On the Randomness Complexity of Interactive Proofs and Statistical Zero-Knowledge Proofs*

Benny Applebaum† Eyal Golombek*

Abstract

We study the randomness complexity of interactive proofs and zero-knowledge proofs. In particular, we ask whether it is possible to reduce the randomness complexity, R, of the verifier to be comparable with the number of bits, C_V, that the verifier sends during the interaction. We show that such *randomness sparsification* is possible in several settings. Specifically, unconditional sparsification can be obtained in the non-uniform setting (where the verifier is modelled as a circuit), and in the uniform setting where the parties have access to a (reusable) common-random-string (CRS). We further show that constant-round uniform protocols can be sparsified without a CRS under a plausible worst-case complexity-theoretic assumption that was used previously in the context of derandomization.

All the above sparsification results preserve statistical-zero knowledge provided that this property holds against a *cheating verifier*. We further show that randomness sparsification can be applied to honest-verifier statistical zero-knowledge (HVSZK) proofs at the expense of increasing the communication from the prover by $R-F$ bits, or, in the case of honest-verifier perfect zero-knowledge (HVPZK) by slowing down the simulation by a factor of 2^{R-F}. Here F is a new measure of *accessible bit complexity* of an HVZK proof system that ranges from 0 to R, where a maximal grade of R is achieved when zero-knowledge holds against a "semi-malicious" verifier that maliciously selects its random tape and then plays honestly. Consequently, we show that some classical HVSZK proof systems, like the one for the complete Statistical-Distance problem (Sahai and Vadhan, JACM 2003) admit randomness sparsification with no penalty.

Along the way we introduce new notions of pseudorandomness against interactive proof systems, and study their relations to existing notions of pseudorandomness.

1 Introduction

Randomness is a valuable resource. It allows us to speed-up computation in various settings and it is especially useful, or even essential, at the presence of adversarial behavior. Consequently, an extensive body of research has been devoted to the question of minimizing the randomness complexity in various contexts. Notably, the seminal notion of *pseudorandomness* [BM82, Yao82] has been developed as a universal approach for saving randomness or even completely removing the need for random bits. In this paper, we study this general question in the context of *(probabilistic) interactive proofs*.

*To appear in ITC'21.

†Tel-Aviv University, Israel bennyap@post.tau.ac.il, eyalrs21@gmail.com. Supported by the European Union's Horizon 2020 Programme (ERC-StG-2014-2020) under grant agreement no. 639813 ERC-CLC, and the Check Point Institute for Information Security.

Interactive proofs, presented by [GMR89, BM88], form a natural extension of non-deterministic polynomial time computation (NP). A computationally-bounded probabilistic verifier V wishes to decide whether an input x is a member of a promise problem[1] $\Pi = (\Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}})$ with the aid of a computationally-unbounded untrusted prover P who tries to convince V that x is a yes-instance. Towards this end, the two parties exchange messages via a protocol, and at the end the verifier decides whether to accept or to reject the input. The protocol should achieve *completeness* and *soundness*. The former asserts that yes-instances should be accepted except for some small probability (*completeness error*), and the latter asserts that no-instances should be rejected regardless of the prover's strategy except for some small probability (*soundness error*). (See Definition 2.2.)

The celebrated result of [LFKN92, Sha92] shows that interactive proofs are as strong as polynomial-space computations (i.e., $\mathsf{IP} = \mathsf{PSPACE}$). Moreover, randomness seems essential for this result: If one limits the verifier to be deterministic then interaction does not really help – the prover can predict the verifier messages and so can send all the answers at once – and the power of such proof systems is limited to NP. Put differently, randomness provides "unpredictability" which is crucial for achieving soundness, i.e., for coping with a cheating prover. In fact, even in cases where soundness can be achieved deterministically (i.e., when the underlying problem is in NP) one may want to use a randomized proof system. This is the case, for example, when the prover wants to hide some information from the verifier like in the case of *zero-knowledge proofs* [GMR89]. Indeed, deterministic proof systems inherently allow the verifier to convince others in the validity of the statement, a property that violates zero-knowledge for non-trivial languages [Ore87]. In this context, randomness is used for *hiding information* similarly to its use in the setting of randomized encryption [GM84].

How much randomness is needed for interactive proofs? We would like to understand how randomness complexity scales with other resources. Specifically, we would like to relate it to the communication complexity of the protocol – a measure that was extensively studied in the context of interactive proofs and for which we have better understanding (e.g., [GH98, GVW02]). We therefore ask:

> Given an interactive proof system $\langle P, V \rangle$ for a problem Π, can we always *sparsify* the randomness complexity R to be comparable with the amount of communication complexity? Can we do this while preserving zero-knowledge?

We use the term *randomness sparsification* to highlight the point that we do not aim for full *de-randomization*, rather we only try to make sure that the randomness complexity is not much larger than the communication complexity.

1.1 Related works

Clearly the question of sparsification becomes trivial for public-coin protocols (aka Arthur-Merlin protocols) in which all the randomness of the verifier is being sent during the protocol. Goldwasser and Sipser [GS86] showed that any general interactive proof protocol can be transformed into public-coin protocol, however,

[1] A promise problem [ESY84] is a partition of the set of all strings into three sets: Π_{yes} the set of *yes instances*, Π_{no} the set of *no instances*, and $\{0,1\}^* \setminus (\Pi_{\mathsf{yes}} \cup \Pi_{\mathsf{no}})$ the set of "disallowed strings". The more common notion of a *language* corresponds to the special case where Π_{no} is the complement of Π_{yes} (i.e., there are no disallowed strings). The promise problem formalization is especially adequate for the study of interactive proofs and is therefore adopted for this paper. See [Gol06] for a thorough discussion.

this transformation increases the randomness complexity of the new system and therefore does not resolve the sparsification question.

Information-theoretically, if the verifier sends at most C_V bits during the whole interaction, it should be possible to emulate it with about C_V bits of randomness (in expectation). Indeed, in the context of two-party communication complexity games, it is well known [New91] that randomized protocols that use R random bits can be converted into protocols whose randomness complexity is not much larger than the communication complexity C. While this result can be generalized to the setting of interactive proof systems [AV19], it does not preserve the computational complexity of the verifier. Specifically, this sparsification is essentially based on an inefficient pseudorandom generator G whose existence follows from the probabilistic method.

The question of *efficient sparsification* in the related context of *information-theoretic secure multi-party computation* (ITMPC) was addressed by Ishai and Dubrov [DI06]. They introduced the notion of *non-Boolean PRG* (nb-PRG) and showed that such a PRG can be used to sparsify *efficiently-computable* protocols with *passive* security.[2] The definition of nb-PRG generalizes the standard notion of PRG by considering non-Boolean distinguishers. Formally, a (T, C, ε) nb-PRG $G : \{0,1\}^S \to \{0,1\}^R$ fools any T-time non-Boolean algorithm $D : \{0,1\}^R \to \{0,1\}^C$ with C output bits in the sense that $D(U_R)$ is ε-close (in statistical distance) to $D(G(U_S))$ where U_N denotes the uniform distribution over N-bit strings. For polynomially related parameters, nb-PRGs with an optimal seed length of $O(C)$ bits can be obtained either based on (exponentially strong) cryptographic assumptions [DI06] or based on standard worst-case complexity-theoretic assumptions [AS17, AASY16]. In order to sparsify a passively-secure efficient ITMPC protocol, it suffices to invoke the parties over pseudorandom tapes that are selected according to (T, C, ε) nb-PRG where C upper-bounds the number of bits communicated to the adversary and T is the total computational complexity of the protocol. The main idea is to note that any fixed coalition of corrupted parties receives from the honest parties at most C bits of incoming messages whose distribution can be generated by applying a procedure D to the pseudorandom tapes of the honest parties. The procedure D is obtained by "gluing" together the codes of all parties, and can therefore be implemented with complexity T. Since the underlying nb-PRG fools D, the sparsified protocol remains information-theoretic private: An *external* unbounded environment that examines the view of the adversary "learns" nothing on the honest parties inputs.

The above argument relies on the efficiency of all *internal parties* that participate in the protocol. It is therefore unclear whether it can be extended it to our setting where prover, even when played honestly, may be computationally unbounded.[3] Nuida and Hanaoka [NH12] pointed out to the limitation of the nb-PRG approach in the context of "leaky" distinguishing games with an *internal* computationally-unbounded adversary, and suggested to use exponentially-strong cryptographic pseudorandom generators (whose distinguishing advantage is exponential in the leakage available to the adversary). It should be mentioned, however, that although the original sparsification argument of [DI06] fails, we do not know whether nb-PRGs suffice for sparsification neither in our context nor in the more general context suggested by [NH12]. In fact, known concrete constructions of nb-PRGs (e.g., ones that are based on exponential cryptographic-PRGs) seem to suffice for this purpose.

Finally, let us mention that several works have studied other aspects of randomness complexity in the

[2] More precisely, their sparsification applies to protocols with privacy against parties that passively follow the protocol but may select their random tape arbitrarily. (In addition, they used an indistinguishability-based definition which is equivalent to unbounded simulation, however, their result seems to generalize to the case of efficient simulation as well.)

[3] In contrast, one can use nb-PRGs (against arbitrary polynomial-time adversaries) to sparsify *efficiently-computable argument systems*. In such systems correctness holds with respect to an efficient prover strategy, and soundness is required to hold only against efficient provers. However, this setting has no information-theoretic flavor and a standard cryptographic pseudorandom generator can be used as well.

context of public-coin interactive proof systems. This includes randomness-efficient methods for round-reduction [BR94] and for error-reduction [BGG93].

1.2 Our Results and Techniques

In this paper, we present several sparsification results for interactive proofs and for zero-knowledge proofs. We begin with the former case.

1.2.1 General Interactive Proofs

Before stating our results, we set-up some notation.

Notation 1.1. *For polynomially-bounded integer-valued functions R, C_V, T_V, C_P and k we consider proof systems that on an n-bit input, the parties exchange $k(n)$ messages, where the verifier V uses $R(n)$ random bits, sends a total number of $C_V(n)$ bits, and runs in time $T_V(n)$, and the prover sends a total number of $C_P(n)$ bits. We refer to such protocols as $\mathsf{IP}_k[R, C_V, T_V, C_P]$ protocols. We also consider* non-uniform *$\mathsf{IP}_k[R, C_V, T_V, C_P]$ protocols in which the verifier is implemented by a T_V-size circuit. We sometimes omit k and use $\mathsf{IP}[R, C_V, T_V, C_P]$ (or* non-uniform *$\mathsf{IP}[R, C_V, T_V, C_P]$) to denote a protocol with an unspecified round complexity. (Observe that in any case k is upper-bounded by $C_V + C_P$.) Similarly, we let IP (resp.,$\mathsf{IP}/\mathsf{poly}, \mathsf{IP}_k$) denote the union of $\mathsf{IP}[R, C_V, T_V, C_P]$ (resp., non-uniform $\mathsf{IP}[R, C_V, T_V, C_P]$, $\mathsf{IP}_k[R, C_V, T_V, C_P]$) where R, C_V, T_V, C_P range over all polynomially-bounded functions.*

PRGs against interactive proofs. Let us begin by presenting a natural definition for a PRG against an interactive proof. Consider an $\mathsf{IP}[R, C_V, T_V, C_P]$ proof system $\langle P, V\rangle$ for a problem Π with completeness error of δ_c and soundness error of δ_s. For a length-extending function $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ we define the verifier $V^G(x)$ to be the verifier that samples a *seed* $s \leftarrow\!\!\shortmid \{0,1\}^{S(n)}$ and invokes V with the random tape $G(s)$ on the input x. We say that G *ε-fools* the protocol $\langle P, V\rangle$ if $\langle P, V^G\rangle$ forms an interactive proof system for Π with an additive penalty of ε in the completeness and soundness error, i.e., the completeness error and soundness errors are upper-bounded by $\delta_c + \varepsilon$ and by $\delta_s + \varepsilon$, respectively.

We begin by noting that, in the non-uniform setting, one can construct such PRGs *unconditionally* with a seed length that is linear in the verifier's communication complexity and logarithmic in its running time.

Theorem 1.2. *For every functions $T_V(n), C_V(n), C_P(n), R(n) : \mathbb{N} \to \mathbb{N}$ and $\epsilon : \mathbb{N} \to [0,1]$, there exists a $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ that can be computed by a non-uniform $\widetilde{O}(RT_V)$-size circuit and ε-fools every non-uniform $\mathsf{IP}[R, C_V, T_V, C_P]$ protocol where $S = 2C_V + 2\log(1/\varepsilon) + \log T_V + \log\log T_V + O(1)$.*

As an immediate corollary we derive the following result.

