Breaking Fine-Grained Cryptography (and How We Might Fix It)

Applied cryptography final project report

Jakob Nogler, Nathan Sheffield, and Zoe Xi

Spring 2026

Abstract

The goal of *fine-grained* cryptography is to construct cryptographic primitives secure against *fixed-polynomial*-time adversaries, under weaker assumptions than would be needed for security against arbitrary polynomial-time adversaries. One might hope to base this security on the hardness of central problems from fine-grained complexity, such as the k-SUM or k-Orthogonal-Vectors problems. A recent paper of Alman, Huang, and Yeo (Eurocrypt 2025), however, observed what could be considered a barrier to this goal: assuming the non-existence of one-way functions, the average-case k-SUM hypothesis is false.

We give a very simple presentation of Alman, Huang, and Yeo's main result. We then show that their argument can be directly extended to give algorithms (conditional on the non-existence of OWFs) for a large class of other problems, including the k-Orthogonal-Vectors problem. As a final note, we briefly discuss a couple of (failed) attempts we made to construct fine-grained one-way functions from natural worst-case hardness assumptions.

1 Introduction

The security and functionality of modern life rely heavily on the integrity of cryptographic primitives and the formal guarantees they provide. However, the uncomfortable fact is that the vast majority of these cryptographic constructions rely on unproven assumptions. In fact, almost all current systems rely on computational assumptions, with the conjecture that $\mathsf{P} \neq \mathsf{NP}$ serving as a baseline requirement. However, it remains a wide-open question whether $\mathsf{P} \neq \mathsf{NP}$ is even sufficient to construct the most fundamental cryptographic building blocks, such as one-way functions. For practical constructions, the $\mathsf{P} \neq \mathsf{NP}$ gap is rarely enough; instead, cryptographers must assume the existence of one-way functions or the hardness of specific problems like integer factoring and discrete logarithms. A natural question that arises is the following:

Question 1: Can we build cryptography based on weaker assumptions?

This question is important not only to make cryptographic primitives more believable, but also because such constructions can serve as an answer to the following additional question:

Question 2: Is meaningful cryptography possible in a scenario where some of the stronger assumptions fail?

Constructing new assumptions. If the goal is to base constructions on weaker assumptions, it is clear that we may only be able to achieve weaker primitives. Thus, we must relax current definitions to see what is achievable. In modern cryptography, primitives typically aim for security against super-polynomial time

adversaries. One straightforward way to relax these notions is to target only polynomial-time gaps between adversaries and honest parties. That is, to obtain security against adversaries with a polynomial running time $O(n^c)$ for a fixed constant c. This distinction could even enable the construction of primitives in a world where $\mathsf{P} = \mathsf{NP}$, as it is conceivable that security is still guaranteed if c is large enough. (Ideally, one would like to construct such a primitive for every fixed c, allowing us to choose the constant that best fits the application.)

Fine-grained complexity. To achieve security against polynomial-time adversaries, we must find ways to either prove unconditional hardness or rely on assumptions that provide this guarantee. The former, however, seems far from achievable; even for NP-complete problems, we do not know how to construct unconditional super-linear lower bounds, meaning we must once again rely on some form of assumption. Consequently, for computational hardness within P, we also rely on conjectures; the study of these is known as *fine-grained complexity* (see [WIL] for a comprehensive survey).

We highlight two central conjectured hard problems in fine-grained complexity, mentioned here because they play a major role in this write-up. The first is k-SUM: given k sets $S_1, \ldots, S_k$ of n numbers each (where we can assume the universe $U = \text{poly}(n)$ for a sufficiently large polynomial), the goal is to determine if there exist elements $s_1 \in S_1, \ldots, s_k \in S_k$ such that $s_1 + \ldots + s_k = 0$. The conjecture states that no algorithm can solve k-SUM in time $O(n^{\lceil k/2 \rceil - \varepsilon})$ for any $\varepsilon > 0$; that is, we cannot do significantly better than a simple meet-in-the-middle approach. Notably, k-SUM is attractive because the conjecture is thought to hold for any constant k, making it suitable for designing cryptographic primitives where we can specifically choose the polynomial exponent.

The second problem is Orthogonal Vectors (OV): given two sets S_1, S_2 of n vectors each in $\{0,1\}^d$, the goal is to find two vectors $s_1 \in S_1$ and $s_2 \in S_2$ that are orthogonal, such that $\langle s_1, s_2 \rangle = 0$. The conjecture (which is implied by the Strong Exponential Time Hypothesis) states that for every $\varepsilon > 0$, there exists a $c \geq 1$ such that this problem cannot be solved in time $O(n^{2-\varepsilon})$ for $d = c \log n$; essentially, we cannot do better than naively checking all pairs for orthogonality. OV itself has a generalization, k-OV, which allows us to derive hardness for any constant k (again implied by the Strong Exponential Time Hypothesis). Instead of two sets, we are given k sets $S_1, \ldots, S_k$ of n vectors each, and the goal is to find k vectors $s_1 \in S_1, \ldots, s_k \in S_k$ such that for each coordinate $i \in \{1, \ldots, d\}$, the product $s_1[i] \cdot s_2[i] \cdot \cdots \cdot s_k[i] = 0$. Note that for $k = 2$, this definition corresponds exactly to the standard orthogonality condition. The hardness of k-OV is conjectured to be $n^{k-o(1)}$.

