) = 0\right] = \Pr[\langle S, u\rangle = \langle S', u\rangle = 0],$$

where $u = \lfloor \alpha v_i/q \rceil$ with $\alpha \sim \mathbb{Z}_{pq}^*$ and $v_i \sim \{(1-q)/2, \ldots, (q-1)/2\}$ are uniformly distributed. But, according to (26), we have

$$\Pr[\langle S, u\rangle = \langle S', u\rangle = 0] \leq \Pr[\langle S, u\rangle = 0] \leq O(1/p + 1/q),$$

which, together with $\Pr[V(x) = 0] = 1/p^2$ implies (36).

Step 4. We prove (35). Recall, again, that x, y are sampled by taking $x \sim \mathbb{Z}_p^r$ uniformly at random, and setting $y_i = (\gamma x_i + \gamma' u_i) \,\%\, p$ with $\gamma', \gamma \sim \mathbb{Z}_p^*$ and $u_i = \lfloor \alpha \cdot v_i \rceil$, with $\alpha \sim \mathbb{Z}_{pq}^*$ and $v_i \sim \{(1-q)/2, \ldots, (q-1)/2\}$. We further decompose Q_3 into two parts, $Q_3 = Q_4 + Q_5$, according to whether $\gamma\langle S, x\rangle = \langle S, y\rangle$ or not:

$$Q_4 = \sum_{\mu \in \mathbb{Z}_p^*} \Pr_{x,y}\Big[\mu V(x) = V(y) \wedge \mu \neq \gamma \wedge 0 \notin \{\langle S, x\rangle, \langle S, y\rangle\}\Big],$$

$$Q_5 = \Pr_{x,y}\Big[\gamma V(x) = V(y) \wedge 0 \notin \{\langle S, x\rangle, \langle S, y\rangle\}\Big],$$

and define the auxiliary probabilities

$$\begin{aligned}\beta &= \Pr\left[\gamma\langle S, x\rangle = \langle S, y\rangle \wedge \big(0 \notin \{\langle S, x\rangle, \langle S, y\rangle\}\big)\right],\\ \eta &= \Pr[0 \notin \{\langle S, x\rangle, \langle S, y\rangle\}].\end{aligned}$$

We make four claims:

- **Simplifying Q_4:** $\quad Q_4 = (\eta - \beta)/p,$
- **Upper bounding β:** $\quad \beta \leq O(1/p + 1/q),$
- **Lower bounding η:** $\quad \eta \geq 1 - 2/p,$
- **Upper bounding Q_5:** $\quad Q_5 \leq O(1/q^2 + \log(q)/(pq)).$

Since $Q_3 = Q_4 + Q_5$, these claims clearly imply (35).

Lower bounding η. Recall that both x and y are uniformly distributed in $\mathbb{Z}_p^r$, and so $\langle S, x\rangle$ and $\langle S, y\rangle$ are uniformly distributed in $\mathbb{Z}_p$. Thus, by a union bound, we have $\eta \geq 1 - 2/p$, as asserted.

Upper bounding β. Recall that $y_i = (\gamma x_i + \gamma' u_i) \,\%\, p$, and hence the event that $\gamma\langle S, x\rangle = \langle S, y\rangle$ is exactly the event that $\langle S, u\rangle = 0$. The probability of this latter event may be upper bounded by $O(1/p + 1/q)$ using (26). Hence,

$$\beta \leq \Pr[\langle S, u\rangle = 0] \leq O(1/p + 1/q),$$

as asserted.

Simplifying Q_4. As an appetizer, note that if we would replace in Q_4 the requirement of $\mu V(x) = V(y)$ by $\mu\langle S, x\rangle = \langle S, y\rangle$, and call the result Q_4', then we would get $Q_4' + \beta = \eta$. All that is left in order to prove (1) is to show that $pQ_4 = Q_4'$.

Observe that we may assume that

$$\exists \ell \in [r] : S_\ell = 0 \wedge S_\ell' \neq 0. \tag{37}$$

To reduce to this case, we choose any ℓ with $S'_\ell \neq 0$, and replace S by $S - \frac{S_\ell}{S'_\ell} S'$, which is also nonzero. (Note that the sum on the left hand side of (27) does not change by this replacement.)