Theorem 1.3 (Non-Uniform Randomness Sparsification for IP)**.** *Suppose that a promise problem Π has a (possibly non-uniform) $\mathsf{IP}[R, C_V, T_V, C_P]$ interactive proof $\langle P, V\rangle$ with completeness error δ_c and soundness error δ_s. Then, for every $\varepsilon(n)$, the promise problem Π also has a non-uniform proof system $\langle P, V'\rangle$ whose verifier is a non-uniform algorithm with randomness complexity $R' = O(C_V + \log(1/\varepsilon) + \log T_V)$ and computational complexity of $T'_V = T_V + \widetilde{O}(T_V(R + \log(1/\varepsilon)))$ and with identical communication complexity ($C'_V = C_V$ and $C'_P = C_P$), and identical round complexity. The soundness and completeness*

error of the new system are $\delta'_s \leqslant \delta_s + \varepsilon$ and $\delta'_c \leqslant \delta_c + \varepsilon$. Moreover, if the original proof system has a prefect completeness then so is the new system.[4]

The PRG construction (Theorem 1.2) is based on a family of t-wise independent hash functions. That is, we show that, for a properly chosen parameter t, a randomly chosen t-wise independent hash function is likely to fool $\mathsf{IP}[R, C_V, T_V, C_P]$. Unfortunately, one has to invest too many random bits in order to sample a hash function, and so we use non-uniformity to hard-wire one "good" hash function. (See Section 3 for details.) An alternative solution is to select the hash function via a common-random-string (CRS) that is available to both parties and can be reused among many invocations.[5] This also leads to uniform sparsification in an amortized setting where many instances are considered together. In such a case one can even remove the CRS and let the verifier sample it once for all the instances. (See Corollary 3.8.)

Single-Instance Sparsification in the uniform setting without CRS? A natural way for achieving randomness sparsification in the uniform setting is to "sparsify" the process of selecting the hash function. That is, to use a different pseudorandom generator to sample a hash function. Indeed, this approach was taken by [AASY16] to construct nb-PRGs. The idea is to show that given the description of a hash function h_z one can determine with "not-too-large-complexity" (e.g., low in the polynomial hierarchy) whether h_z fools an interactive proof system. If such a decision can be made by some "algorithm" D then we can select the hash function by using a PRG that fools D. Unfortunately, our definition of "fooling interactive proofs" does not seem to be efficiently-decidable. First, the definition implicitly refers to inputs that satisfy the promise of the underlying problem Π, and deciding whether an input x belongs to $\Pi_{\mathsf{yes}} \cup \Pi_{\mathsf{no}}$ may be very hard. Second, as part of the pseudorandomness requirement, the new system $\langle P, V^G \rangle$ should preserve completeness (up to an error of ε). However, this property depends on the behavior of the honest prover P which is an *inefficient* procedure on which we have no "handle". In particular, even if we try to design an interactive proof system for deciding whether h_z is a good PRG, it is not clear how to make sure that the unbounded prover really uses the honest P when needed.

Strong PRGs. We solve both problems by *strengthening* the notion of pseudorandomness against interactive proofs. Specifically, we say that G *strongly ε-fools* the protocol $\langle P, V \rangle$ if for every string $x \in \{0,1\}^*$ and every possible prover strategy P^*, the gap between the acceptance probability of $V(x)$ when interacting with $P^*(x)$ and the acceptance probability of $V^G(x)$ when interacting with $P^*(x)$ is at most ε. While this definition seems stronger than the previous one, the proof of Theorem 1.2 actually shows that random hash functions *strongly* fool interactive proofs. Crucially, this new definition makes no reference to the underlying promise problem or to the honest prover P. (Indeed, one may say that G ε-fools the *interactive machine* V.) As a result, the above-mentioned obstacles are removed and we can show that the problem of checking whether a given hash function h_z strongly-fools an IP_k proof system admits an IP_{k+1} proof system. For constant k, this puts the language of "bad" hash functions in the class AM and so we can select our hash function by a pseudorandom generator that fools AM – a well-studied object in complexity theory. Specifically, known constructions of such PRGs [NW94, IW97, KVM02, SU06] can be based on the assumption that $\mathsf{E} = \mathsf{DTime}(2^{O(n)})$ is hard for exponential size non-deterministic circuits. (See Theorem 4.3 for details). In Section 4 we prove the following result.

[4] All our transformations preserve perfect completeness. From now on, we omit this point throughout this section.

[5] In many scenarios such a CRS is available "for free". Furthermore, the fact that it is re-usable and that it should not be kept private from the prover even before the protocol begins, makes it highly-attractive even compared to *public coins*.

Theorem 1.4 (Uniform Randomness Sparsification for constant-round proofs)**.** *Suppose that* E *is hard for exponential size non-deterministic circuits. Then, for every inverse polynomial* ε*, every constant* k *and every polynomially-bounded functions* R, C_V, T_V, C_P*, there exists a* PRG *computable in uniform polynomial time of* $T'_V = \widetilde{O}(T_V \cdot (R + \log n))$ *that strongly* ε*-fools non-uniform* $\mathsf{IP}_k[R, C_V, T_V, C_P]$ *proof systems with seed length of* $R' = 2C_V + O(\log n)$*. Consequently, every* $\mathsf{IP}_k[R, C_V, T_V, C_P]$ *proof system can be transformed into a new* $\mathsf{IP}_k[R', C_V, T'_V, C_P]$ *with an additive penalty of* ε *in the soundness and completeness errors. Moreover, perfect completeness is preserved.*

The underlying assumption can be viewed as a natural extension of $\mathsf{EXP} \neq \mathsf{NP}$ to the non-uniform settings. Similar assumptions were made in the literature (e.g., [BOV03, Dru13, FL92, GW02, TV00]).

Remark 1.5. *One should note that when* k *is constant the underlying assumption suffices for full de-randomization of the protocol (via a sequence of transformations). Still, one may prefer to use the sparsified protocol (that still uses some randomness), either due to its efficiency properties (in terms of computation and communication) or due to its zero-knowledge properties as discussed in Section 1.2.2.*

The seed length of our PRGs is dominated by the number of bits, C_V, sent by the verifier. (This is the case both in the uniform and non-uniform settings.) It is not hard to show that such a dependency is essentially optimal even if one considers the weaker variant of IP PRGs.

Proposition 1.6 (Sparsification lower-bound)**.** *For every functions* $T_V(n), C_V(n), C_P(n), R(n) : \mathbb{N} \to \mathbb{N}$ *where* $C_V < R$ *every* $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ *that* 0.1*-fools* $\mathsf{IP}[R, C_V, T_V, C_P]$ *protocols must have a seed length of* $\Omega(C_V)$*.*

Proof. Assume that $S < \alpha C_V$ for some small constant $\alpha < 1$. A simple information-theoretic argument shows that $y = G(U_S)$ is predicatable in the following sense. There exists an index $i \in [C_V]$ such that given the $(i-1)$-prefix $y[1 : i-1]$, one can guess (possibly inefficiently) the next bit $y[i]$ with success probability of, say, 0.8. Indeed, letting $p_i := H(y_i | y[1 : i-1])$ denote the conditional entropy of y_i given the prefix, we know that $\sum_{i=1}^{C_V} p_i \leqslant H(y) = S < \alpha C_V$ and so, by an averaging argument, there exists an index i for which $p_i < \alpha$. For sufficiently small constant α, this implies that y_i is predictable with probability 0.8. Consider the following proof system for the trivial empty language ($\Pi_{\mathsf{yes}} = \emptyset$ and $\Pi_{\mathsf{no}} = \{0,1\}^*$). The verifier samples $r \in \{0,1\}^R$ and sends $r[1 : i-1]$ to the prover who responds with a single bit b. The verifier accepts if $b = y[i]$. When r is random the soundness error is $1/2$, but when $r = G(U_S)$, the error grows to 0.8. □

nb-PRGs are not IP-PRGs. We also show (in Appendix A) that, under plausible cryptographic assumptions, some nb-PRGs do not fool IP protocols. Roughly, this is done by constructing a nb-PRG which is *malleable*. That is, although the prover cannot tell whether the verifier uses random bits or bits that were generated via the nb-PRG, she can provide a short hint that allows a computationally-bounded algorithm (the original verifier) to distinguish between the two cases. Our results therefore show that the inapplicability of nb-PRGs to our setting reflects an inherent limitation and it is not just an artifact of the previous proof techniques.

1.2.2 Zero-Knowledge Proofs

We move on and study randomness-sparsification for statistical zero-knowledge proofs. In the following we focus on constant-round zero-knowledge protocols with a uniform verifier and base our results on the

assumption from Theorem 1.4. If one is willing to make the verifier non-uniform (or to allow a public common reference string), then the following results can be proved unconditionally without assumptions for protocols with an arbitrary number of rounds.

Let $\mathsf{SZK}_k[R, C_V, T_V, C_P]$ be an $\mathsf{IP}_k[R, C_V, T_V, C_P]$ statistical zero-knowledge protocol, whose zero-knowledge property holds against an arbitrary, possibly malicious, verifier that may deviate from the protocol. We begin by noting that PRG-based randomness-sparsification trivially preserves such a strong zero knowledge property.

Theorem 1.7 (Uniform Randomness Sparsification for constant-round SZK)**.** *Suppose that* E *is hard for exponential size non-deterministic circuits. Then, for every inverse polynomial* ε*, every constant-round* $\mathsf{SZK}_k[R, C_V, T_V, C_P]$ *proof system can be transformed into a new* $\mathsf{SZK}_k[R', C_V, T'_V, C_P]$ *with randomness of* $R' = 2C_V + O(\log n)$*, (uniform) verifier's complexity of* $T'_V = \widetilde{O}(T_V \cdot (R + \log n))$ *and with an additive penalty of* ε *in the soundness and completeness errors.*

The proof is straightforward: Any malicious verifier strategy that can be played in the original protocol $\langle P, V \rangle$ can be also played in the sparsified protocol $\langle P, V^G \rangle$. Indeed, SZK is a feature of the honest prover that remains unchanged in the sparsified proof system.

Sparsifying HVSZK? We move on and ask whether such a theorem can be proved for the case of *honest-verifier* statistical zero-knowledge protocols (HVSZK). While there are known transformations from HVSZK to SZK (e.g., [Vad99, GSV98, HRV18]) these transformations incur a communication complexity overhead that is at least as large as the randomness complexity of the original protocol. Therefore, the problem of sparsifying HVSZK is not known to be reducible to the sparsification of SZK.

It is instructive to see why Theorem 1.7 does not immediately generalize to the HVSZK setting. Consider for simplicity a 2-message proof system $\langle P, V \rangle$ where V sends a message a and receives a message b. The view of an honest verifier consists of the input x, the random tape r and the incoming message b. In the sparsified system, $\langle P, V^G \rangle$, the view consists of the input x, a PRG seed s and the message a. Suppose that the original verifier admits a simulator that, given x, samples the pair (r, a). How can we use such a simulator to sample (s, a)? If we use the original simulator then a random r is unlikely to land in the image of G which is *sparse* in the set of all R-bit strings. Moreover, even if we hit the image, it is not clear how to invert G and find an appropriate seed. We observe that the second problem can be easily solved by exploiting the concrete structure of our PRGs. Specifically, by using algebraic constructions of t-wise independent hash functions we can efficiently invert the PRGs in polynomial-time.[6] To handle the sparsity problem we suggest two possible approaches:

- Our first solution exploits the prover. We show that the simulation problem can be avoided by asking the prover to supply R random bits at the beginning of the interaction.
- In the context of honest-verifier *perfect* zero-knowledge proofs, we show that randomness can be traded by a simulation slow-down. Specifically, the sparsified protocol (without any modifications) can be simulated with an overhead of time 2^{R-S} where S is the seed-length of the generator. (See Corollary 5.14.) Such a simulation implies witness-indistinguishability [FS90] and can be meaningful

[6] This does not contradict security since our PRG fools verifiers of predetermined fixed polynomial-time (corresponding to the running time of the verifier) but can be inverted in larger polynomial time. This feature of the fixed-polynomial-time setting (that is typically used in the context of derandomization [NW94]) seems novel to this work.

when the underlying problem is harder than 2^{R-S}. Specifically, one can tune S, i.e., the level of sparsification, according to the hardness of the problem.

How much should we pay? In the above solutions we pay a communication overhead of R (resp., simulation slow-down of 2^{R-S}) in the sparsification of HVSZK systems (resp., HVPZK) whereas in the case of SZK proof systems (with security against cheating verifier) we pay nothing. It turns out that one can interpolate between these two extremes based on a single measure. Roughly, we say that a proof system is an F-semi-malicious statistical zero-knowledge system (F-SMSZK), for some function $0 < F < R$ if it is possible to simulate every verifier that plays honestly except that it selects the first F-bits of its random tape by some arbitrary (efficiently-computable) distribution (the other $R - F$ coins are chosen uniformly).[7] We prove the following theorem. (See Corollaries 5.7 and 5.10.)