Fine-grained cyptography. The main hurdle when constructing crypto primitives out of the conjectured hard problem from fine grained complexity, such as k-OV and k-SUM, is that fine-grained complexity is built on worst-case conjectures, while cryptography demands average-case hardness.

The first line of work to introduce cryptography based on fine-grained complexity (the so-called *fine-grained cryptography*) was by Ball et al. [BRSV17; BRSV18], who developed average-case hard distributions based on worst-case assumptions like OV, APSP, and 3-SUM. Unfortunately, very little is known about what can be built from these specific distributions (among the few for instance there is permissionless consensus [BGHKP24]). In fact, even more significantly, the same authors showed that building fine-grained one-way functions (OWFs) or key-exchange protocols from these distributions would break other believed conjectures, such as NCSETH (a weaker version of the Strong Exponential Time Hypothesis mentioned earlier).

In a second line of work, the average-case hardness of fine-grained problems in specific distributions is not proven from the worst case, but is instead conjectured directly. It turns out that by assuming the average-case hardness of k-SUM and Zero-k-Clique, it is possible to construct fine-grained OWFs and key-exchange protocols [LLV19; DLW20]. While this achieves the construction goal, the average-case hardness of these problems is far less understood than their worst-case counterparts, which have been studied extensively. Consequently, this approach does not fully address Question 1 regarding the construction of primitives based on more widely believed hypotheses. This adds one more open problem we wish to address: namely, whether we can achieve fine-grained cryptography without compromising the worst-case hardness of the

underlying problems or the expressiveness of the primitives we can build.

Question 3: Can we construct the primary fine-grained cryptographic primitives based on worst-case fine-grained assumptions?

Recent developments in fine-grained cryptography and open questions. In very recent work, Alman et al. [AHY25] show that average-case k-SUM and Zero-k-Clique suffer from a further limitation beyond their lack of a proven connection to worst-case hardness. Specifically, they demonstrate that if OWFs do not exist, then average-case k-SUM and Zero-k-Clique must also fail. This implies that the primitives constructed from these problems do not provide a satisfactory answer to Question 2.

Consequently, the findings of Alman et al. [AHY25] represent a step back in the quest to answer Question 2, requiring us to find new ways to pursue the tantalizing goals described so far. In another concurrent work, Künemann et al. [KPR25] propose a distribution for k-OV that is plausible to be hard in the average case. Such construction that has remained elusive until now because k-OV is easy to solve under some natural distributions and parameter choices [KW19]. This k-OV distribution (henceforth called AVG-k-OV) possesses many desirable properties, offering hope for constructing primitives that do not suffer from the issues identified by Alman et al. [AHY25] for k-SUM and Zero-k-Clique.

However, since these two works are nearly concurrent, there has been little exploration into whether AVG-k-OV suffers from the same problem. The authors of [KPR25] only noted that it "might be possible to generalize [Alman et al.'s] results to AVG-k-OV," leaving the actual status of the problem open. This brings us to our final and most specific question.

Question 4: Does the result of [AHY25] extend to AVG-k-OV?

Our contributions

In this project, we resolve the more specific Question 4 and attempt to make partial progress toward answering the highly tantalizing Questions 1, 2, and 3. As a first step, we provide a simple presentation of the main result by Alman, Huang, and Yeo [AHY25]. This simplified presentation allows us to observe that their argument can be directly extended to provide algorithms (conditional on the non-existence of OWFs) for a broad class of other problems, which informally reads as follows:

Theorem 1.1 (Informal)**.** Consider a problem and a distribution $\mathcal{D}$ of instances for which there exists an appropriate "planting algorithm" satisfying certain conditions that takes an instance from $\mathcal{D}$ along with information on where to "plant" a solution. If OWFs do not exist, then there exists some constant c such that the problem can be solved on $\mathcal{D}$ in time $O(n^c)$.

In particular, for problems such as AVG-k-SUM that possess such a planting algorithm, we find that the conjecture for AVG-k-SUM breaks for $k > c$. By rephrasing the distribution for AVG-k-OV and identifying an appropriate planting algorithm, we resolve Question 4.

Corollary 1.2. If one-way functions do not exist, then AVG-k-OV is false.

As a final note, we attempt to address the more general Questions 1, 2, and 3. Among these approaches, we explore basing fine-grained cryptography on the hardness of matrix multiplication; this represents a promising possibility toward answering Question 3, as worst-case to average-case reductions already exist for matrix multiplication [AGGS22; HS23; GSS24; HS25b]. Furthermore, we discuss the possibility of constructing fine-grained primitives based on circuit lower bounds.

Organization of this write-up. In Section 2, we introduce the notations and formalize the fine-grained conjectures used throughout this work, specifically AVG-k-SUM and AVG-k-OV. In Section 3, we present a simplified treatment of the results by Alman et al. [AHY25]. Building on this, Section 4 explores how these

results can be extended to other problems and distributions. Finally, in Section 5, we give our attempts to construct new fine-grained cryptographic primitives.

2 Preliminaries

The problem AVG-k-SUM. We define more formally AVG-k-SUM.

Consider the group ensemble $\{\mathbb{Z}_{m(n)}\}_{n\in\mathbb{N}}$, i.e., the additive group of integers modulo $m(n)$. Define the input distribution associated with $\{\mathbb{Z}_{m(n)}\}_{n\in\mathbb{N}}$ as follows: the inputs are k lists $L_1,\ldots,L_k$ each consisting of n elements, where each element is sampled uniformly at random from $\mathbb{Z}_{m(n)}$.