We condition on the values of γ, u, γ' and $\{x_j : j \neq \ell\}$ (i.e. on the σ-algebra generated by these variables). The only information that is missing in the probability space is x_ℓ – it is uniformly distributed under the current conditioning. While the contribution to Q'_4 is fixed under the current conditioning (as we assumed $S_\ell = 0$), we claim there exists exactly one value of x_ℓ that would contribute to the probability expressed by Q_4. To see this, let μ be the unique element of $\mathbb{Z}_p^*$ that has $\mu\langle S, x\rangle = \langle S, y\rangle$. In order to have $\mu V(x) = V(y)$ we must also have $\mu\langle S', x\rangle = \langle S', y\rangle$. Under our conditioning, this latter equation is a linear equation in x_ℓ with the linear coefficient $(\mu - \gamma)S'_\ell$ (recall how y_ℓ depends on x_ℓ), and some constant coefficient which is deterministic under our conditioning. This equation has a unique solution in x_ℓ. Since x_ℓ has a uniform distribution, we have $Q_4 = Q'_4/p = (\eta - \beta)/p$, as asserted.

Step 5. Lastly, we upper bound Q_5. Recall

$$V(y) = \begin{pmatrix} \langle S, y\rangle \\ \langle S', y\rangle \end{pmatrix} = \gamma \begin{pmatrix} \langle S, x\rangle \\ \langle S', x\rangle \end{pmatrix} + \gamma' \begin{pmatrix} \langle S, u\rangle \\ \langle S', u\rangle \end{pmatrix}.$$

Hence, the event $\gamma V(x) = V(y)$ is simply $\{\langle S, u\rangle = 0 \wedge \langle S', u\rangle = 0\}$, implying

$$Q_5 \leq \Pr_u[\langle S, u\rangle = 0 \wedge \langle S', u\rangle = 0].$$

We use an argument similar to the argument we had in Lemma 4.18 (specifically, (26)) to upper bound this last quantity. Let $l \in [r]$ be any coordinate with $S_l \neq 0$, and let ℓ be a coordinate with $S'_\ell \neq 0$ while $S_\ell = 0$, whose existence we assumed (see (37)). Recall $u_j = \lfloor \alpha v_j/q \rceil$. Condition on any specific values for $\alpha, \{u_j : j \notin \{l, \ell\}\}$. Similarly to Step 4, we can upper bound the probability that $\langle S, u\rangle = 0$ by the probability that u_l turns out to be just the right value that would make $\langle S, u\rangle = 0$ true. This probability is upper bounded by $\frac{1}{q}$ times the maximal number of elements $v'' \in \{(1-q)/2, \ldots, (q-1)/2\}$ that map to the same u'' under $u'' = \lfloor \alpha v''/q \rceil \,\%\, p$. Using the same reasoning as in Lemma 4.18, we see that this probability is upper bounded by $(|Z_\alpha| + 1)/q$, where

$$Z_\alpha = \{\sigma \in \{(1-q)/2, \ldots, (q-1)/2\} : (\alpha\sigma) \in pq\mathbb{Z} + (-q, q)\}.$$

Under the described conditioning, $\langle S, u\rangle$ is determined by u_l, and by further conditioning on the value of u_l, $\langle S', u\rangle$ is determined by u_ℓ. Thus, the event that both these quantities are equal zero has probability

$$\Pr[\langle S, u\rangle = 0 \wedge \langle S', u\rangle = 0] \leq (|Z_\alpha| + 1)^2/q^2. \tag{38}$$

Our task of upper bounding Q_5 is hence reduced to understanding the second moment of $|Z_\alpha|$. For any fixed α,

$$\begin{aligned} |Z_\alpha|^2 &= \sum_{\sigma_1=(1-q)/2}^{(q-1)/2} \sum_{\sigma_2=(1-q)/2}^{(q-1)/2} \mathbb{1}\big\{\{\sigma_1 \cdot \alpha, \sigma_2 \cdot \alpha\} \subseteq pq\mathbb{Z} + (-q, q)\big\} \\ &\leq 4 \sum_{\sigma_1=0}^{q-1} \sum_{\sigma_2=0}^{q-1} \mathbb{1}\big\{\{\sigma_1 \cdot \alpha, \sigma_2 \cdot \alpha\} \subseteq pq\mathbb{Z} + (-q, q)\big\}. \end{aligned}$$

Denote $\tau'_j := (\sigma_j \cdot \alpha \,\%\, pq)$, then we have $\tau'_1 \cdot \sigma_2 \,\%\, (pq) = \tau'_2 \cdot \sigma_1 \,\%\, (pq)$. Using the assumption $q \le p$, we may leverage this equation into an equation over the integers (i.e. not involving a modulus), in the following way.