Theorem 1.8 (Trading randomness with prover's communication or simulation slowdown)**.** *Suppose that* E *is hard for exponential size non-deterministic circuits. Then, every promise problem* Π *that admits a constant-round* F-$\mathsf{SMSZK}_k[R, C_V, C_P, T_V]$ *proof system* $\langle P, V\rangle$ *also has:*

- *An* $\mathsf{HVSZK}_{k+1}[R' = 2C_V + O(\log n), C_V, T'_V = \widetilde{O}(T_V \cdot (R + \log n)), C'_P = C_P + R - F]$ *proof system. Specifically, the new protocol consists of an additional preliminary message from the prover that consists of a random string of length* $R - F$ *bits.*
- *In the prefect zero-knowledge setting, where* $\langle P, V\rangle$ *is* F-$\mathsf{SMPZK}_k[R, C_V, C_P, T_V]$ *system, the problem* Π *admits an* $\mathsf{HVPZK}_k[2C_V + O(\log n), C_V, C_P, T_V]$ *proof system whose simulator runs in time* $\mathrm{poly}(n)2^{R-F}$.

Observe that $0 \leqslant F \leqslant R$ and that any HVSZK proof system is also an 0-SMSZK and every SZK proof system is R-SMSZK. Thus Theorem 1.8 implies Theorem 1.7. Interestingly, some classical HVSZK proof systems also achieve full accessibility of $F = R$. Most notably, this is the case for the classical protocol for the complete *statistical-distance* problem of [SV03] as well as the classical proof system for *graph-non-isomorphism* (GNI) of [GMW91]. (See Section B.) In fact, these proof systems have only two messages and therefore they are known to be insecure against a cheating verifier [Ore87, Theorem 8] (unless the underlying problems are in BPP). It follows that even the notion of R-SMSZK proof systems is likely to be weaker then SZK.

We further mention that even when $F = 0$, we can get some non-trivial simulation for HVPZK. Specifically by exploiting the concrete properties of our PRG we can get a simulator whose complexity is $\mathrm{poly}(n)2^{R-S}$ where R is the original randomness complexity and S is the seed length of the simulator. (See Section 5.5.) As an application, one can adjust the seed length (i.e., the level of sparsification) according to a given time-bound on the simulation (that may be dictated by the intractability of the underlying language).

Organization. Following some preliminaries (Section 2), we study, in Section 3, randomness sparsification for interactive proofs in the non-uniform setting and in the amortized sparsification in the uniform setting. Section 4 is devoted to randomness sparsification for constant-round uniform interactive proofs, and Section 5 to statistical zero-knowledge proofs.

[7] One should not be confused with our notion of semi-malicious SZK proof systems and the one suggested by [LQR+19] that applies to zero-knowledge PCPs.

2 Preliminaries

Probabilistic notation. For every $n \in \mathbb{N}$ we denote by U_n the uniform distribution over the set $\{0,1\}^n$ of binary strings of length n. For a probability distribution $\mathcal{D}$, we use the notation $x \leftarrow\!\!\shortmid \mathcal{D}$ to denote a value x that is sampled according to $\mathcal{D}$. When $\mathcal{D}$ is a finite set, the notation $x \leftarrow\!\!\shortmid \mathcal{D}$ denotes a value x that is sampled uniformly from $\mathcal{D}$. We follow the standard way of defining distance between two distributions:

Definition 2.1 (Statistical Distance)**.** *Given X, Y two probability distributions over some discrete universe Ω the statistical difference between them is defined:*

$$\mathsf{SD}(X,Y) = max_{S \subset \Omega} |\Pr[X \in S] - \Pr[Y \in S]|.$$

Definition 2.2 (Interactive proof system [GMR89])**.** *A pair of interactive machines $\langle P, V \rangle$ is called an interactive proof system with completeness error of δ_c and soundness error of δ_s for a promise problem $\Pi = (\Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}})$ if the followings hold:*

- ***Completeness:*** *For every $x \in \Pi_{\mathsf{yes}}$ we have*

  $$\Pr[(P,V)(x) = 1] \geqslant 1 - \delta_c(|x|)$$

  where the probability is taken over the randomness of V and P and we write $(P,V)(x) = 1$ to denote the event that, after interacting with $P(x)$, the verifier $V(x)$ accepts.

- ***Soundness:*** *For any cheating strategy for the prover P^* and every $x \in \Pi_{\mathsf{no}}$, it holds that*

  $$\Pr[(P^*,V)(x) = 1] \leqslant \delta_s(|x|).$$

When the parameters δ_c and δ_s are unspecified we assume that they are taken to be $o(1)$.[8] *By default, we assume that V is efficient, i.e., it runs in time $T_V(|x|)$ for some polynomially-bounded function T_V. In the non-uniform setting, we assume that V can be implemented by a non-uniform family of $T_V(|x|)$-size probabilistic circuits.*

Following Notation 1.1, we let k, R, C_V, C_P denote the number of messages sent in the protocol, the randomness complexity of V, the number of bits sent by V, and the number of bits sent by P.

Definition 2.3 (Statistical Zero-Knowledge)**.** *An interactive proof system $\langle P, V \rangle$ for a promise problem $\Pi = (\Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}})$ is a* Statistical Zero-Knowledge proof system *(*SZK*) with a simulation error of δ_z if for every computationally-unbounded verifier V^* there exists a simulator* Sim *that runs in time polynomial in the complexity of V^* such that for every yes-instance $x \in \Pi_{\mathsf{yes}}$ it holds that*

$$\mathsf{SD}(\mathsf{view}_{V^*}(x), \mathsf{Sim}(x)) \leqslant \delta_z(|x|),$$

where $\mathsf{view}_{V^*}(x)$ is the random variable that corresponds to the view of $V^*(x)$ when interacting with $P(x)$ which consists of the random tape and all the incoming messages that were sent by P.

The proof system is an Honest-Verifier Statistical Zero-Knowledge proof system *(*HVSZK*) if the above holds for the special case where $V^* = V$. We also denote by* HVSZK *and* SZK *the class of all promise problems that posses such an interactive proof system (with error parameters of $o(1)$).*

[8] Standard ρ-fold parallel repetition reduces the errors exponentially with ρ at the expense of increasing the communication and computation complexity by a factor of ρ and without affecting the round complexity (see e.g., [Gol98]).

3 Non-uniform randomness sparsification for IP

In this section we study the possibility of reducing the randomness of a general proof system (P, V). We begin by defining a strong form of pseudo-random generators against interactive proof systems.

Definition 3.1 (Strongly fooling a protocol)**.** *Let $\langle P, V\rangle$ be a protocol and $R(n)$ denote the randomness complexity of V. For a length-extending function $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ we define the verifier $V^G(x)$ to be the verifier that samples a* seed $s \leftarrow \{0,1\}^{S(n)}$ *and invokes $V(x; r)$ with randomness $r = G(s)$.*

We say that G strongly ε-fools *the protocol $\langle P, V\rangle$ if for every input x and any possible prover strategy P^* it holds that*

$$|\Pr[(V, P^*)(x) = 1] - \Pr[(V^G, P^*)(x) = 1]| \leqslant \varepsilon.$$

We say that G strongly ε-fools $\mathsf{IP}[R, C_V, T_V, C_P]$ *if it strongly ε-fools any interactive proof $\langle P, V\rangle \in \mathsf{IP}[R, C_V, T_V, C_P]$.*

Recall that is the class $\mathsf{IP}[R, C_V, T_V, C_P]$ is the class of IP protocols in which on an n-bit input the verifier runs in $T_V(n)$ time, uses at most $R(n)$ random bits and sends at most $C_V(n)$ bits to the prover, and the total length of the prover responds is at most $C_P(n)$ bits. Observe that a PRG strongly fools a protocol regardless of the prescribed prover, and it is a trait of the verifier.

Observation 3.2. *Suppose that $\langle P, V\rangle$ is an interactive proof system for a promise problem Π with completeness error δ_c and soundness error δ_s and G strongly ε-fools $\langle P, V\rangle$. Then $\langle P, V^G\rangle$ is an interactive proof system for a promise problem Π with completeness error $\delta_c + \varepsilon$ and soundness error $\delta_s + \varepsilon$. Moreover, if the original system has perfect completeness then so is the new system.*

Proof. The first part is immediate from Definition 3.1. The "Moreover" part holds for any G since for any yes instance x a bad (faulty) random string s in $\langle P, V^G\rangle$ for which $(V^G, P)(x)$ rejects translate into a random tape $r = G(s)$ for which $(V, P)(x)$ rejects as well. □

We continue by showing that pseudo-random generators against circuits with very small error can be used to fool protocols.

Lemma 3.3 (Fooling protocols via circuit-PRGs)**.** *For any functions $T_V(n), C_V(n), C_P(n), R(n) : \mathbb{N} \to \mathbb{N}$ and $\epsilon : \mathbb{N} \to [0,1]$, any* PRG $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ *that $\varepsilon/2^{C_V(n)}$-fools $3T_V$-size circuits also strongly ε-fools non-uniform $\mathsf{IP}[R, C_V, T_V, C_P]$ protocols.*

Proof. Let $\langle P, V\rangle$ be some (possibly non-uniform) $\mathsf{IP}[R, C_V, T_V, C_P]$ proof system and let $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ be a PRG that $\varepsilon/2^{C_V(n)}$-fools $3T_V$-size circuits. Fix some input $x \in \{0,1\}^n$ and let $C_V = C_V(n), T_V = T_V(n), C_P = C_P(n)$ and $S = S(n)$. Fix some proof strategy P^*. Let $\mathsf{view}_V(r)$ denote the verifier's view when interacting with P^* on the shared input x with randomness r. This view consists of (x, r), the concatenation, $\vec{a}$ of all the messages sent from V to P^* during the interaction and the messages $\vec{b}$ that were sent from P^* to V during the interaction.[9] In the following, we will think of $(\vec{a}, \vec{b})$ as random variables whose distribution is induced by a random choice of the verifier's random coins.

[9] In the context of this proof, we omit the seed s from the verifier's view. While such an omission will be problematic later when discussing zero-knowledge, it has no consequences in the current proof.

We will show that (*) $\mathsf{view}_V(U_r)$ is ε indistinguishable from $\mathsf{view}_V(G(U_S))$ by T_V-size circuits. Note that (*) implies that $|\Pr[(V,P^*)(x)=1]-\Pr[(V^G,P^*)(x)=1]| \leqslant \epsilon$ since V decides whether to accept its view by applying a predicate which is computable by a circuit of size at most T_V. Let us assume, without loss of generality, that the strategy P^* is deterministic. Indeed, if (*) does not hold for some randomized P^* then, by an averaging argument, there exists a deterministic P^* that violates (*).

We proceed by proving (*). Assume towards contradiction that there exists some distinguisher D of complexity at most T_V that violates (*). Then, we can write

$$\begin{aligned}
\varepsilon < \Bigg| & \sum_{a\in\{0,1\}^{C_V}} \Pr_{r\leftarrow U_R}[D(x,r,\vec{a},\vec{b})=1 \mid \ \vec{a}=a] \Pr_{r\leftarrow U_R}[\vec{a}=a] \\
& - \sum_{a\in\{0,1\}^{C_V}} \Pr_{r\leftarrow G(U_S)}[D(x,r,\vec{a},\vec{b})=1 \mid \ \vec{a}=a] \Pr_{r\leftarrow G(U_S)}[\vec{a}=a] \Bigg| \\
\leqslant \sum_{a\in\{0,1\}^{C_V}} \Bigg| & \Pr_{r\leftarrow U_R}[D(x,r,\vec{a},\vec{b})=1 \mid \ \vec{a}=a] \Pr_{r\leftarrow U_R}[\vec{a}=a] \\
& - \Pr_{r\leftarrow G(U_S)}[D(x,r,\vec{a},\vec{b})=1 \mid \ \vec{a}=a] \Pr_{r\leftarrow G(U_S)}[\vec{a}=a] \Bigg|,
\end{aligned}$$

where the inequality is due to the triangle inequality. By an averaging argument, we conclude that there should be at least one element a^* such that

$$\frac{\varepsilon}{2^{C_V}} < \left| \Pr_{r\leftarrow U_R}[D(x,r,\vec{a},\vec{b})=1 \mid \ \vec{a}=a^*] \Pr_{r\leftarrow U_R}[\vec{a}=a^*] - \Pr_{r\leftarrow G(U_S)}[D(x,r,\vec{a},\vec{b})=1 \mid \ \vec{a}=a^*] \Pr_{r\leftarrow G(U_S)}[\vec{a}=a^*] \right|.$$

Recall that the prover is deterministic and therefore once the verifier's messages are fixed to a^*, the prover's messages become fixed as well to some value b^*. We now can define a new distinguisher $D' : \{0,1\}^{R(n)} \to \{0,1\}$ that holds (x,a^*,b^*) as a non-uniform advice and operates as follows. Given an input $r \in \{0,1\}^{R(n)}$, the distinguisher D' invokes the verifier $V(x)$ using r as the random coins, and emulates the prover P^* by responding according to b^*. If the resulting transcript disagrees with (a^*,b^*) the distinguisher D' rejects. Otherwise, D' return $D(x,r,a^*,b^*)$. Clearly, D' distinguishes between $r \leftarrow\!\!\!\shortmid U_R$ to $r \leftarrow\!\!\!\shortmid G(U_S)$ with advantage $\varepsilon/2^{C_V}$. Moreover, D' can be implemented by a circuit of size $T_V + (C_V + C_P) + T_V \leqslant 3T_V$, and therefore we derive a contradiction to the pseudorandomness of G and (*) follows. □

The following claim from [AASY16] shows that good circuit PRGs can be obtained from t-wise independent hash functions. In the following we say that a family of functions $\mathcal{H} = \{h_z : X \to Y\}$ is *t-wise independent* [CW79] if for every t distinct inputs $x_1,\ldots,x_t \in X$ and uniformly chosen $h_z \leftarrow\!\!\!\shortmid \mathcal{H}$, the random variable $(h_z(x_1),\ldots,h_z(x_t))$ is uniformly distributed over Y^t.