Definition 2.1 (k-SUM, [AHY25])**.** For $k \geq 0$, on input k lists $L_1,\ldots,L_k$, where each list consists of n elements from $\mathbb{Z}_{m(n)}$, we write that an algorithm A solves k-SUM if A satisfies the following:

- If there exists a solution $y_1,\ldots,y_k \in [n]$ such that $L_1[y_1]+\cdots L_k[y_k]=0$, then A outputs any solution $y_1,\ldots,y_k$.
- If there does not exist a solution, then A outputs $\perp$.

We write that A has error probability $\gamma(n)$ if $\Pr[A(L_1,\ldots,L_k) \text{ is correct}] \geq 1-\gamma(n)$, where the randomness is over the internal randomness of A and the random choice of $L_1,\ldots,L_k$ sampled from the input distribution.

Then the conjecture that k-SUM is hard on average can be stated as follows.

Conjecture 2.2 (AVG-k-SUM, [AHY25])**.** For any integer $k \geq 3$, group size $m=\Theta(n^k)$, and constant $\varepsilon>0$, there is no $O(n^{\lceil k/2\rceil-\varepsilon})$-time algorithm that solves the k-SUM search problem on average with error probability $1/\mathsf{poly}(n)$.

The problem AVG-k-OV. We also define more formally AVG-k-OV.

Definition 2.3 (Distribution for AVG-k-OV)**.** Let $k \geq 2$, $d=\alpha(n)\log n$ for $\alpha(n)=\omega(1)$ be parameters. Then, the following distribution generates instance for k-OV:

i) Generate k matrices $U_1,\ldots,U_k \in \{0,1\}^{n\times d}$ where all entries are i.i.d. p-biased bits with probability

$$p \coloneqq \left(1-2^{\frac{-2k}{\alpha(n)}}\right).$$

ii) Sample $(s_1,\ldots,s_k)$ uniformly at random from $[n]^k$.

iii) For each coordinate $1 \leq j \leq d$ do the following: let m be the number of ones among $U_{1,s_1}[j],\ldots,U_{k,s_k}[k]$. If $k-m$ is even, flip $U_{k,s_k}[j]$ with probability $\left(\frac{p}{1-p}\right)^{k-m}$.

iv) Return $U_1,\ldots,U_k$.

At this point, we provide several remarks regarding Definition 2.3 to better explain how it yields a distribution for k-OV. First, note that the sets $S_1,\ldots,S_k$ described in the standard presentation of k-OV are represented here by matrices, where the i-th row of U_i corresponds to the i-th vector of S_i. After the first step of Definition 2.3, the authors of [KPR25] show that $U_1,\ldots,U_k$ contain no solution with high probability. For the indices $(s_1,\ldots,s_k)$ sampled in the second step, notice that the modification in step three guarantees that for each coordinate $1 \leq j \leq d$, at least one of the entries $U_{1,s_1}[j],\ldots,U_{k,s_k}[j]$ is zero. This effectively means that the returned instance is always a yes instance.

We define AVG-k-OV as the problem of solving k-OV under the distribution specified in Definition 2.3. Here "solving k-OV" is defined as follows:

Definition 2.4. For $k \geq 2$ and $d \geq 1$, on input k matrices $U_1,\ldots,U_k \in \{0,1\}^{n\times d}$, we write that algorithm A solves k-OV if A satisfies the following:

- If there exists a solution $y_1, \ldots, y_k \in [n]$ such that for each $j \in [d]$ we have $U_{1,y_1}[j] \cdot U_{2,y_2}[j] \cdot \ldots, \cdot U_{k,y_k}[j] = 0$, then A outputs any solution $y_1, \ldots, y_k$.
- If there does not exists a solituion, then A outputs $\perp$.

We write that A has error probability $\gamma(n)$ if $\Pr[A(L_1, \ldots, L_k) \text{ is correct}] \geq 1 - \gamma(n)$, where the randomness is over the internal randomness of A and the random choice of $L_1, \ldots, L_k$ sampled from the input distribution.

The conjecture relative to Definition 2.3 given by the authors of AVG-k-OV reads as follows:

Conjecture 2.5 ([KPR25])**.** For any $\alpha(n) = \omega(1)$ and $\varepsilon > 0$, there exists no algorithm that solve AVG-k-OV with any failure probability $\gamma(n) < 1/2$ in time $O(n^{k-\varepsilon})$.

Since it may be difficult to grasp why AVG-k-OV seems hard to solve (perhaps because of odd phrasing of the distribution), we highlight one of its key properties: for any subset of indices $C \subsetneq \{1, \ldots, k\}$ with $|C| \leq k-1$, the joint distribution of $\{U_i\}_{i \in C}$ is identical to that of $|C|$ matrices with entries sampled as i.i.d. p-biased coins. In other words, if one does not examine all k sets simultaneously, the matrices appear to be fully random and, with high probability, contain no solution.

3 An alternative presentation of [AHY25]

Recall our main question (Question 2): is meaningful cryptography possible in a scenario where some of the stronger assumptions fail? Or more concretely, what happens to the assumptions that underlie fine-grained cryptography in a world where standard cryptographic assumptions are false? Given that we are mainly interested in fine-grained cryptography in a world where we cannot use standard cryptography (for otherwise standard cryptography provides stronger security guarantees), this is a natural question to study.

In a recent work, Alman, Huang, and Yeo [AHY25] give a negative answer to this question: they show as their main result that if one-way functions (OWFs) do not exist, then two assumptions (about the average-case hardness of two central problems in fine-grained complexity, k-SUM and Zero-k-Clique) that many fine-grained cryptography constructions (e.g., [LLV19], [DLW20]) have been based on are false.