Let $\tau_j := \min\{\tau'_j, pq - \tau'_j\}$, so the above equation reads either $\tau_1 \cdot \sigma_2 \,\%\, (pq) = \tau_2 \cdot \sigma_1 \,\%\, (pq)$ or $\tau_1 \cdot \sigma_2 \,\%\, (pq) = -\tau_2 \cdot \sigma_1 \,\%\, (pq)$. Observe that $\tau_j \le q$, and hence, $\tau_1\sigma_2, \tau_2\sigma_1 \le q^2 \le pq$. Therefore, over the integers we must have

$$\tau_1 \cdot \sigma_2 = \tau_2 \cdot \sigma_1 \qquad \text{or} \qquad \tau_1 \cdot \sigma_2 + \tau_2 \cdot \sigma_1 = pq. \tag{39}$$

In order to bound from above the expectation of $|Z_\alpha|^2$ over the $|\mathbb{Z}^*_{pq}| = \phi(pq)$ possible values of α, we note that α can be recovered uniquely from σ_1, τ'_1, as $\alpha = \tau'_1 \cdot \sigma_1^{-1} \,\%\, pq$. Thus, any given quadruple $(\sigma_1, \sigma_2, \tau_1, \tau_2)$ corresponds to at most two different values of α, and so, when we sum over all values of α, each solution of each of the equations in (39) is counted at most twice.

Therefore, we have

$$\mathop{\mathbb{E}}_{\alpha}[|Z_\alpha|^2] \le \frac{4 \cdot 2}{\phi(pq)} \Big(\#\{(a,b,c,d) : ab = cd\} + \#\{(a,b,c,d) : ab + cd = pq\} \Big), \tag{40}$$

where a, b, c, d take values in $\{0, 1, \ldots, q-1\}$.

We bound the number of such quadruples (a, b, c, d) by the following simple number-theoretic lemma, whose proof is given below.

Lemma 4.22. *Let $q, N > 0$ be positive integers. Define*

$$P = \{(a,b,c,d) \,|\, ab + cd = N\} \subseteq \{0, 1, \ldots, q-1\}^4,$$
$$Q = \{(a,b,c,d) \,|\, ab = cd\} \subseteq \{0, 1, \ldots, q-1\}^4.$$

Then, $|P| \le O(q^2 \log(q))$ and $|Q| \le O(q^2 \log(q))$.

By Lemma 4.22, the number of these quadruples (a, b, c, d) is $O(q^2 \log(q))$. Combining (38) and (40) with Lemma 4.22, we arrive at

$$Q_5 \le O\Big(\frac{1}{q^2} \cdot \big(1 + \frac{q^2 \log(q)}{pq}\big)\Big) \le O\Big(\frac{1}{q^2} + \frac{\log(q)}{pq}\Big),$$

concluding the proof of Lemma 4.21.

Summary of the proof of Lemma 4.21. We parsed the left hand side of (27), and interpreted it as the bias introduced in an event, ($V(x)$ is proportional to $V(y)$) caused by dependence between $x, y \sim \mathbb{Z}_p^r$. The core of the argument upper bounds this bias by posing the problem as a Littlewood-Offord-type problem, and using an antichain argument along with simple number theoretic estimates. □

Proof of Lemma 4.22. First, notice that we may consider, in both cases, $a, b, c, d > 0$, as there are only $O(q^2)$ quadruples with $0 \in \{a, b, c, d\}$ and either $ab = cd$ or $ab + cd = N$. Indeed, regarding $ab = cd$, we must have 0 on both sides, which implies that there are only $O(q^2)$ possible pairs. Regarding $ab + cd = N$, if $a = 0$, then b, c have q^2 options, and they determine d, totaling in $\le O(q^2)$ pairs. We call the analogs of P, Q, with the quadruples containing 0 removed, P', Q', respectively.