Claim 3.4 (PRGs from hash functions (Claim 5.2 in [AASY16]))**.** *For every T and $\varepsilon,\delta \in [0,1]$, and every family $\mathcal{H} = \{h_z : \{0,1\}^s \to \{0,1\}^r\}$ of t-wise independent hash functions with $t = 4T\log T + 2\log(1/\delta)$ and $s = 2\log(1/\varepsilon) + \log t$ the following holds. With probability $1-\delta$, a random member $h_z \leftarrow\!\!\!\shortmid \mathcal{H}$ ε-fools any T-size circuit.*

By combining Claim 3.4 with Lemma 3.3 we derive the following theorem.

Theorem 3.5 (Fooling protocols via hashing)**.** *For every functions $T_V(n), C_V(n), C_P(n), R(n) : \mathbb{N} \to \mathbb{N}$ and $\epsilon, \delta : \mathbb{N} \to [0,1]$, the following holds for every family $\mathcal{H} = \left\{h_z : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}\right\}$ of t-wise independent hash functions where $t = O(T_V \log T_V + \log(1/\delta))$ and $S = 2C_V + 2\log(1/\varepsilon) + \log T_V + \log\log T_V + \log\log(1/\delta) + O(1)$, it holds that*

$$\Pr_{h_z \leftarrow \mathcal{H}}[h_z \text{ strongly } \varepsilon\text{-fools non-uniform } \mathsf{IP}[R, C_V, T_V, C_P]] > 1 - \delta.$$

Remark 3.6 (Canonical construction of t-wise independent hash functions)**.** *Throughout the paper we use the following standard construction of t-wise independent hash functions $\mathcal{H} = \left\{h_z : \{0,1\}^S \to \{0,1\}^R\right\}$ where $t < 2^S < 2^R$. Let $\mathbb{F} = GF(2^R)$ denote the finite field of 2^R elements. We identify field elements with binary strings of length R via some canonical representation that supports arithmetic operations with a computational cost of $\widetilde{O}(R)$ bit operations (For instance [Sil99]). It is well known [CW79] that the family $\mathcal{H}' = \{h'_z : \mathbb{F} \to \mathbb{F}\}_{z \in \mathbb{F}^t}$ where h'_z denotes the degree-t univariate polynomial whose coefficients are given by the vector $z \in \mathbb{F}^t$ is a family of t-wise independent hash functions from $\{0,1\}^R$ to $\{0,1\}^R$. We define $\mathcal{H}$ by restricting the domain of $\mathcal{H}$ to some fixed 2^S subset. Specifically, Let h_z denote the function that takes an input $x \in \{0,1\}^S$, maps it to $\mathbb{F}$ by padding it with $R - S$ zeroes, and outputs $h'_z(x)$. Then, $\mathcal{H} = \{h_z\}_{z \in \mathbb{F}^t}$ is a t-wise independent family. Observe that one can sample an index z by sampling a tR random bits, and that given z and $x \in \{0,1\}^S$ we can evaluate $h_z(x)$ by making $O(t)$ arithmetic operations. Hence the total bit complexity of sampling and evaluating a function in $\mathcal{H}$ is $\widetilde{O}(tR)$.*

By hard-wiring a "good" hash function as a non-uniform advice to Theorem 3.5 we derive the following corollary (that strengthens Theorem 1.2 from the introduction.).

Corollary 3.7. *For every functions $T_V(n), C_V(n), C_P(n), R(n) : \mathbb{N} \to \mathbb{N}$ and $\epsilon : \mathbb{N} \to [0,1]$, there exists a $\mathsf{PRG} : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ that can be computed by a non-uniform $\widetilde{O}(RT_V)$-time and strongly ε-fools non-uniform $\mathsf{IP}[R, C_V, T_V, C_P]$ where $S = 2C_V + 2\log(1/\varepsilon) + \log T_V + \log\log T_V + O(1)$.*

Theorem 1.3 follows immediately.

The amortized setting. For a promise problem $\Pi = (\Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}})$ and a polynomial $k(\cdot)$ define the problem $\Pi^k = (\Pi'_{\mathsf{yes}}, \Pi'_{\mathsf{no}})$ by letting Π'_{yes} denote the set of all tuples $\vec{x} = (x_1, \ldots, x_{k(n)}) \in (\{0,1\}^n)^{k(n)}$ such that $x_i \in \Pi_{\mathsf{yes}}$ for every i and by letting Π'_{no} denote the set of tuples $\vec{x} = (x_1, \ldots, x_{k(n)}) \in (\{0,1\}^n)^{k(n)}$ such that, for every i, $x_i \in \Pi_{\mathsf{yes}} \cup \Pi_{\mathsf{no}}$ and for at least one i, $x_i \in \Pi_{\mathsf{no}}$.

Corollary 3.8 (Uniform Amortized sparsafication of many instances)**.** *Let Π be a promise problem that admits a uniform $\mathsf{IP}[R, C_V, T_V, C_P]$ proof system with negligible soundness and correctness errors. Then, for every polynomial $k(\cdot)$, the promise problem Π^k admits a (uniform) $\mathsf{IP}[R', C'_V = kC_V, T'_V, C'_P = kC_P]$ proof system with constant error where $R' = R \cdot \widetilde{O}(T_V) + O(k(C_V + \log k + \log T_V))$ and $T'_V = k\widetilde{O}(T_V R)$.*

So for sufficiently large k, the amortized randomness complexity R'/k is $O(C_V + \log k + \log T_V)$ per instance.

Proof. Let $\varepsilon = 1/(10k)$ and $\delta = 0.1$. Let $\mathcal{H}$ be a family of t-wise hash function that expand S bits to R bits where $t = O(T_V \log T_V)$ and $S = 2C_V + 2\log(1/\varepsilon) + \log T_V + \log\log T_V + \log\log(1/\delta) + O(1) \leqslant 2C_V + 2\log k + 2\log T_V + O(1)$. The verifier samples a function $h_z \leftarrow \mathcal{H}$ and given $\vec{x} = (x_1, \ldots, x_{k(n)})$ applies, for each i, the original verifier $V(x_i; h_z(s_i))$ where s_i is chosen uniformly and independently from

U_S. At the end, we accept if and only if all interactions accepted. The prover simply runs the original protocol k times.

By Theorem 3.5, with probability $1-\delta$ the hash function h_z ε-fools the original protocol. Therefore, conditioned on this event, the error in each instance is at most $\varepsilon + n^{-\omega(1)}$, and by a union bound the total error is at most $\delta + k\varepsilon + kn^{-\omega(1)} \leqslant 0.2$, as required. The communication grows by a factor of k, the randomness complexity is $O(tR)$ for sampling the hash function (Remark 3.6) plus $O(kS)$ for sampling the seeds. The computational complexity for sampling h_z is $\widetilde{O}(tR)$ and each instance has an additional cost of $T_V + \widetilde{O}(tR)$ (again see Remark 3.6). □

4 Uniform Randomness Sparsification for Constant-Round Protocols

In this section we extend the randomness reduction seen in the previous section from the non-uniform setting to the uniform setting. Recall that in the previous section we reduced the randomness of general IP proofs by using a non-uniform advice that consisted of a description of a "good" hash function that can be used as a PRG. As explained in Section 1.2 we cannot afford to to sample the hash function uniformly since this requires too much randomness (larger than the amount of randomness that is needed for the original protocol). Instead, we describe a randomness-efficient method for sampling a "good" hash function via a uniform algorithm by reducing the problem to a more standard de-randomization problem. We further show that for constant number of rounds, the latter problem can be solved under standard complexity-theoretic assumptions.

We begin by defining a promise problem whose no-instances corresponds to hash functions that "fool a given protocol" and its "yes" instances are hash functions that "fail to fool the protocol".

Definition 4.1. *Let $\langle P, V\rangle$ be a k-round (possibly non-uniform) $\mathsf{IP}[R, C_V, C_P, T_V]$ protocol for a promise problem L with a polynomial-time verifier, and let $\varepsilon(n)$ be some inverse polynomial. Fix some efficiently computable family of hash functions $\mathcal{H} = \left\{h_z : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}\right\}_{z\in\{0,1\}^{Z(n)}}$ that satisfies Theorem 3.5 with respect to $\mathsf{IP}[R, C_V, C_P, T_V]$ protocols where the underlying parameters ε, δ are taken both to be $\varepsilon(n)$. We define a promise problem $\Pi = \Pi_{P,V,\varepsilon}$ over strings $z \in \{0,1\}^*$ as follows:*

- *The set of yes instances, Π_{yes}, consists of all strings z such that h_z does* not *strongly 2ε-fools $\langle P, V\rangle$.*
- *The set of no instances, Π_{no}, consists of all strings z such that h_z strongly ε-fools $\langle P, V\rangle$.*

We prove the following lemma.

Lemma 4.2 ($\Pi_{P,V,\varepsilon} \in \mathsf{IP}_{k+1}$)**.** *For any k-message protocol $\langle P, V\rangle$ (resp., non-uniform protocol $\langle P, V\rangle$) and inverse polynomial ε the promise problem $\Pi_{P,V,\varepsilon}$ is in IP_{k+1} (resp., $\mathsf{IP}/\mathsf{poly}_{k+1}$) and the computational complexity of the corresponding verifier is $O(T_V + T_H)$ where T_V is the complexity of V and T_H is the computational complexity of universal evaluation of $\mathcal{H}$. Consequently, for constant k, $\Pi_{P,V,\varepsilon}$ is in AM (resp., in $\mathsf{AM}/\mathsf{poly}$).*

Proof. On a shared input $z \in \{0,1\}^*$, the prover will try to convince the verifier that h_z does **not** strongly 2ε-fools $\langle P, V\rangle$. Recall that this means that one of the following holds for some input x:

- (Case 0:) There exists P^* Strategy such that $\Pr[(V^{h_z}, P^*)(x) = 1] - \Pr[(V, P^*)(x) = 1] > 2\varepsilon$.

- (Case 1:) There exists P^* Strategy such that $\Pr[(V, P^*)(x) = 1] - \Pr[(V^{h_z}, P^*)(x) = 1] > 2\varepsilon$.

Accordingly, the prover first declares x and whether case (0) or case (1) holds and then proceeds to prove its claim via an interactive proof. Specifically, on common input $(1^n, z)$ the parties invoke the following $(k+1)$-move protocol.

1. The prover finds an input $x \in \{0,1\}^n$ and a proof strategy P^* such that Case $c \in \{0,1\}$ holds.
   The prover sends x and c.
2. The verifier samples two strings, $r_0 \leftarrow\!\!\shortmid U_R$, $r_1 \leftarrow\!\!\shortmid h_z(U_S)$ and a random bit $b \in \{0,1\}$.
   The two parties invoke an interactive protocol where the prover plays $P^*(x)$ and the verifier plays $V(x; r_b)$.
   Let $v \in \{0,1\}$ denote the output (acceptance bit) of $V(x; r_b)$.
3. The verifier accepts if $b = v \oplus c$.

Completeness: Assume that h_z does not strongly 2ε-fool $\langle P, V\rangle$ and let us assume that case (0) holds. (The other case is proved symmetrically.) Then, the probability that the verifier accepts is

$$\frac{1}{2}\Pr[(P^*, V^{h_z})(x) = 1] + \frac{1}{2}(1 - \Pr[(P^*, V)(x) = 1]) \geqslant \frac{1}{2} + \varepsilon.$$

Soundness: Fix some no instance z for which h_z strongly ε-fool $\langle P, V\rangle$. We analyze the acceptance probability of the verifier when interacting with a cheating prover. Fix an arbitrary first message (x, c) of the prover and let us denote by P^* the strategy that the prover plays in Step 2 of the protocol. Since h_z strongly ε-fool $\langle P, V\rangle$, it holds that the difference between the quantities

$$q = \Pr[(P^*, V)(x) = 1] \qquad \text{and} \qquad q_z = \Pr[(P^*, V^{h_z})(x) = 1]$$

is at most ε in absolute value. Suppose that $c = 0$ (the other case is symmetric). Then, the verifier accepts with probability

$$\frac{1}{2}q_z + \frac{1}{2}(1 - q) \leqslant 1/2 + \varepsilon/2,$$

as required.

Overall, the protocol has completeness of $1/2+\varepsilon$ and soundness of $1/2+\varepsilon/2$. Since $\varepsilon = \Omega(1/\mathrm{poly}(n))$, we can use standard parallel amplification theorems to reduce the error (cf. [Gol98, Appendix A]). This completes the proof of the first part of the lemma. The "Consequently" part, follows from the equivalence between constant-round IP protocols and AM proofs [GS86, BM88]. □

We will make use of the following result.