In this section, we give an overview of their result that if OWFs do not exist, then k-SUM is not hard on average. We later use their proof idea of constructing a "planting algorithm" for k-SUM to prove analogous results for a large body of problems, including k-OV.

3.1 If k-SUM is hard on average, then OWFs exist

We focus on Conjecture 2.2. Alman, Huang, and Yeo show the following.

Theorem 3.1 (Theorem 1.1 (Informal), [AHY25])**.** If one-way functions do not exist, then AVG-k-SUM is false.

We give a high-level sketch of their proof. We show the contrapositive: assume that AVG-k-SUM holds, and our goal is to show that one-way functions exist. The main idea is to construct an appropriate "planting algorithm" for k-SUM, which is a linear time algorithm that takes as input an instance of k-SUM and indices at which to plant a solution, plants a solution at these indices, and outputs this new instance with the planted solution. Alman, Huang, and Yeo show that the output distribution of this planting algorithm when the input is a random string is statistically close to the uniform distribution on k-SUM instances conditional on the instance having a solution. Intuitively, this implies that if we are able to invert the planting algorithm on a string sampled from this output distribution with negligible error probability in time $O(n^c)$ for some constant $0 < c \leq \lfloor k/2 \rfloor - \Theta(1)$, then k-SUM is not hard on average. Since we assume AVG-k-SUM holds, this implies that for all $c > 0$, there is some function that is computable in $O(n)$ time and not invertible in $O(n^c)$ time. Now this looks almost like the definition of a OWF, but is not quite: for a OWF, we would like to swap the quantifiers, i.e., we would like a fixed function f that is computable in $O(n)$ time and not

invertible in $O(n^c)$ time for any $c > 0$. But it turns out that we can obtain a OWF from what we have using a "universal one-way function," [Gol04] which simulates all programs running in time $O(n)$.

In more detail, consider the following planting algorithm for k-SUM, which chooses one index in each of the k lists, chooses values for all the numbers in all the lists except for the number at the index in the last list, and sets this number to be the negative of the sum of the numbers at the other k indices: given indices $y_1, \dots, y_k \in [n]$ and a k-SUM instance $(L_1, \dots, L_{k-1}, L'_k)$, where each L_i is a list of n elements in $\mathbb{Z}_m$ and L'_k is a list of $n-1$ elements in $\mathbb{Z}_m$, output k-SUM instance $(L_1, \dots, L_{k-1}, L^*_k)$, where L^*_k is L'_k with $-(L_1[y_1] + \cdots L_{k-1}[y_{k-1}])$ inserted at index y_k. The output has a solution at indices $(y_1, \dots, y_k)$, and every k-SUM instance with a solution is a possible output of the algorithm, or in other words the algorithm is surjective onto the set of k-SUM instances with solutions. The algorithm also runs in time linear in the size of the input $N \coloneqq (kn-1)\log m$.

It turns out that it suffices to consider the case $m \approx n^k$, since if $m \gg n^k$ then there is with high probability no solution, and if $m \ll n^k$, then we can ignore some of the numbers and there will still with high probability be a solution, bringing us back to the $m \approx n^k$ case. In this case, the planting algorithm is expanding, meaning it maps input strings to output strings of longer length. It turns out that this expanding property together with the property that the algorithm is surjective onto the set of instances with solutions imply that the output distribution of the algorithm when the input string is sampled uniformly at random, which we call the planted distribution, is close in total variation distance to the uniform distribution on k-SUM instances that have a solution. Specifically, it is close in the sense that any instance of k-SUM with a solution appears with probability at least $1/N^{100}|\mathcal{P}|$, where $\mathcal{P}$ is the set of all k-SUM instances. This means that if there is an algorithm that inverts a string sampled from the planted distribution with negligible error probability negl, then there is an algorithm that solves uniformly random k-SUM instances with error probability at most $N^{100} \cdot \mathsf{negl}$, which is still negligible.

Since we assume that AVG-k-SUM holds, this implies that there is no such inverter algorithm that runs in time $O(n^c)$ for any constant $0 < c \le \lfloor k/2 \rfloor - \Theta(1)$, which in turn implies that for all $d > 0$, there is some function that is computable in $O(N)$ time and not invertible in $O(N^d)$ time. Then a universal one-way function, which at a high level simulates all $O(N)$-time programs by diagonalizing over them, is a OWF, that is, it is a fixed function computable in time polynomial in N that is not invertible in time $O(N^d)$ for any d, and we are done.

Notice that this proof sketch did not rely on anything so specific to k-SUM: the key step was to design an expanding and surjective planting algorithm. In the next section, we apply this idea of designing appropriate planting algorithms to show analogous results for a large class of other problems that share a similar structure to k-SUM, including k-OV.

4 Extending to other problems

The average-case k-SUM problem follows a simple structure: we're given a large number of independent random elements, and want to know if there exist k of them satisfying some relation. Formally:

Definition 4.1. For a universe U, a distribution $\mathcal{D}$, and a relation $R \subseteq U^k$, define $(R, \mathcal{D})$-SEARCH to be the problem of, given $(x_1, \dots, x_n) \leftarrow \mathcal{D}^n$, returning $i_1, \dots, i_k$ such that $(x_{i_1}, \dots, x_{i_k}) \in R$, if any such tuple exists.