Second, we count $|P'|$. Fixing a, c, we see that b, d must satisfy the linear equation $ab + cd = N$. The different solutions (b, d) for this equation differ by integral multiples of the vector

$(c/\gcd(a,c), -a/\gcd(a,c))$. Since both b, d are integers in $[1, q)$, the number of such solutions is at most $q\gcd(a,c)/\max(a,c)$. Denoting $g = \gcd(a,c)$ we arrive at

$$|P'| \leq \sum_{g=1}^{q} \sum_{g|a} \sum_{g|c} \frac{qg}{\max(a,c)} \leq \sum_{g=1}^{q} \sum_{g|c} \frac{2c}{g} \cdot \frac{qg}{c} \leq \sum_{g=1}^{q} \sum_{g|c} 2q \leq \sum_{g=1}^{q} 2q^2/g = O(q^2 \log(q)),$$

as required.

Lastly, bounding $|Q'|$ is done likewise, this time considering the equation $ab - cd = 0$.

□

We note that the $\log(q)$ factor in the bound $O\left(\frac{1}{q^2} + \frac{\log(q)}{pq}\right)$ of Lemma 4.22 is the reason for the logarithmic loss in Theorem 4.1. Unfortunately, one can show that the assertion of Lemma 4.22 is tight up to a constant factor, at least regarding the size of Q.

4.3.6 Wrapping up the proof of the obfuscation lemma

Proof of Lemma 4.7. Recall that given an algorithm $B\colon \mathbb{Z}_p^r \to \binom{[r]}{k}$ which always reports a k-tuple of its input numbers whose sum is 0 (and is allowed to report failure), we define $B'\colon \mathbb{Z}_p^r \to [0,1]$ by

$$B'(x) = \Pr_{P,\gamma}[A(P(\gamma \cdot x)) = P(\{1, 2, \ldots, k\})],$$

where $P \sim S_r$ and $\gamma \sim \mathbb{Z}_p^*$, and the probability is taken also over A's internal randomness. By Lemma 4.10, we have

$$\sum_{i=1}^{k} x_i \neq 0 \quad \Longrightarrow \quad B'(x) = 0,$$

and

$$\mu := \mathbb{E}[B'(x)] \leq 1/\binom{r}{k}, \tag{41}$$

and if (x, y) has a (p, q, r)-arithmetic-distribution then

$$\Pr[P^{-1}(A(P(x))) = Q^{-1}(A(Q(y)))] = \binom{r}{k} \mathop{\mathbb{E}}_{x,y}[B'(x)B'(y)]. \tag{42}$$

Furthermore, by Claim 4.12, $B'(x) = B'(\gamma x)$ for all $\gamma \in \mathbb{Z}_p^*$ and $\widehat{B'}(S + S') = \widehat{B'}(S)$ for all S, with S' as defined in Claim 4.12.

Hence, we may apply Lemma 4.20 with

$$C = O(p \log(q)/q^2),$$

as provided by Lemma 4.21 (notice that we assume $p \geq q$), to conclude that

$$\mathrm{Cov}(B'(x), B'(y)) \leq O\Big(\frac{p}{q}\mu^2 + \frac{C}{p}\mu\Big) = O\Big(\frac{p}{q}\mu^2 + \frac{\log(q)}{q^2}\mu\Big).$$

Notice that as both x, y are uniformly distributed in $\mathbb{Z}_p^r$, we have

$$\mathbb{E}[B'(x)B'(y)] = \mu^2 + \mathrm{Cov}(B'(x), B'(y)).$$

Combining these estimates with (41) and (42), we obtain

$$\Pr[P^{-1}(A(P(x))) = Q^{-1}(A(Q(x)))] \leq O\Big(\frac{p}{q\binom{r}{k}} + \frac{\log(q)}{q^2}\Big),$$

completing the proof. □

References

[1] A. Abboud and V. V. Williams, *Popular conjectures imply strong lower bounds for dynamic problems*, in FOCS 2014, IEEE Computer Society, 2014, pp. 434–443.

[2] A. Abboud, V. V. Williams, and O. Weimann, *Consequences of faster alignment of sequences*, in ICALP 2014, J. Esparza, P. Fraigniaud, T. Husfeldt, and E. Koutsoupias, eds., vol. 8572 of Lecture Notes in Computer Science, Springer, 2014, pp. 39–51.