Theorem 4.3 (PRGs against $\mathsf{AM}/\mathsf{poly}$ [IW97, KVM02, SU06])**.** *Suppose that $E = \mathsf{DTime}(2^{O(n)})$ is hard for exponential-size non-deterministic circuits*[10] *, i.e., there exists a language L in E and a constant $\beta > 0$, such that for every sufficiently large n, circuits of size $2^{\beta n}$ fail to compute the characteristic function of L on inputs of length n.*

[10] A non-deterministic circuit C has additional "non-deterministic input wires". Such a circuit evaluates to 1 on x if and only if there exist an assignment to the non-deterministic input wires that makes C output 1 on x. Non-Deterministic circuits can be therefore viewed as a non-uniform version of the class NP.

Then for every polynomial $T(\cdot)$ and inverse polynomial $\varepsilon(\cdot)$, there exists a pseudo-random generator G that stretches seeds of length $\rho = O(\log m)$ into a string of length m in time $\mathrm{poly}(m)$ such that G ε-fools every promise problem $\Pi = (\Pi_{\mathsf{yes}}, \Pi_{\mathsf{no}})$ that admits an $\mathsf{AM}/\mathsf{poly}$ proof system with a T-size verifier in the following sense. For every sufficiently large m and $b \in \{\mathsf{yes}, \mathsf{no}\}$

$$\left| \Pr_{z \leftarrow U_m}[z \in \Pi_b] - \Pr_{z \leftarrow G(U_\rho)}[z \in \Pi_b] \right| \leqslant \varepsilon(m).$$

By combining the above theorem with Lemma 4.2, we derive the following corollary.

Corollary 4.4 (uniform PRG against constant-round IP protocols)**.** *Under the assumption of Theorem 4.3 for every polynomials $T_V(n), C_V(n), C_P(n), R(n) : \mathbb{N} \to \mathbb{N}$, constant $k \in \mathbb{N}$ and inverse polynomial $\varepsilon : \mathbb{N} \to [0,1]$ there exists a polynomial-time computable PRG that strongly ε-fools non-uniform $\mathsf{IP}_k[R, C_V, C_P, T_V]$ with seed length of $2C_V + O(\log n)$.*

Proof. Let $\varepsilon' = \varepsilon/4$. Fix some non-uniform $\langle P, V \rangle$ interactive proof in $\mathsf{IP}_k[R, C_V, C_P, T_V]$ and let $\Pi = \Pi_{P,V,\varepsilon'}$ denote the corresponding promise problem defined in Definition 4.1. Recall that

$$\mathcal{H} = \left\{ h_z : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)} \right\}_{z \in \{0,1\}^{Z(n)}}$$

is a family of t-wise independent hash functions where $t = O(T_V \log T_V + \log(1/\varepsilon'))$ and $S = 2C_V + 2\log(1/\varepsilon') + \log T_V + \log\log T_V + \log\log(1/\varepsilon') + O(1)$ that can be evaluated by a $\mathrm{poly}(n)$-time universal evaluation algorithm $H : \{0,1\}^{Z(n)} \times \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$. As shown in Lemma 4.2, the promise problem Π is in $\mathsf{AM}/\mathsf{poly}$. Let us denote by $T(n)$ the time complexity of the verifier in the corresponding proof system (and recall that $T = O(T_V + T_H)$ and so it depends only on ε, R, C_V, C_P and T_V). Let $G' : \{0,1\}^{\rho(n)} \to \{0,1\}^{Z(n)}$ be the PRG that ε'-fools $\mathsf{AM}/\mathsf{poly}$ problems with T-time verifiers as promised in Theorem 4.3. Recall that $\rho(n) = O(\log Z(n)) = O(\log n)$.

We define the PRG against non-uniform $\mathsf{IP}_k[R, C_V, C_P, T_V]$ that maps a random seed of length $\rho(n) + S(n)$ into a pseudorandom string of length $R(n)$ as follows. Given a seed (s_1, s_2) where $s_1 \in \{0,1\}^{\rho(n)}$ and $s_2 \in \{0,1\}^{S(n)}$, output $H(G'(s_1), s_2) = h_{G'(s_1)}(s_2)$. Note that PRG is indeed efficiently computable and that its definition depends only in the parameters R, C_V, C_P, T_V and ε. We prove that PRG strongly ε-fools $\langle P, V \rangle$. For this it suffices to show that, except with probability $\varepsilon/2$, over the choice of s_1, it holds that $h_{G'(s_1)}(s_2)$ strongly ε-fools $\langle P, V \rangle$. Indeed,

$$\Pr_{s_1}[G'(s_1) \in \Pi_{\mathsf{no}}] \geqslant \Pr_{z}[z \in \Pi_{\mathsf{no}}] - \varepsilon' \geqslant 1 - 2\varepsilon' \geqslant 1 - \varepsilon/2$$

where the first inequality follows from the pseudo-randomness of G' and the second inequality follows from Theorem 3.5. The corollary follows. □

Theorem 1.4 follows immediately from Corollary 4.4

5 Zero Knowledge Proofs

In this section we study the problem of randomness sparsification for zero-knowledge proof systems.

5.1 SZK proof systems

We begin by noting that PRG-based sparsification trivially preserves zero-knowledge against malicious verifier.

Observation 5.1. *If $\langle P, V\rangle$ is a constant-round* SZK *proof system and G ε-fools $\langle P, V\rangle$ then $\langle P, V^G\rangle$ is an* SZK *proof system whose soundness error and completeness error increase by ε. Moreover, if $\langle P, V\rangle$ has perfect completeness the so is $\langle P, V^G\rangle$.*

Proof. By Observation 3.2, the system $\langle P, V^G\rangle$ is an interactive proof system with the desired parameters. Since zero-knowledge against cheating verifier is a property of P the new system is also zero-knowledge. □

By combining the above observation with Corollary 4.4 we derive Theorem 1.7.

5.2 Semi-Malicious SZK Proof Systems

We move on to study sparsification for semi-malicious SZK proof systems. We begin by introducing this new variant of zero-knowledge.

Definition 5.2 (F semi-malicious SZK)**.** *Let $F : \mathbb{N} \to \mathbb{N}$ be an integer valued function and let $\langle P, V\rangle$ be a proof system with randomness complexity of R for a promise problem Π. Let $D(1^n; s)$ be an efficiently-computable algorithm that given randomness s outputs $F(n)$ bits. Define the verifier $V_D(x)$ as follows:*

- *Sample random coins s for D, and compute the $F(|x|)$-bit string $f = D(1^{|x|}, s)$.*
- *Sample $r' \hookleftarrow \{0,1\}^{R(|x|)-F(|x|)}$.*
- *invoke $V(x)$ on the concatenated random tape $f \circ r'$.*

Let $\mu(D)$ denote the completeness error of the proof system $\langle P, V_D\rangle$ with respect to Π, and let $\mathsf{view}_{V_D}(x)$ denote the random variable that corresponds to the view of the verifier $V_D(x)$ when interacting with P on a common input x.

We say that $\langle P, V\rangle$ is F semi-malicious zero-knowledge proof system *with zero-knowledge error of δ_z, abbreviated (F, δ_z)-*SMSZK*, if for every efficiently-computable algorithm $D(1^n; s)$ there exists a simulator Sim_D that runs in expected polynomial-time such that for every yes instance x,*

$$\mathsf{SD}(\mathsf{Sim}_D(x), \mathsf{view}_{V_D}(x)) \leqslant \delta_z + \mu(D). \tag{1}$$

*By default, we assume that δ_z is negligible and in this case we refer $\langle P, V\rangle$ as F-*SMSZK *proof system. The notion of F* semi-malicious perfect zero-knowledge proof system *(F-*SMPZK *is short) is defined analogously, except that the simulator's deviation in (1) must be zero. We refer to the F-bit prefix of the verifier's tape as the* accessible *bits.*

Remark 5.3 (On the additive term $\mu(D)$)**.** *One could consider a more restrictive definition of F-*SMSZK *in which the deviation of the simulator Sim_D is bounded by δ_z regardless of the completeness error $\mu(D)$ of D. While our reductions are compatible with this alternative variant as well, we choose to employ the*

current definition since it is more liberal. Further note that the additive term $\mu(D)$ intuitively allows the simulator to deviate when the protocol outputs non-accepting transcripts. Thus, one can roughly think of our definition as restricting the attention to semi-malicious distributions D that put most of their mass on strategies for which completeness hold.

Observe that $0 \leqslant F \leqslant R$ and that any HVSZK proof system is also an 0-SMSZK and every SZK proof system is R-SMSZK. On the other hand, as mentioned in Section 1.2.2, the classical HVSZK proof system for the complete *statistical-distance* problem of [SV03] can be shown to have maximal accessible bit complexity of $F = R$ too. (See Section B.) Thus, even $R -$ SMSZK complexity is a weaker notion than SZK complexity.

Remark 5.4. *One can use a more general definition in which the "accessible bits" are not necessarily the first ones and can be taken to be any set of $F(|x|)$ indices that can be efficiently computable and possibly depend on the input x itself. However, in this case one can always modify the verifier (by pre-permuting the random tape) and make sure that the accessible bits are located in the first $F(|x|)$ indices.*

Remark 5.5. *Typical* SMSZK *systems (e.g., for statistical-distance [SV03] or for GNI [GMW91]) satisfy the following stronger definition. There exists a "universal" simulator* Sim *such that for every yes instance x and every fixing $f \in \{0,1\}^{F(|x|)}$ of the first $F(|x|)$ bits of the verifier, the distribution $\mathsf{Sim}(x, f)$ is $(\delta_z + \mu(f))$-close, in statistical-distance, to the view of V_f when interacting with P on the input x, where V_f denotes the verifier that given an input x and a random tape $r' \leftarrow \{0,1\}^{R(|x|)-F(|x|)}$ invokes $V(x)$ on the concatenated random tape $f \circ r'$.*

5.3 SMSZK: Randomness vs. Prover's Communication/CRS

Theorem 5.6. *Let $\langle P, V\rangle$ be an (F, δ_z)-$\mathsf{SMSZK}_k[R, C_V, C_P, T_V]$ proof system for the promise problem Π. Suppose that $G : \{0,1\}^S \to \{0,1\}^R$ ε-fools non-uniform $\mathsf{IP}[R, C_V, C_P, T_V]$ protocols. Consider the following proof system $\langle P', V'\rangle$ that on shared input x of length n proceeds as follows:*

1. *P' sends a random message a of length $R - F$ where $R = R(n)$ and $F = F(n)$.*
2. *The verifier reads his random tape $s \leftarrow U_{S(n)}$, computes $r_2 = G(s)$, expands a to an R-bit string $r_1 = 0^F \circ a$ and sets $r = r_1 \oplus r_2$. From now on, the prover plays $P(x)$ and verifier plays $V(x; r)$.*

Then, $\langle P', V'\rangle$ is an HVSZK_{k+1} proof system with zero-knowledge error of $\delta_z + \varepsilon$ and an ε additive penalty in the correctness and soundness error.

Proof. We begin by showing that $\langle P', V'\rangle$ is an IP_{k+1} proof system for Π. For any fixing of $a \in \{0,1\}^{R-F}$, define the proof system $\langle P_a, V_a\rangle$ in which the verifier expands a to r_1 like in the above description, samples r_2 uniformly and calls $V(x; r_1 \oplus r_2)$ and the prover operates as before. Clearly, the soundness and correctness of this system is the same as the original one. Next, define the a-residual proof system $\langle P'_a, V'_a\rangle$ which is identical to the *sparsified* system $\langle P', V'\rangle$ except that a is hard-coded into V' who skips the first step of the above protocol. The proof system $\langle P'_a, V'_a\rangle$ is the G-sparsified version of $\langle P_a, V_a\rangle$, and since G ε-fools non-uniform proof systems, the system $\langle P'_a, V'_a\rangle$ is sound and complete (with an additive error of ε). Since this is true for every choice of a, it follows that $\langle P', V'\rangle$ is an IP_{k+1} proof system for Π.

Let $D(1^n; s)$ be the algorithm that samples $s \leftarrow U_{S(n)}$ and outputs the $F(n)$-bit prefix of $G(s)$, and let Sim_D denote the simulator of the original F-$\mathsf{SMSZK}_k[R, C_V, C_P, T_V]$ proof system with respect to the

distribution $D(1^n)$. We define a simulator Sim' for $\langle P', V' \rangle$ that, on an input x of length n, operates as follows:

1. Let $S = S(n), R = R(n)$ and $F = F(n)$. Invoke $\mathsf{Sim}_D(x)$ and sample a view (x, s', α, c') where s' is the S-bit seed sampled for D, $\alpha \in \{0,1\}^{R-F}$ form the uniform part of the verifier's random tape, and c' is the (simulated) sequence of incoming messages.
2. Compute $r'_2 = G(s)$ and set $a' \in \{0,1\}^{R-F}$ to be the XOR of α with the $(R-F)$-bit suffix of r'_2.
3. Output the tuple (x, s', a', c').