In the case of average-case k-SUM, the distribution is uniform over some finite field, and the relation consists of all tuples that sum to the target value. We observe now that the techniques of [AHY25] directly extend to a large class of such problems.

Definition 4.2. Say that a problem $(R, \mathcal{D})$-SEARCH is ***concentrated*** if, for any n such that

$$\mathbb{E}_{x_1, \dots, x_n}[\#\text{ of tuples } (x_{i_1}, \dots, x_{i_k}) \in R] \ge \Omega(1),$$

we also have

$$\Pr_{x_1,\ldots,x_n}[\exists \text{ tuple } (x_{i_1},\ldots,x_{i_k}) \in R] \geq 1/\mathsf{polylog}(n).$$

Definition 4.3. Say that a problem $(R,\mathcal{D})$-SEARCH is ***satisfier-samplable*** if there exists an $n^{1+o(1)}$-time randomized algorithm which outputs a distribution negligibly close in total-variation distance to the conditional distribution

$$(x_1,\ldots,x_k) \sim \mathcal{D} \mid (x_1,\ldots,x_k) \in R.$$

Theorem 4.4. If one-way functions don't exist, then every concentrated, satisfier-samplable $(R,\mathcal{D})$-SEARCH problem can be solved in $O(n^c)$ time for some fixed c independent of k.

Proof. Recall that, by the universal OWF argument, the non-existence of OWFs implies that there is some fixed c such that every one-way function computable in near-linear time can be inverted (with negligible failure probability) in $O(n^c)$ time. We design a linear-time sampler such that solving $(R,\mathcal{D})$-SEARCH on average reduces to inverting this sampler.

First, observe that if n' is such that there are more than $\mathsf{polylog}(n)$-many R-satisfying k-tuples in expectation, then, by the fact that the $(R,\mathcal{D})$-SEARCH problem is concentrated, there must with high probability exist at least one such tuple among $x_1,\ldots,x_{n'}$. So, we can discard the remaining x_i, and still whp have a solution. Thus, we consider without loss of generality the case that there are at most $\log^{c'}(n)$ many R-satisfying k-tuples in expectation for some fixed constant c'.

Now, we design a sampler. First, we sample k random indices $i_1,\ldots,i_k$. Then, we sample values $x_{i_1},\ldots,x_{i_k}$ such that $(x_{i_1},\ldots,x_{i_k}) \in R$ using the satisfier-sampler. Finally, we sample each x_j for $j \in [n] \setminus \{i_1,\ldots,i_k\}$ independently from $\mathcal{D}$, and output the resulting list.

Observe that we can invert this sampler with negligible failure probability. In order to show that this solves the $(R,\mathcal{D})$-SEARCH problem with negligible failure probability, it suffices to show that that, for any instance x with a solution, the probability of x under $\mathcal{D}^n$ is at most a fixed polynomial factor larger than the probability of x under the output of this sampler. But this now follows from a simple counting argument. First, for simplicity, let us assume that $\mathcal{D}$ is the uniform distribution over U (this follows without loss of generality by padding U with the bits used in sampling it — note that we don't need to feed these bits to the algorithm, this is just for analysis). So, each instance has probability exactly $|U|^{-n}$ under $\mathcal{D}^n$. But now, observe that every instance with a solution is at least a *possible* output of the sampler, so it suffices to lower bound the probability of any given possible output of the sampler being output. If an instance has a solution, then there is at least a $1/n^k$ chance that the sampler correctly chooses the indices of some such solution. Then, there is a $|U|^{k-n}$ chance that the sampler correctly chooses the values x_j for all of the other indices correctly. Finally, observe that, since the overall expected number of satisfying k-tuples was at most $\log^{c'}(n)$, there must be at most $|U|^k \log(c')/n^k$ many satisfying k-tuples, meaning that the probability of choosing all of these correctly is at least $n^k/|U|^k \log(c')$. Overall, this gives an output probability of at least $|U|^{-n}/\log(c')$, which is only a $\log(c')$ factor smaller than the probability under $\mathcal{D}^n$. ■

This applies to a very large class of problems — for instance, k-OV:

Proposition 4.5. AVG-k-OV is concentrated and satisfier-samplable.

Proof. For concentration, we use the second moment method (Paley–Zygmund inequality), observing that $\Pr[\text{exists solution}] \geq \frac{\mathbb{E}[\#\text{ solutions}]^2}{\mathbb{E}[\#\text{ solutions}^2]}$. It is straightforward to bound $\mathbb{E}[\#\text{ solutions}^2]$ by considering pairs of k-tuples and bounding the probability of both jointly being solutions in terms of the size of their intersection.

For satisfier-samplability, note that conditional a set of k vectors from the distribution having generalized-inner product 0, the values of the vectors for each coordinate are still indepedent of each other. So we

can sample each coordinate separately, and it is easy to sample from any distribution supported on only a constant number of values. ■

We note that the concentration requirement is satisfied by quite a large class of relations and distributions, as evidenced by the Park–Pham theorem. The satisfier-samplability requirement seems more restrictive, but unclear how to remove in general. (And indeed some version of this requirement ought to be necessary, given that any problem we can solve must at least belong to search-NP.)