[3] M. Y. An, *Log-concave probability distributions: Theory and statistical testing*, game theory and information, University Library of Munich, Germany, 1996.

[4] B. Aronov and S. Har-Peled, *On approximating the depth and related problems*, SIAM J. Comput., 38 (2008), pp. 899–921.

[5] M. Ball, A. Rosen, M. Sabin, and P. N. Vasudevan, *Proofs of work from worst-case assumptions*, in CRYPTO 2018, H. Shacham and A. Boldyreva, eds., vol. 10991 of Lecture Notes in Computer Science, Springer, 2018, pp. 789–819.

[6] A. Becker, A. Joux, A. May, and A. Meurer, *Decoding random binary linear codes in $2^{n/20}$: How 1 + 1 = 0 improves information set decoding*, in EUROCRYPT 2012, D. Pointcheval and T. Johansson, eds., vol. 7237 of Lecture Notes in Computer Science, Springer, 2012, pp. 520–536.

[7] F. Benhamouda, T. Lepoint, J. Loss, M. Orrù, and M. Raykova, *On the (in)security of ROS*, in EUROCRYPT 2021, A. Canteaut and F. Standaert, eds., vol. 12696 of Lecture Notes in Computer Science, Springer, 2021, pp. 33–53.

[8] A. Blum, A. Kalai, and H. Wasserman, *Noise-tolerant learning, the parity problem, and the statistical query model*, J. ACM, 50 (2003), pp. 506–519.

[9] C. Bouillaguet, C. Delaplace, and P. Fouque, *Revisiting and improving algorithms for the 3xor problem*, IACR Trans. Symmetric Cryptol., 2018 (2018), pp. 254–276.

[10] Z. Brakerski, N. Stephens-Davidowitz, and V. Vaikuntanathan, *On the hardness of average-case k-sum*, in APPROX/RANDOM 2021, M. Wootters and L. Sanità, eds., vol. 207 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, pp. 29:1–29:19.

[11] P. Camion and J. Patarin, *The knapsack hash function proposed at Crypto'89 can be broken*, in EUROCRYPT 1991, D. W. Davies, ed., vol. 547 of Lecture Notes in Computer Science, Springer, 1991, pp. 39–53.

[12] A. Degwekar, V. Vaikuntanathan, and P. N. Vasudevan, *Fine-grained cryptography*, in CRYPTO 2016, M. Robshaw and J. Katz, eds., vol. 9816 of Lecture Notes in Computer Science, Springer, 2016, pp. 533–562.

[13] I. Dinur, *An algorithmic framework for the generalized birthday problem*, Des. Codes Cryptogr., 87 (2019), pp. 1897–1926.

[14] Y. Dodis, D. Khovratovich, N. Mouha, and M. Nandi, *T_5: Hashing five inputs with three compression calls*, in ITC 2021, S. Tessaro, ed., vol. 199 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, pp. 24:1–24:23.

[15] O. Dunkelman, N. Keller, and A. Shamir, *Slidex attacks on the Even-Mansour encryption scheme*, J. Cryptol., 28 (2015), pp. 1–28.

[16] P. Erdős, *On a lemma of Littlewood and Offord*, Bull. Amer. Math. Soc., 5 (1945), pp. 898–902.

[17] A. Gajentaan and M. H. Overmars, *On a class of $O(n^2)$ problems in computational geometry*, Comput. Geom., 5 (1995), pp. 165–185.

[18] A. Golovnev, S. Guo, T. Horel, S. Park, and V. Vaikuntanathan, *Data structures meet cryptography: 3sum with preprocessing*, in STOC 2020, K. Makarychev, Y. Makarychev, M. Tulsiani, G. Kamath, and J. Chuzhoy, eds., ACM, 2020, pp. 294–307.

[19] N. Howgrave-Graham and A. Joux, *New generic algorithms for hard knapsacks*, in EUROCRYPT 2010, H. Gilbert, ed., vol. 6110 of Lecture Notes in Computer Science, Springer, 2010, pp. 235–256.

[20] R. Impagliazzo, L. A. Levin, and M. Luby, *Pseudo-random generation from one-way functions (extended abstracts)*, in STOC 1989, D. S. Johnson, ed., ACM, 1989, pp. 12–24.