Fix a yes instance x. We analyze the statistical distance between the simulated tuple (x, s', a', c') and the "real" tuple (x, s, a, c) that corresponds to the distribution of the real view of V' when interacting with P. It suffices to show that if the original simulator is perfect the two distributions are identical. (Indeed, since the new simulator makes a single call to the original simulator, a deviation of $\delta_z + \varepsilon$ of the original simulator can increase the statistical distance of the new one by at most $\delta_z + \varepsilon$.)

First observe that in both experiments s and s' are distributed uniformly. Fix some value for $s = s'$, and consider the conditional distributions $[(a', c')|s']$ and $[(a, c)|s]$. Next observe that a is uniform and that a' is uniform as well (since α is uniform). Finally, conditioned on $(s, a) = (s', a')$ the transcript c is sampled according to the experiment $\langle P, V \rangle (x; r)$ where $r = (0^F \circ a) \oplus G(s)$ and similarly the simulated transcript c' is sampled according to the experiment $\langle P, V \rangle (x; r')$ where $r' = (0^F \circ a') \oplus G(s') = (G(s')[1:F] \circ \alpha)$ and so the tuples are identically distributed. □

By combining Theorem 5.6 with Corollary 4.4 we derive the following corollary which implies the first part of Theorem 1.8 from the introduction.

Corollary 5.7 (Trading randomness with prover's communication for SMSZK)**.** *Suppose that* E *is hard for exponential size non-deterministic circuits. Then, for every inverse polynomial* ε*, every constant-round* (F, δ_z)-$\mathsf{SMSZK}_k[R, C_V, C_P, T_V]$ *proof system can be transformed into a new*

$$\mathsf{HVSZK}_{k+1}[R' = 2C_V + O(\log n), C_V, T'_V = \widetilde{O}(T_V \cdot (R + \log n)), C'_P = C_P + R - F]$$

system with an additive penalty of ε *in the soundness and completeness error and an additive penalty of* $\varepsilon + \delta_z$ *in the simulation error. Specifically, the new protocol consists of an additional preliminary message from the prover that consists of a random string of length* $R - F$ *bits. Moreover, the transformation preserves perfect completeness, and if the original proof system is semi-malicious* perfect *zero-knowledge then the resulting scheme admits a perfect simulation (i.e., it is* $\mathsf{HVPZK}_{k+1}[R', C_V, T'_V, C'_P]$*).*

Remark 5.8. *Corollary 5.7 can be converted to a statement regarding* HVSZK *in the common reference string model by replacing the first message of the prover with a common reference string* ρ*. This CRS can be chosen by the prover (a malicious choice does not affect the soundness). However, the CRS is* not *reusable among several invocations.*

5.4 SMPZK: Randomness vs. Simulation Complexity

In the perfect setting, SMPZK proof systems can be sparsified at the expense of slowing-down the simulation by a factor of 2^{R-F}.

Lemma 5.9. *Let $\langle P, V\rangle$ be an F-SMPZK proof system for a promise problem Π. Let $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ be a $\mathrm{poly}(n)$-time computable function that ε-fools $\langle P, V\rangle$. Then the protocol $\langle P, V^G\rangle$ is a proof system with an additive penalty of ε in soundness and completeness errors that has a perfect honest-verifier simulator Sim' with expected running-time of $(\mathrm{poly}(n))2^{R(n)-F(n)}$.*

Proof. Let $D(1^n; s)$ be the algorithm that samples $s \leftarrow\!\!\!\!\rightarrow U_{S(n)}$ and outputs the $F(n)$-bit prefix of $G(s)$, and let Sim_D denote the simulator of the original F-SMPZK proof system with respect to the distribution $D(1^n)$. The view of V^D in interaction with P over a yes-instance $x \in \{0,1\}^n$ is parsed into (x, s, β, c) where $s \leftarrow\!\!\!\!\rightarrow U_{S(n)}$, $\beta \leftarrow\!\!\!\!\rightarrow U_{R(n)-F(n)}$ and c is the vector of incoming messages. We define a new simulator $\mathsf{Sim}'(x)$ as follows: (1) Sample $(x; s', \beta', c')$ by invoking $\mathsf{Sim}_D(x)$ (2) If the last $R(n) - F(n)$ bits of $G(s')$ equal to β' output the transcript $(x; s', \beta', c')$ and halt; otherwise, goto (1).

Since β' is uniformly distributed, at each iteration Sim' halts with probability $2^{F(n)-R(n)}$, and so the expected running time is $\mathrm{poly}(n)2^{R(n)-F(n)}$. Perfect simulation follows by noting that (s', β') are distributed identically to the random tape of V^G and that conditioned on every fixing of these coins, (s, β), the simulated transcript c' is distributed just like a real interaction between $P(x)$ and $V^G(x; s, \beta)$ (since Sim_D is a perfect simulator). □

By combining Lemma 5.9 with Corollary 4.4 we derive the following corollary which implies the second part of Theorem 1.8 from the introduction.

Corollary 5.10. *Assuming that E is hard for exponential size non-deterministic circuits, let $\varepsilon : \mathbb{N} \to [0,1]$ be an inverse polynomial and $R, C_V, C_P, T_V : \mathbb{N} \to \mathbb{N}$ be polynomially bounded functions where $C_v = \omega(\log n)$. Suppose that the promise problem Π admits a constant-round F-SMPZK$_k[R, C_V, C_P, T_V]$ proof system. Then Π admits an $\mathsf{IP}_k[R' = 2C_V + O(\log n), C_V, C_P, T_V]$ proof system with an honest-verifier perfect simulator that runs in expected time of $\mathrm{poly}(n)2^{R-F}$ and with ε penalty in the soundness and completeness errors.*

5.5 HVPZK: Randomness vs. Simulation Complexity

Corollary 5.10 shows that F-SMPZK systems can be sparsified with a simulation slow-down of 2^{R-F}. In this section we describe a different simulation strategy that yields a slow-down of 2^{R-S} where S is the seed-length of the PRG. This holds even when $F = 0$, i.e., for HVPZK proof systems. This theorem is based on a PRG that satisfies some additional features (e.g., regularity and the existence of an efficient inversion algorithm). We later show that our PRGs meet these requirements.

Definition 5.11. *We say that a function $G : \{0,1\}^S \to \{0,1\}^R$ is δ-*regular *if $G(U_S)$ is δ-close in statistical distance to $U(\mathsf{Image}(G))$, the uniform over the image of G. (In particular, a 0-regular function maps the same number of inputs to each of its outputs.) A* uniform inversion *algorithm for G is a randomized algorithm that given an input $y \in \{0,1\}^R$ outputs $\perp$ if y is not in the image of G, and, otherwise, outputs a uniformly chosen preimage of y under G.*

Lemma 5.12. *Let $\langle P, V\rangle$ be an HVPZK proof system for a promise problem Π whose simulator Sim runs in time T_{Sim}. Let $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ be a $\mathrm{poly}(n)$-time computable function that ε-fools $\langle P, V\rangle$, can be uniformly inverted in expected time of $T_{G^{-1}}$, and is δ-regular. Then the protocol $\langle P, V^G\rangle$ is a proof system with an additive penalty of ε in soundness and completeness errors and with an honest-verifier simulator Sim' with statistical deviation of δ and expected running-time of $(T_{\mathsf{Sim}} + T_{G^{-1}})\frac{2^R}{|\mathsf{Image}(G)|}$.*

Proof. For a given instance x, the view of the original verifier V can be parsed to (x, r, v) where r is the randomness and v is the transcript. Let us parse the view of V^G (in an interaction $(P, V^G)(x)$) as a tuple (x, s, r, v) where s is the seed $r = G(s)$ and v is the transcript v. (While r is redundant it will be useful to keep it as part of the view.) The simulator $\mathsf{Sim}'(x)$ does the following: (1) Sample (r', v') by calling $\mathsf{Sim}(x)$; (2) Call the G-inverter on r' and denote its output by s'. If the output is $\perp$ output $\perp$; otherwise, output the tuple (x, s', r', v').

Let us analyze the statistical deviation of Sim'. Fix some yes instance x and consider the distribution (x, s, r, v) in the real interaction $(P, V^G)(x)$. Observe that, conditioned on $r = r'$ the simulated tuple (x, s', r', v') is distributed identically to the real distribution (x, s, r, v). Indeed, in both cases s is uniform preimage of r and v is a random transcript that corresponds to an interaction between $P(x)$ and $V(x; G(s))$. Therefore, the statistical distance between the simulated view (conditioned on not outputting $\perp$) and the real view is exactly the statistical distance between $r = G(U_S)$ and $r' = U(\mathsf{Image}(G))$ which is at most δ since G is δ-regular. Finally, observe that the success probability (that r' hits $\mathsf{Image}(G)$) is exactly $|\mathsf{Image}(G)|/2^R$, and so the expected number of iterations is $2^R/|\mathsf{Image}(G)|$ as required. □

We move on and show that our PRGs are invertable and almost-uniform.

Proposition 5.13. *Let $k \in \mathbb{N}$ be a constant, $R, C_V, C_P, T_V : \mathbb{N} \to \mathbb{N}$ polynomially-bounded functions and $\varepsilon : \mathbb{N} \to [0, 1]$ be an inverse polynomial. Let $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ be the uniform PRG (resp., non-uniform PRG) that ε-fools non-uniform $\mathsf{IP}_k[R, C_V, C_P, T_V]$ (resp., non-uniform $\mathsf{IP}[R, C_V, C_P, T_V]$) that is promised by Corollary 4.4 (resp., Corollary 3.7). Then G is $(\mathrm{poly}(n)2^{-S(n)})$-regular, the image of G, on n-bit inputs, consists of at least $2^{S(n)}/\mathrm{poly}(n)$ strings and there is an algorithm that, given the description of G, uniformly inverts G in expected $\mathrm{poly}(n)$ time.*

Proof. We begin with the non-uniform version of G (from Corollary 3.7). As explained in Remark 3.6, G is defined by some degree-t univariate polynomial $h_z : \mathbb{F} \to \mathbb{F}$ over the field $\mathbb{F} = GF(2^R)$ and $t = \mathrm{poly}(n)$. To compute G on an input $x \in \{0,1\}^S$, we map x to a field element (by padding with $R - S$ zeroes) and output the evaluation of h_z on the padded-version of x.

Let y be a string in the image of G. First observe that the number of preimages under G is at most t since the polynomial $h_{z,y} = h_z(x) - y$ is of degree $t = \mathrm{poly}(n)$. Hence, $|\mathsf{Image}(G)| \geqslant 2^S/\mathrm{poly}(n)$ and $G(U_S)$ samples every element $y \in \mathsf{Image}(G)$ with probability $p_y \in [1/|\mathsf{Image}(G)|, t/|\mathsf{Image}(G)|]$. Since $U(\mathsf{Image}(G))$ samples each element from $\mathsf{Image}(G)$ with weight $1/|\mathsf{Image}(G)|$, it follows that G is δ regular for $\delta = O(t/|\mathsf{Image}(G)|) = \mathrm{poly}(n)/2^{S(n)}$.

Next, observe that there exists a randomized algorithm A that given y lists in expected time of $T_A = \mathrm{poly}(t, R) = \mathrm{poly}(n)$ all the pre-images of y under G. (This can be done, for example, by factoring $h_{z,y}$ to its irreducible components via the algorithm of [CZ81] and by noting that, for each root a of $h_{z,y}$, the polynomial $x - a$ must appear in the factorization.) We can therefore sample a random preimage in expected-polynomial time.

We move on to the uniform setting. Recall that in this setting (Corollary 4.4), the PRG G is defined as follows: (1) Sample a short seed s_1 of length $O(\log n)$ and a long seed s_2 of length $S - O(\log n)$; (2) Feed the short seed s_1 into a PRG G_1 that fools $\mathsf{AM}/\mathsf{poly}$ languages (with properly chosen parameters) and use the resulting string $z = G_1(s_1)$ to select a degree-t univariate polynomial $h_z : \mathbb{F} \to \mathbb{F}$ over the field $\mathbb{F} = GF(2^R)$ where $t = \mathrm{poly}(n)$ as before; (3) Output $h_z(s_2)$.

It follows that each point in the image of G has at most $t \cdot |\mathsf{Image}(G_1)| \leqslant \mathrm{poly}(n)$ preimages. Therefore,

$|\mathsf{Image}(G)| \geqslant 2^{S(n)}/\mathrm{poly}(n)$ and G is δ-regular for $\delta = O(\mathrm{poly}(n)/2^S)$. Finally, in order to uniformly invert $y \in \mathsf{Image}(G)$ we compute, for every s_1, the list $L_{s_1} = \left\{(s_1, s_2) : h_{G_1(s_1)}(s_2) = y\right\}$ (using the aforementioned algorithm for h_z where $z = G_1(s_1)$), and then sample a preimage (s_1, s_2) uniformly from the union of all these (polynomially-many) lists. The expected running time is $O(2^{|s_1|}\mathrm{poly}(n)) = \mathrm{poly}(n)$, as required. □

By combining Lemma 5.12 and Proposition 5.13, we derive the following corollary.