5 Some half-baked approaches to constructing fine-grained OWFs

The above argument rules out some kind of fine-grained one-way function in general. In particular, the observation about universal one-way functions means that, if standard OWFs don't exist, then there must be some c such that we can't get security against time-n^c adversaries with a linear-time computable function. However, we could still hope to get *some* kind of fine-grained security. That is, we'd just like the following:

Definition 5.1. Say that a function f is a ***non-trivial*** fine-grained one-way function if, for some constants $c < c'$, it is computable in time $O(n^c)$, but no time-$O(n^{c'})$ algorithm can achieve constant probability of inverting it.

The question now becomes, what reasonable assumptions could we hope to base such functions on in a world where standard cryptography fails? As our intuition for believing, e.g., the average-case k-OV or k-SUM conjectures doesn't seem to depend on k, these seem potentially suspect foundation — we know that in a world without standard OWFs these have to fail for *some* k, so we'd like some more justification to believe that they would nonetheless hold for small enough k.

A pipe dream would be to base non-trivial fine-grained one-way functions purely on the *worst-case* fine-grained hardness of well-studied algorithmic problems. This is a major open problem, and we do not resolve it in this project, but we discuss briefly here a couple of approaches we investigated towards resolving it, and some barriers that would need to be overcome to make them work.

5.1 Basing fine-grained cryptography on hardness of matrix multiplication

One of the most important problems studied in algorithms and fine-grained complexity is *matrix multiplication*: given two matrices $A, B \in \mathbb{F}_p^{n \times n}$ (for some $p = n^{o(1)}$), compute the matrix product AB. The trivial algorithm takes $n^{3+o(1)}$ time, but a long series of works have shown that the problem can in fact be solved in time $O(n^{2.3714})$ [ADWXXZ25]. Whether the problem is in fact solvable in time $n^{2+o(1)}$ — i.e. near-linear in the input length — is a major open question. Some evidence for the belief that n^2 is possible: 2.3714 is a ridiculous number, surely this problem should have a simple runtime. Some evidence for the belief that n^2 is not possible: people have tried quite hard and have only been able to make increasingly incremental progress on decreasing the exponent. Also, there are known barriers showing that current approaches (based on the Coppersmith-Winograd tensor) are fundamentally incapable of achieving runtime $n^{2+o(1)}$ [AFL15; AW18].

We consider here the possibility of basing fine-grained cryptography on the conjecture that $\omega > 2$ — that is, that it is impossible to solve matrix multiplication in time $n^{2+o(1)}$. We stress again that it is highly unclear whether or not this conjecture is true — however its truth seems like quite a distinct question from whether or not standard one-way functions exist. In particular, this is a question of worst-case hardness, as opposed to average-case hardness over planted distributions.

The reason this would seem a potentially attractive assumption to base fine-grained cryptography on is that matrix multiplication is one of the only types of problems for which we know non-trivial *worst-case to average-case* reductions. For instance, given an $n^{c+o(1)}$-time algorithm for matrix multiplication which, for uniform random $A, B \leftarrow \mathbb{F}_p^{n \times n}$, gets $n^2/p + n^2/1000$ entries of the product correct in expectation, we can achieve an $n^{c+o(1)}$-time algorithm to solve matrix multiplication perfectly in worst-case [HS25a]. So, one

might hope to convert the *worst-case* assumption that $\omega > 2$ to some useful *average-case* assumption upon which to base fine-grained cryptography.

The above result does indeed let us deduce that computing the product AB given uniform random $A, B \leftarrow \mathbb{F}_p^{n \times n}$ is hard. However, this does not seem to be the sort of hardness useful for cryptography: we would like to be able to generate a problem from a hard distribution with a *planted* solution, faster than it would have taken to solve the problem directly. Towards this goal, we propose the following candidate fine-grained one-way function:

Conjecture 5.2. Fix some $\log(n) \ll p \ll n^{o(1)}$, and consider the function $f : \mathbb{F}_p^{n \times n} \times \mathbb{F}_p^{n \times n} \times [n] \times [n] \to \mathbb{F}_p$, where $f(A, B, i, j) = AB[i, j]$ — that is, the i, jth entry of their product. We conjecture that, for uniform random $A, B \leftarrow \mathbb{F}_p^{n \times n}$, $i, j \leftarrow [n]$, and $r \leftarrow \mathbb{F}_p$, no time-$n^{2.01}$ algorithm can distinguish between the distributions $(A, B, f(A, B, i, j))$ and (A, B, r) with constant advantage.

That is, over a large finite field, it is hard to distinguish a random entry of the product of two matrices from a random field element.

Proposition 5.3. Assuming Conjecture 5.2, there exist non-trivial fine-grained one-way functions.

Proof. Consider the algorithm which, on input (A, B, i, j), outputs $(A, B, AB[i, j])$. This function is computable in $n^{2+o(1)}$ time — i.e. linear in the size of the input/output — since computing $AB[i, j]$ simply requires multiplying and adding $O(n)$ values. However, if this function was invertible in $n^{2+o(1)}$ time with constant probability, then we could break Conjecture 5.2. It is exceedingly unlikely that (A, B, r) would have a valid preimage for a random $r \leftarrow \mathbb{F}_p$, since r is exceedingly unlikely to be an entry of the matrix product. So, if we could invert with constant probability, we could distinguish $(A, B, AB[i, j])$ from (A, B, r) with constant advantage. ■

So, average-case hardness of this particular form of matrix multiplication would be enough to give fine-grained cryptography. And indeed, under worst-case hardness of standard fine-grained complexity problems, we can show that this problem is at least hard on worst-case:

Proposition 5.4. Suppose that there exists an $O(n^{2.01})$-time algorithm that perfectly distinguishes $(A, B, AB[i, j])$ from (A, B, r) for r a non-entry of AB. Then, there exists an $O(n^{2.01+o(1)})$-time algorithm for distinguishing between triangle-free n-vertex graphs and n-vertex graphs which contain exactly one triangle.