[21] Z. Jafargholi and E. Viola, *3sum, 3xor, triangles*, Algorithmica, 74 (2016), pp. 326–343.

[22] A. Joux, *Algorithmic cryptanalysis*, CRC Press, 2009.

[23] R. LaVigne, A. Lincoln, and V. V. Williams, *Public-key cryptography in the fine-grained setting*, in CRYPTO 2019, A. Boldyreva and D. Micciancio, eds., vol. 11694 of Lecture Notes in Computer Science, Springer, 2019, pp. 605–635.

[24] G. Leurent and F. Sibleyras, *Low-memory attacks against two-round even-mansour using the 3-xor problem*, in CRYPTO 2019, A. Boldyreva and D. Micciancio, eds., vol. 11693 of Lecture Notes in Computer Science, Springer, 2019, pp. 210–235.

[25] J. E. Littlewood and A. C. Offord, *On the number of real roots of a random algebraic equation (III)*, Rec. Math. (Mat. Sbornik). Nouvelle Série, 54 (1943), pp. 277–286.

[26] Y. Lu and S. Vaudenay, *Faster correlation attack on bluetooth keystream generator E0*, in CRYPTO 2004, M. K. Franklin, ed., vol. 3152 of Lecture Notes in Computer Science, Springer, 2004, pp. 407–425.

[27] V. Lyubashevsky, *The parity problem in the presence of noise, decoding random linear codes, and the subset sum problem*, in APPROX/RANDOM 2005, C. Chekuri, K. Jansen, J. D. P. Rolim, and L. Trevisan, eds., vol. 3624 of Lecture Notes in Computer Science, Springer, 2005, pp. 378–389.

[28] F. Mendel, N. Pramstaller, C. Rechberger, M. Kontak, and J. Szmidt, *Cryptanalysis of the GOST hash function*, in CRYPTO 2008, D. A. Wagner, ed., vol. 5157 of Lecture Notes in Computer Science, Springer, 2008, pp. 162–178.

[29] L. Minder and A. Sinclair, *The extended k-tree algorithm*, J. Cryptol., 25 (2012), pp. 349–382.

[30] J. Nederlof and K. Wegrzycki, *Improving Schroeppel and Shamir's algorithm for subset sum via orthogonal vectors*, in STOC 2021, S. Khuller and V. V. Williams, eds., ACM, 2021, pp. 1670–1683.

[31] I. Nikolic and Y. Sasaki, *Refinements of the k-tree algorithm for the generalized birthday problem*, in ASIACRYPT 2015, T. Iwata and J. H. Cheon, eds., vol. 9453 of Lecture Notes in Computer Science, Springer, 2015, pp. 683–703.

[32] M. Patrascu, *Towards polynomial lower bounds for dynamic problems*, in STOC 2010, L. J. Schulman, ed., ACM, 2010, pp. 603–610.

[33] S. Pettie, *Higher lower bounds from the 3sum conjecture*. Fine-Grained Complexity and Algorithm Design Workshop at the Simons Institute, 2015.

[34] V. Vasillevska-Williams, *On some fine-grained questions in algorithms and complexity*, Proceedings of the International Congress of Mathematicians (ICM), 2018 (2019), pp. 3447–3487.

[35] D. A. Wagner, *A generalized birthday problem*, in CRYPTO 2002, M. Yung, ed., vol. 2442 of Lecture Notes in Computer Science, Springer, 2002, pp. 288–303.

[36] V. V. Williams and R. Williams, *Finding, minimizing, and counting weighted subgraphs*, SIAM J. Comput., 42 (2013), pp. 831–854.

A Wagner's k-tree Algorithm

In this appendix we sketch the details of Wagner's k-tree algorithm for solving the k-XOR problem and its generalization published in [29]. The variant for solving k-SUM is similar. For more details, we refer the reader to the original publications [29, 35].

A.1 The 4-XOR algorithm

We begin by describing the algorithm applied to a 4-list variant of 4-XOR. In this problem, the input consists of 4 lists $\{x^{(j)}\}_{j=1}^{4}$, where each $x^{(j)} \in \{0,1\}^{2^{n/3} \times n}$ is chosen uniformly at random. The goal is to find 4 vectors, one from each list, whose XOR is 0_n, namely, output a 4-tuple $\{i_j\}_{j=1}^{4}$, where $i_j \in [2^{n/3}]$ such that $\bigoplus_{j=1}^{4} x_{i_j}^{(j)} = 0_n$. It is easy to see the 4-list variant is equivalent to the single-list variant (Definition 1.3) up to $O(1)$ factors in success probability and complexity.