Corollary 5.14. *Assuming that* E *is hard for exponential size non-deterministic circuits, let* $\varepsilon : \mathbb{N} \to [0, 1]$ *be an inverse polynomial and* $R, C_V, C_P, T_V : \mathbb{N} \to \mathbb{N}$ *be polynomially bounded functions where* $C_v = \omega(\log n)$. *Suppose that the promise problem* Π *admits a constant-round* $\mathsf{HVPZK}_k[R, C_V, C_P, T_V]$ *proof system. Then* Π *admits an* $\mathsf{IP}_k[R' = 2C_V + O(\log n), C_V, C_P, T_V]$ *proof system with an honest-verifier simulator with negligible deviation error and expected running time of* $\mathrm{poly}(n)2^{R-S}$.

References

[AASY16] Benny Applebaum, Sergei Artemenko, Ronen Shaltiel, and Guang Yang. Incompressible functions, relative-error extractors, and the power of nondeterministic reductions. *computational complexity*, 25(2):349–418, 2016.

[AS17] Sergei Artemenko and Ronen Shaltiel. Pseudorandom generators with optimal seed length for non-boolean poly-size circuits. *ACM Transactions on Computation Theory (TOCT)*, 9(2):1–26, 2017.

[AV19] Benny Applebaum and Prashant Nalini Vasudevan. Placing conditional disclosure of secrets in the communication complexity universe. In *10th Innovations in Theoretical Computer Science Conference, ITCS 2019,*, pages 4:1–4:14, 2019.

[BGG93] Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Randomness in interactive proofs. *Computational Complexity*, 3(4):319–354, 1993.

[BM82] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo random bits. In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 112–117. IEEE Computer Society, 1982.

[BM88] László Babai and Shlomo Moran. Arthur-merlin games: A randomized proof system, and a hierarchy of complexity classes. *J. Comput. Syst. Sci.*, 36(2):254–276, 1988.

[BOV03] Boaz Barak, Shien Jin Ong, and Salil Vadhan. Derandomization in cryptography. In *Annual International Cryptology Conference*, pages 299–315. Springer, 2003.

[BR94] Mihir Bellare and John Rompel. Randomness-efficient oblivious sampling. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 276–287. IEEE Computer Society, 1994.

[CW79] J Lawrence Carter and Mark N Wegman. Universal classes of hash functions. *Journal of computer and system sciences*, 18(2):143–154, 1979.

[CZ81] David G Cantor and Hans Zassenhaus. A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, pages 587–592, 1981.

[DI06] Bella Dubrov and Yuval Ishai. On the randomness complexity of efficient sampling. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 711–720, 2006.

[Dru13] Andrew Drucker. Nondeterministic direct product reductions and the success probability of sat solvers. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 736–745. IEEE, 2013.

[ESY84] Shimon Even, Alan L. Selman, and Yacov Yacobi. The complexity of promise problems with applications to public-key cryptography. *Inf. Control.*, 61(2):159–173, 1984.

[FL92] Uriel Feige and Carsten Lund. On the hardness of computing the permanent of random matrices. In *Proceedings of the twenty-fourth annual ACM symposium on Theory of computing*, pages 643–654, 1992.

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In Harriet Ortiz, editor, *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 416–426. ACM, 1990.

[FY38] Ronald A Fisher and Frank Yates. *Statistical tables: For biological, agricultural and medical research*. Oliver and Boyd, 1938.

[GH98] Oded Goldreich and Johan Håstad. On the complexity of interactive proofs with bounded communication. *Inf. Process. Lett.*, 67(4):205–214, 1998.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[GMR89] Shafi Goldwasser, Silvio Micali, and Charles Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 18(1):186–208, 1989.

[GMW91] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. *Journal of the ACM (JACM)*, 38(3):690–728, 1991.

[Gol98] Oded Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*, volume 17 of *Algorithms and Combinatorics*. Springer, 1998.

[Gol01] Oded Goldreich. *The Foundations of Cryptography - Volume 1: Basic Techniques*. Cambridge University Press, 2001.

[Gol06] Oded Goldreich. On promise problems: A survey. In Oded Goldreich, Arnold L. Rosenberg, and Alan L. Selman, editors, *Theoretical Computer Science, Essays in Memory of Shimon Even*, volume 3895 of *Lecture Notes in Computer Science*, pages 254–290. Springer, 2006.

[GS86] Shafi Goldwasser and Michael Sipser. Private coins versus public coins in interactive proof systems. In *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, pages 59–68, 1986.

[GSV98] Oded Goldreich, Amit Sahai, and Salil Vadhan. Honest-verifier statistical zero-knowledge equals general statistical zero-knowledge. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 399–408, 1998.

[GVW02] Oded Goldreich, Salil Vadhan, and Avi Wigderson. On interactive proofs with a laconic prover. *Computational Complexity*, 11(1-2):1–53, 2002.

[GW02] Oded Goldreich and Avi Wigderson. Derandomization that is rarely wrong from short advice that is typically good. In *International Workshop on Randomization and Approximation Techniques in Computer Science*, pages 209–223. Springer, 2002.

[HRV18] Pavel Hubáček, Alon Rosen, and Margarita Vald. An efficiency-preserving transformation from honest-verifier statistical zero-knowledge to statistical zero-knowledge. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 66–87. Springer, 2018.

[IW97] Russell Impagliazzo and Avi Wigderson. P= bpp if e requires exponential circuits: Derandomizing the xor lemma. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 220–229, 1997.

[KVM02] Adam R Klivans and Dieter Van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002.

[LFKN92] Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992.

[LQR+19] Alex Lombardi, Willy Quach, Ron D. Rothblum, Daniel Wichs, and David J. Wu. New constructions of reusable designated-verifier nizks. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 670–700. Springer, 2019.

[New91] Ilan Newman. Private vs. common random bits in communication complexity. *Information processing letters*, 39(2):67–71, 1991.

[NH12] Koji Nuida and Goichiro Hanaoka. On the security of pseudorandomized information-theoretically secure schemes. *IEEE transactions on information theory*, 59(1):635–652, 2012.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of computer and System Sciences*, 49(2):149–167, 1994.

[Ore87] Yair Oren. On the cunning power of cheating verifiers: Some observations about zero knowledge proofs. In *28th Annual Symposium on Foundations of Computer Science (sfcs 1987)*, pages 462–471. IEEE, 1987.

[Sha92] Adi Shamir. IP = PSPACE. *J. ACM*, 39(4):869–877, 1992.

[Sil99] Joseph H Silverman. Fast multiplication in finite fields gf (2 n). In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 122–134. Springer, 1999.

[SU06] Ronen Shaltiel and Christopher Umans. Pseudorandomness for approximate counting and sampling. *computational complexity*, 15(4):298–341, 2006.

[SV03] Amit Sahai and Salil Vadhan. A complete problem for statistical zero knowledge. *Journal of the ACM (JACM)*, 50(2):196–249, 2003.

[TV00] Luca Trevisan and Salil Vadhan. Extracting randomness from samplable distributions. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 32–42. IEEE, 2000.

[Vad99] Salil Pravin Vadhan. *A study of statistical zero-knowledge proofs*. PhD thesis, Massachusetts Institute of Technology, 1999.

[Yao82] Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91. IEEE Computer Society, 1982.

A nb-PRGs do not fool interactive-proofs

A *cryptographic nb-PRG* [DI06] is a strong form of nb-PRG that fools non-boolean distinguishers with an *arbitrary polynomial* time complexity. (One can define a similar variant in the non-uniform setting by considering circuit families of an arbitrary polynomial size).

Definition A.1 (based on [DI06])**.** *Let $C(n), R(n), S(n) : \mathbb{N} \to \mathbb{N}$ be all polynomial in n and assume that $C(n) < R(n)$. A polynomial-time computable function $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ is a* C-crypto-nb-PRG *if for and every polynomial-time non-Boolean distinguisher $D : \{0,1\}^{R(n)} \to \{0,1\}^{C(n)}$ the statistical distance*

$$\mathsf{SD}(D(U_{R(n)}) \quad , \quad D(G(U_{S(n)})))$$

is negligible in n.

We show that even crypto-nb-PRG's cannot fool IP protocols in general. For this, we introduce the notion of malleable PRG. Roughly speaking, a PRG G is malleable if G can be broken by an efficient adversary that is allowed to succinctly communicate with an all powerful oracle. In more detail, G is malleable if there exists an efficient algorithm H that given a pseudorandom string r, computes a short digest of r, denoted by $h = H(r)$, and sends h to a computationally-unbounded algorithm T that sends back a short "trapdoor" $t = T(h)$ such that given t and the pseudorandom string r one can efficiently "break" the PRG and certify that it is in the image of G. We proceed with a formal definition.

Definition A.2. *A C-crypto-nb-PRG $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ is* malleable *if there exist a polynomial-time algorithm $H : \{0,1\}^{R(n)} \to \{0,1\}^{L_1(n)}$ and a computationally-unbounded algorithm $T : \{0,1\}^{L_1(n)} \to \{0,1\}^{L_2(n)}$ where $L(n) = L_1(n) + L_2(n)$ is at most $C(n)$ and a polynomial-time computable function $A : \{0,1\}^{R(n)} \times \{0,1\}^{L_2(n)} \to \{0,1\}$ such that*

$$\Pr_{r \leftarrow U_{R(n)}}[A(r, T(H(r))) = 1] = o(1) \qquad \textit{but} \qquad \Pr_{r \leftarrow G(U_{S(n)})}[A(r, T(H(r))) = 1] = 1 - o(1). \tag{2}$$

Lemma A.3. *Suppose that $G : \{0,1\}^{S(n)} \to \{0,1\}^{R(n)}$ is a malleable C-crypto-nb-PRG. Then there exists a two-message $\mathsf{IP}_2[R, C_V, T_V, C_P]$ proof-system $\langle P, V\rangle$ for the trivial language $\{0,1\}^*$ with $C_V + C_P < C$ and completeness error of $o(1)$ (and, vacuously, perfect soundness) such that $\langle P, V^G\rangle$ has completeness error of $1 - o(1)$.*

Proof. Let H, T and A be as in Definition A.2. Given an input $x \in \{0,1\}^n$, the verifier V samples $r \leftarrow\!\!\shortmid U_{R(n)}$, sends $h = H(r)$ to the prover, and receives the response $t = T(h)$. The verifier accepts if $A(r, t) = 0$. By Eq. (2), $\langle P, V\rangle$ accepts every $x \in \{0,1\}^n$ with probability $1 - o(1)$ but $\langle P, V^G\rangle$ accepts every x with probability $o(1)$. □

Constructing malleable nb-PRG. Crypto-nb-PRGs were constructed in [DI06] based on cryptographic one-way permutations (OWP) with *incompressible hardcore* bits. Roughly speaking, a hardcore bit b of a OWP f is ℓ-incompressible, if no pair of adversaries (D, E) can win with probability significantly better than $1/2$ in the following compressing game. The *compressor* D is a polynomial-time algorithm that, given $y = f(x)$ for a random x, sends at most $\ell \ll |y|$ bits y' to the computationally-unbounded *decompressor* E that, based on the digest y', should guess the hardcore bit $b(x)$. This notion naturally strengthens the standard notion of hardcore bits that corresponds to the case $\ell = 1$. Dubrov and Ishai [DI06] observed that standard transformations of OWP to PRGs can be used to transform OWP with ℓ-incompressible hardcore bits to $(\ell - 1)$-crypto-nb-PRG.

We show that a variant of their construction yields malleable PRGs when it is instantiated with *trapdoor permutations*. The following definition naturally generalizes the notion of one-way permutation with incompressible hardcore bits to the setting of trapdoor permutations.

Definition A.4. *A collection of permutations $\mathcal{F} = \{f_z\}$ is a* trapdoor-permutation with ℓ-incompressible hardcore bits *if the following properties hold:*

- *(Efficient indexing, evaluation and inversion) There exists a triple of probabilistic polynomial-time algorithms (I, F, F^{-1}) such that the followings hold. The* index-sampler $I(1^n)$ *uses $\rho(n)$ random bits to sample a description $z \in \{0,1\}^{L_1(n)}$ of some permutation $f_z : \{0,1\}^{m(n)} \to \{0,1\}^{m(n)}$ and a trapdoor information $t \in \{0,1\}^{L_2(n)}$. In addition, for every (z, t) in the support of $I(1^n)$ and every $x, y \in \{0,1\}^{m(n)}$ the* evaluator $F(z, x)$ *outputs $f_z(x)$, and the* inverter $F^{-1}(t, y)$ *outputs $f_z^{-1}(y)$.*
- *(Incompressibility) There exists an efficiently computable hardcore bit $b : \{0,1\}^{L_1(n)} \times \{0,1\}^{m(n)} \to \{0,1\}$ which is ℓ-*incompressible *in the following sense: For every $\mathrm{poly}(n)$-time compressor $D : \{0,1\}^{L_1(n)} \times \{0,1\}^{m(n)} \to \{0,1\}^{\ell(n)}$ and every computationally unbounded decompressor E it holds that*

$$\Pr_{(z,t)\leftarrow I(1^n), x\leftarrow\!\!\shortmid \{0,1\}^{m(n)}}[E(z, (D(z, f_z(x)))) = b(x)] \leqslant 1/2 + \varepsilon(n),$$

for some negligible function ε.