Proposition 5.5. Given a graph G, we can take the adjacency matrix A. Determining whether any vertex of G is involved in exactly one triangle is equivalent to determining whether there are any entries where both $(A + I)^2$ and $(A + I)$ are 1. Consider the block matrix product

$$\begin{bmatrix} n^{10}(A + I) & (A + I) \end{bmatrix} \begin{bmatrix} I \\ (A + I) \end{bmatrix}.$$

Observe that this product will contain the entry $n^{10} + 1$ if and only if there is some entry where both $(A+I)^2$ and $(A + I)$ are 1. (This is true for the matrix product over $\mathbb{Z}$ — since all of the entries are small, it would also hold over $\mathbb{F}_p$ for any $p \gg n^{10}$).

So, in order to get non-trivial fine-grained one-way functions based solely on the worst-case hardness of unique triangle detection, it would suffice to give a worst-case to average-case reduction for Conjecture 5.2. Unfortunately, such a reduction does not seem to follow from standard techniques. The usual worst-to-average reductions for matrix multiplication problems relies on the fact that the output is a low-degree multivariate polynomial in the input: each entry of the product matrix is a degree-2 polynomial of the entries of the two input matrices. So, we can evaluate this polynomial on a few random points and do polynomial interpolation to get the result at any particular point we care about. Conjecture 5.2 does *not* deal with a low-degree polynomial in the input, so worst-to-average reductions would have to use very

different techniques. And it seems difficult to base fine-grained cryptography on the hardness of any low-degree polynomial in the matrix product, because either that polynomial would have to depend on many entries of the product (in which case its likely hard to compute even for the planter), or to depend on very few (in which case those would have to be a fixed few entries, so its easy for both the planter and the solver).

5.2 Basing fine-grained cryptography on circuit lower bounds

Another very enticing hope would be to construct fine-grained one-way functions based only on the existence of variants of worst-case circuit lower bounds. We have long known how to construct "complexity theoretic" pseudorandom generators based only on such assumptions. In particular, we know for instance that the existence of a function in $\mathsf{TIME}[2^{O(n)}]$ but not $\mathsf{SIZE}[2^{\varepsilon n}]$ would imply, for some constants $c < c'$, the existence of a length-extending function computable in time $n^{c'}$ but whose outputs are indistinguishable from random to time-n^c algorithms.

Of course, this is not the type of guarantee we need for cryptography. We really wanted $c' < c$ — i.e. we want a function that's easy to compute but fools even stronger adversaries. Here's an idea for how one might boost the former guarantee into the latter: given a huge amount of randomness as input, leave most of it unchanged, but replace a small portion of it with the output of a complexity-theoretic PRG. That is, given uniform random $u_1, \ldots, u_{n^c}$, seed s, and index i, output $(u_1, \ldots, u_i, G(s), u_{i+1}, \ldots, u_{n^c})$. Observe that this output can be computed in linear time. However, one might heuristically hope that this is difficult to distinguish from random: the naive approach would be to consider each entry of the list separately and run the $n^{c'}$-time distinguisher, which would take a total of $n^{c'+c}$-time — much more than it took to compute the output!

Formalizing this intuition seems difficult, however. We did discover a recent paper analyzing a fine-grained one-way function construction very similar to this, but they had to simply assert the existence of functions with the appropriate kind of average-case hardness [BC22]. In order to formally get this particuar kind of "direct sum" hardness, one would need unusual guarantees on the complexity-theoretic generator G, which it is unclear how to get from the standard Impagliazzo–Nisan–Wigderson construction. In particular, the place the standard argument seems to fail is in the Yao "decision-to-prediction" reduction. It still seems to us that building the appropriate kind of direct-sum-hard complexity-theoretic PRGs could be a promising approach to building fine-grained cryptography from worst-case assumptions, but it is unclear how to do so.

6 Contributions of team members

The theoretical observations of this project were developed in joint meetings involving all team members. Section 1 of this report was written by Jakob Nogler, Section 2 jointly by Jakob Nogler and Zoe Xi, Section 3 by Zoe Xi, and Section 4 and Section 5 by Nathan Sheffield. All team members assisted in the editing and proofreading of the final document.

References

[ADWXXZ25] Josh Alman, Ran Duan, Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. "More asymmetry yields faster matrix multiplication". In: *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2025, pp. 2005–2039.

[AFL15] Andris Ambainis, Yuval Filmus, and François Le Gall. "Fast Matrix Multiplication". In: *Proceedings of the forty-seventh annual ACM symposium on Theory of Computing* (2015), pp. 585–593. DOI: 10.1145/2746539.2746554.

[AGGS22] Vahid R. Asadi, Alexander Golovnev, Tom Gur, and Igor Shinkar. "Worst-case to average-case reductions via additive combinatorics". In: *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2022. Rome, Italy: Association for Computing Machinery, 2022, pp. 1566–1574. ISBN: 9781450392648. DOI: 10.1145/3519935.3520041. URL: https://doi.org/10.1145/3519935.3520041.

[AHY25] Josh Alman, Yizhi Huang, and Kevin Yeo. "Fine-grained complexity in a world without cryptography". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2025, pp. 375–405.