The k-tree algorithm for $k = 4$ is described below.

1. Sort the lists $\{x^{(j)}\}_{j=1}^{4}$.

2. By a linear scan, find all pairs $(x_{i_1}^{(1)}, x_{i_2}^{(2)})$ such that the $n/3$ most significant bits of $x_{i_1}^{(1)} \oplus x_{i_2}^{(2)}$ are zero. Store all values $x_{i_1}^{(1)} \oplus x_{i_2}^{(2)}$ in a new sorted list $y^{(1)}$, along with the corresponding pair $(x_{i_1}^{(1)}, x_{i_2}^{(2)})$.

3. Apply the previous step to $x^{(3)}$ and $x^{(4)}$ and build the sorted list $y^{(2)}$.

4. Find a pair $(y_{j_1}^{(1)}, y_{j_2}^{(2)})$ such that $y_{j_1}^{(1)} \oplus y_{j_2}^{(2)} = 0_n$. Trace $(y_{j_1}^{(1)}, y_{j_2}^{(2)})$ back to a solution to 4-XOR problem and output it.

To analyze the algorithm, note that the expected size of $y^{(1)}$ and $y^{(2)}$ is $2^{n/3}$ (as a pair $(x_{i_1}^{(1)}, x_{i_2}^{(2)})$ is added to $y^{(1)}$ with probability $2^{-n/3}$). Therefore, the algorithm runs in expected time $\tilde{O}(2^{n/3})$. Moreover, on average, there is a single 4-XOR solution to be found in the last step, since any 4-tuple $\{x_{i_j}^{(j)}\}_{j=1}^{4}$ satisfies the $4n/3$ bit constraints imposed by the algorithm with probability $2^{-4n/3}$ (and there are $2^{4n/3}$ such 4-tuples). Based on tail bounds, one can show that the algorithm succeeds with constant probability. We refer the reader to [29] for a rigorous analysis.

A.2 Generalizations

We briefly summarize two important generalizations of the 4-XOR algorithm.

A.2.1 The full k-tree algorithm [35]

The first generalization applies to larger k that is a power of 2. The input consists of k lists, each containing $2^{n/(\log k+1)}$ vectors of n bits. The algorithm merges the k lists in pairs in a tree-like structure with $\log k + 1$ levels. The merging maintains the property that the vectors in all $k/2^{\ell}$ lists in level $\ell \in \{0, 1, \ldots, \log k - 1\}$ have zero $\ell \cdot n/(\log k + 1)$ most significant bits. The final 2-list merge at level $\log k - 1$ zeroes the remaining $2n/(\log k + 1)$ bits, giving a k-XOR solution at the last level with high probability.

When k is not a power of 2, the k-XOR problem can be easily reduced to a k'-XOR problem where k' is the largest power of 2 that is smaller than k.

A.2.2 The extended k-tree algorithm [29]

This generalization applies when the input lists contain less than $2^{n/(\log k+1)}$ vectors (i.e., the input is less dense) and the k-tree algorithm is not directly applicable. The extended algorithm gives a tradeoff between the size of the inputs lists and the time complexity.

Specifically, for 4-XOR, when the input lists are of size r for $2^{n/4} \leq r \leq 2^{n/3}$, we change the second step to find all pairs $(x_{i_1}^{(1)}, x_{i_2}^{(2)})$ such that the $4 \log r - n$ most significant bits of $x_{i_1}^{(1)} \oplus x_{i_2}^{(2)}$ are equal to $0_{4 \log r - 1}$ (we also change the third step similarly). Therefore, the expected size of $y^{(1)}$ and $y^{(2)}$ becomes $2^n/r^2$, and the expected complexity of the algorithm is $\tilde{O}(2^n/r^2)$. Finally, on average, there is a single 4-XOR solution to be found in the last step, since there are $4 \log r$ bit constraints imposed by the algorithm on r^4 4-tuples (once again, a tail bound is required to rigorously compute the success probability).