We move on and present a construction of malleable nb-PRG based on a (slightly modified) version of the BMY construction [BM82, Yao82]. Let us assume for now that the identifier z sampled by $I(1^n)$ is uniformly distributed in $\{0,1\}^{L_1(n)}$ and that $L_1(n) + L_2(n) < \ell(n)$. We will later explain how to relax these assumptions.

Construction A.5 (malleable nb-PRG from incompressible TDPs)**.** *Let $\tau : \mathbb{N} \to \mathbb{N}$ be a polynomially-bounded parameter.*

- *The generator G samples $(z,t) \leftarrow I(1^n)$ and $x^0 \leftarrow \{0,1\}^{m(n)}$, for $i = 0 \ldots \tau(n)-1$, it outputs $b(z,x^i)$ where x^i is defined recursively to be $f_z(x^{i-1})$. In addition it outputs $x^{\tau(n)}$ and z.*
- *The algorithm $H(\vec{b}, x^{\tau(n)}, z)$ outputs the suffix z.*
- *The (computationally-unbounded) algorithm $T(z)$ outputs some trapdoor t that corresponds to z.*
- *The algorithm A parses its first input as $(\vec{b}, x^{\tau(m)}, z)$ and uses its second input t to compute $x^0, \ldots, x^{\tau(m)-1}$ recursively by setting $x^i = F^{-1}(t, x^{i+1})$. The algorithm accepts if, for every i, the i-th entry of $\vec{b}$ equals to $b(z,x^i)$.*

By taking τ to be a sufficiently large polynomial (e.g., larger than the randomness complexity of I) we can guarantee that G expands its input. Also, it is not hard to verify that for super-constant $\tau = \omega(1)$ the malleability property, as defined in Eq. (2), holds. Indeed, a random string is accepted with probability $2^{-\tau}$ whereas a pseudorandom string is accepted with probability 1. Finally, the argument of [DI06, Thm. 3.2] shows that if the underlying TDP has ℓ-incompressible hardcore bits then the construction forms an $(\ell-1)$-*crypto-nb-PRG*. (For completeness we re-prove this in Lemma A.6.) Overall, assuming that the index z and trapdoor t have a total bit-length of $L_1 + L_2$ that is smaller than ℓ, we get a malleable cryptographic nb-PRG.

Lemma A.6. *Suppose that Construction A.5 is instantiated with a family $\mathcal{F}$ of trapdoor permutations with ℓ-incompressible hardcore bits and that the identifier z sampled by $I(1^n)$ is uniformly distributed in $\{0,1\}^{L_1(n)}$ and denote by $\rho(n)$ the randomness complexity of I. Then, for every polynomial $\tau(n) > \rho(n) - L_1(n)$, the resulting function $G : \{0,1\}^{\rho(n)+m(n)} \to \{0,1\}^{\tau(n)+m(n)+L_1(n)}$ is an $(\ell-1)$ crypto-nb-PRG.*

Proof. The output of G can be parsed into $(\vec{b}(z,x), x^\tau, z)$. Let $w = \tau + m + L_1$ denote the output length of G. (To avoid clutter we leave the dependency in n implicit.) Assume, towards a contradiction, that G is not an $(\ell-1)$ crypto-nb-PRG. That is, there exists some polynomial time algorithm $D : \{0,1\}^w \to \{0,1\}^{\ell-1}$, and a polynomial $p = p(n)$ such that $D(G(U_{\rho+m}))$ is $1/p$ far in statistical distance from $D(U_w)$. Using the standard equivalence between statistical distance and indistinguishability by computationally unbounded adversaries, we conclude that there exists an unbounded adversary $E : \{0,1\}^{\ell-1} \to \{0,1\}$ such that the adversary $A(\cdot) := E(D(\cdot))$ distinguishes $G(U_{\rho+m})$ from U_w with distinguishing advantage of $1/p$.

Using a standard hybrid argument (cf. [Gol01, Chapter 3.4]), we can turn A into a predictor algorithm A' that, for some $j \in \{1, \ldots, w\}$, predicts the j-th bit of the random variable $(\vec{b}(z,x), x^\tau, z)$ based on the $(w-j)$-long suffix of this random variable with inverse polynomial advantage of $1/(pw)$. Note that the last $m + L_1$ bits of the PRG (corresponding to the (x^τ, z) part) are uniformly distributed since $\mathcal{F}$ is a family of *permutations*. Therefore, the predictable index j must be smaller or equal to τ. Specifically, A' takes the following form. Given a suffix $(u_{j:\tau}, x^\tau, z)$, the predictor A' samples a random j-bit prefix $u' \leftarrow U_j$ and outputs $u'_j \oplus A'(u', u_{j:\tau}, x^\tau, z)$. We can now construct a compressor $\tilde{D}$ and a de-compressor $\tilde{E}$ that break the OWP family by essentially decomposing A into D and E again.

Formally, we define the operation of $\tilde{D}$ on input (z,y) (supposedly $y = f_z(x)$) as follows: Sample $u' \leftarrow U_j$ and sends the bit u'_j together with $d = D(u', b(z,y), b(z,y^1), ..., b(z,y^{\tau-j-1}(y)), y^\tau, z)$ where $y^i = f^i(y)$. The de-compressor $\tilde{E}$ outputs $E(d) \oplus u'$. It is not hard to verify that when z is random and $y = f_z(U_m)$, the distribution $(u', b(z,y), b(z,y^1), ..., b(z,y^{\tau-j-1}(y)), y^\tau, z)$ is identical to the hybrid distribution $(u', u_{j:\tau}, x^\tau, z)$ defined above. It follows that the success probability of $(\tilde{D}, \tilde{E})$ (over a random z and $y = f_z(U_m)$) is at least $1/2 + 1/(pw)$, as required. □

Relaxing the assumptions. The above argument shows that nb-distinguishers cannot tell the difference between a pseudorandom tuple $(\vec{b}, x^{\tau(n)}, z)$ that was sampled by G, and a random tuple in which $\vec{b}, x^{\tau(n)}$ are uniformly sampled and $z \leftarrow\!\!\!\!\rightarrow I(1^n)$. When the output of I is uniform this is equivalent to our definition of pseudorandom against nb-distinguishers. If z is not uniform (think of an RSA modulus), one can heuristically modify the construction as follows. First, assume that the randomness complexity $\rho(n)$ of I is relatively small compared to $\ell(n)$, say $\rho(n) \leqslant \ell(n)/3$. If this is not the case, reduce the randomness complexity of I via a standard cryptographic PRG – It seems likely that the incompressibility assumption still holds. Denote the random coins of I by r, and assume that z is (almost uniform) over some set Z and let L_1 denote $\log |Z|$. (For RSA this assumption holds assuming that the underlying cryptographic PRG used for sampling is regular.) Then, instead of outputting z, the generator G outputs $h(z)$ where h is a good randomness extractor that outputs essentially all the L_1-bits of entropy of z (h can be randomized as long as it outputs its seed). Now given $h(z)$ the mapping T recovers z, and outputs some random coins r for which $I(1^n, r) = (z, t)$. Finally, A computes $I(1^n; r) = (z, t)$ and proceeds as in the above construction. Observe that $L_1 + L_2 \leqslant 2\rho < \ell$, as required.

B Examples for F-SMSZK proof systems

In this section, we argue that the notion of SMSZK is a natural one by showing that the canonical HVSZK proof systems for the GNI problem [GMW91] and for the SZK problem [SV03, Vad99] achieve full F-SMSZK security where F is as large as the randomness complexity.

B.1 Graph Non-Isomorphism

Protocol B.1 ([GMW91] protocol for GNI)**.** Given shared input $\langle G_0 = (V_0, E_0), G_1 = (V_1, E_1)\rangle$ the renowned interactive proof $\langle P, V\rangle$ described below is an HVPZK proof:

1. The random tape of V consists of a random bit $b \leftarrow\!\!\!\!\rightarrow \{0,1\}$ and randomness r that defines a random permutation $\pi = \pi_r$ over $[n]$. For concreteness, we may assume that $r \in \{0,1\}^{O(n \log n)}$ and that the mapping from r to π is done by using the algorithm Π of [FY38].
2. V permutes the nodes of G_b under π, and sends the permuted graph $H = \pi(G_b)$ to P.
3. P finds $c \in \{0,1\}$ such that $H \cong G_c$ and sends c.
4. V accepts if $c = b$.

Proposition B.1. *The protocol B.1 is R-*SMPZK *for* GNI *with perfect completeness, and soundness error of $1/2$, where R is all the random bits used by V.*

Proof. Given any efficiently computable sampling algorithm $D(1^n, s)$ that samples an $R(n)$ bits, we define a new simulator Sim_D as follows:

- Sample a random tape s for D and generate $(b, r) = D(1^n, s)$.
- Set $\pi = \Pi(r)$ and $H = \pi(G_b)$.

- Output (s, H, b, b, π).

It is not hard to verify that the simulator Sim_D perfectly samples the view of the verifier, V_D that selects its random tape by invoking D. □

The protocol is typically iterated t-times in parallel to reduce the soundness error to 2^{-t}. This version of the protocol is also R-SMPZK via a similar proof.[11]

B.2 The Statistical Distance Protocols

The $\mathsf{SD}^{\alpha,\beta}$ is a promise problem whose input consists of a pair of circuits (X_1, X_2) of size n that map $k = k(n)$ bits to $m = m(n)$ bits. The input is a *Yes* instance (resp., *No* instance) if the output distributions $X_1(U_k)$ and $X_2(U_k)$ are $\alpha(n)$-far (resp., $\beta(n)$-close) in statistical distance.

Protocol B.2 ([SV03, Vad99] protocol for SD)**.** Given a pair of circuits $\langle X_1, X_2 \rangle$ the parties invoke the following protocol:

- The verifier's random tape consists of a random bit $b \hookleftarrow \{0,1\}$ and a random string $u \hookleftarrow U_k$.
- V sends $x = X_b(u)$ to P.
- If $\Pr[X_0 = x] > \Pr[X_1 = x]$ then P responds with $c = 0$, otherwise with $c = 1$.
- V accepts if $b = c$.

It is well known that the above protocol forms an HVSZK proof system for $\mathsf{SD}^{1-\varepsilon,\beta}$ with completeness and simulation error of ε and soundness error of $\frac{1+\beta}{2}$. We show that the protocol is in fact R-SMSZK protocol with similar parameters.

Proposition B.2. *For any $\varepsilon, \beta : \mathbb{N} \to [0,1]$, Protocol B.2 is $(R, 0)$-SMSZK proof system for $\mathsf{SD}^{1-\varepsilon,\beta}$ with completeness error of ε and soundness error of $\frac{1+\beta}{2}$, where $R(n) = k(n) + 1$ is the randomness complexity of Protocol B.2. Moreover, for $\varepsilon = 0$, the protocol forms an R-SMPZK proof system.*

Proof. Completeness and soundness are analyzed in [SV03, Vad99]. We prove that the protocol is $(R, 0)$-SMSZK. Let $\varepsilon, \beta : \mathbb{N} \to [0,1]$ and let $D : 1^n \times \{0,1\}^{S(n)} \mapsto \{0,1\}^{R(n)}$ be some efficient sampling algorithm. We define a simulator Sim_D for the proof system $\langle P, V^D \rangle$ that given a pair of circuits $\langle X_1, X_2 \rangle$ acts as follows:

- Sample $s \hookleftarrow U_{S(n)}$.
- Invoke $D(s)$ and parse it's output into (b, u)
- Generate $x = X_b(u)$
- Output $(s, b, x, c = b)$

[11] More generally, any R-SMPZK with strong simulation property as defined in Remark 5.5 remains R-SMPZK under parallel repetition.

The simulated view is distributed identically to the real view conditioned on the event that real view is being accepted (i.e., $c = b$). Hence for any $X = (X_0, X_1) \in \mathsf{SD}^{1-\varepsilon,\beta}$, it holds that

$$\mathsf{SD}(\mathsf{Sim}_D(X), \mathsf{view}_{V^D}(X)) \leqslant \mu(D) + 0$$

where $\mu(D)$ is the completeness error of $\langle P, V^D \rangle$, i.e., the maximum over all yes instances $\langle X_1, X_2 \rangle$, of

$$\Pr_{(b,u)\leftarrow\!\!\shortmid D(U_S), x = X_b(u)}[|X_b^{-1}(x)| \leqslant |X_{1-b}^{-1}(x)|]$$

where $X_a^{-1}(x)$ is the set of preimages of x under X_a. □

We mention that the classical result of [SV03] shows that the promise problem $\mathsf{SD}^{(1/3,2/3)}$ can be reduced in time $\mathrm{poly}(n, \tau)$ to an instance of $\mathsf{SD}^{(1-2^{-\tau},2^{-\tau})}$.