[AW18] Josh Alman and Virginia Vassilevska Williams. "Limits on All Known (and Some Unknown) Approaches to Matrix Multiplication". In: *arXiv* (2018). DOI: 10.48550/arxiv.1810.08671.

[BC22] Chris Brzuska and Geoffroy Couteau. "On building fine-grained one-way functions from strong average-case hardness". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2022, pp. 584–613.

[BGHKP24] Marshall Ball, Juan Garay, Peter Hall, Aggelos Kiayias, and Giorgos Panagiotakos. "Towards Permissionless Consensus in the nbsp Standard Model via Fine-Grained Complexity". In: *Advances in Cryptology – CRYPTO 2024: 44th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2024, Proceedings, Part II*. Santa Barbara, CA, USA: Springer-Verlag, 2024, pp. 113–146. ISBN: 978-3-031-68378-7. DOI: 10.1007/978-3-031-68379-4_4. URL: https://doi.org/10.1007/978-3-031-68379-4_4.

[BRSV17] Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. "Average-case fine-grained hardness". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2017. Montreal, Canada: Association for Computing Machinery, 2017, pp. 483–496. ISBN: 9781450345286. DOI: 10.1145/3055399.3055466. URL: https://doi.org/10.1145/3055399.3055466.

[BRSV18] Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. "Proofs of Work From Worst-Case Assumptions". In: *Advances in Cryptology – CRYPTO 2018: 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19–23, 2018, Proceedings, Part I*. Santa Barbara, CA, USA: Springer-Verlag, 2018, pp. 789–819. ISBN: 978-3-319-96883-4. DOI: 10.1007/978-3-319-96884-1_26. URL: https://doi.org/10.1007/978-3-319-96884-1_26.

[DLW20] Mina Dalirrooyfard, Andrea Lincoln, and Virginia Vassilevska Williams. "New Techniques for Proving Fine-Grained Average-Case Hardness". In: *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*. 2020, pp. 774–785. DOI: 10.1109/FOCS46700.2020.00077.

[Gol04] Oded Goldreich. *Foundations of cryptography, volume 1*. Cambridge university press Cambridge, 2004.

[GSS24] Ashish Gola, Igor Shinkar, and Harsimran Singh. "Matrix Multiplication Reductions". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2024)*. Ed. by Amit Kumar and Noga Ron-Zewi. Vol. 317. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024, 34:1–34:15. ISBN: 978-3-95977-348-5. DOI: 10.4230/LIPIcs.APPROX/RANDOM.2024.34. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.APPROX/RANDOM.2024.34.

[HS23] Shuichi Hirahara and Nobutaka Shimizu. "Hardness Self-Amplification: Simplified, Optimized, and Unified". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*. STOC 2023. Orlando, FL, USA: Association for Computing Machinery, 2023, pp. 70–83. ISBN: 9781450399135. DOI: 10.1145/3564246.3585189. URL: https://doi.org/10.1145/3564246.3585189.

[HS25a] Shuichi Hirahara and Nobutaka Shimizu. "An optimal error-correcting reduction for matrix multiplication". In: *52nd International Colloquium on Automata, Languages, and Programming (ICALP 2025)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. 2025, pp. 97–1.

[HS25b] Shuichi Hirahara and Nobutaka Shimizu. "Error-Correction of Matrix Multiplication Algorithms". In: *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*. STOC '25. Prague, Czechia: Association for Computing Machinery, 2025, pp. 785–794. ISBN:

9798400715105. DOI: 10.1145/3717823.3718244. URL: https://doi.org/10.1145/3717823.3718244.

[KPR25] David Kühnemann, Adam Polak, and Alon Rosen. "The Planted Orthogonal Vectors Problem". In: *33rd Annual European Symposium on Algorithms, ESA 2025, Warsaw, Poland, September 15-17, 2025*. Ed. by Anne Benoit, Haim Kaplan, Sebastian Wild, and Grzegorz Herman. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025, 95:1–95:17. DOI: 10.4230/LIPICS.ESA.2025.95. URL: https://doi.org/10.4230/LIPIcs.ESA.2025.95.

[KW19] Daniel M. Kane and Richard Ryan Williams. "The Orthogonal Vectors Conjecture for Branching Programs and Formulas". In: *10th Innovations in Theoretical Computer Science Conference (ITCS 2019)*. Ed. by Avrim Blum. Vol. 124. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019, 48:1–48:15. ISBN: 978-3-95977-095-8. DOI: 10.4230/LIPIcs.ITCS.2019.48. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ITCS.2019.48.

[LLV19] Rio LaVigne, Andrea Lincoln, and Virginia Vassilevska Williams. "Public-Key Cryptography in the Fine-Grained Setting". In: *Advances in Cryptology – CRYPTO 2019: 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18–22, 2019, Proceedings, Part III*. Santa Barbara, CA, USA: Springer-Verlag, 2019, pp. 605–635. ISBN: 978-3-030-26953-1. DOI: 10.1007/978-3-030-26954-8_20. URL: https://doi.org/10.1007/978-3-030-26954-8_20.

[WIL] VIRGINIA VASSILEVSKA WILLIAMS. "ON SOME FINE-GRAINED QUESTIONS IN ALGORITHMS AND COMPLEXITY". In: *Proceedings of the International Congress of Mathematicians (ICM 2018)*, pp. 3447–3487. DOI: 10.1142/9789813272880_0188. eprint: https://www.worldscientific.com/doi/pdf/10.1142/9789813272880_0188. URL: https://www.worldscientific.com/doi/abs/10.1142/9789813272880_0188.