Pseudorandom Functions and Lattices

Abhishek Banerjee* Chris Peikert† Alon Rosen‡

September 29, 2011

Abstract

We give direct constructions of pseudorandom function (PRF) families based on conjectured hard lattice problems and learning problems. Our constructions are asymptotically efficient and highly parallelizable in a practical sense, i.e., they can be computed by simple, relatively *small* low-depth arithmetic or boolean circuits (e.g., in NC^1 or even TC^0). In addition, they are the first low-depth PRFs that have no known attack by efficient quantum algorithms. Central to our results is a new "derandomization" technique for the learning with errors (LWE) problem which, in effect, generates the error terms deterministically.

1 Introduction and Main Results

The past few years have seen significant progress in constructing public-key, identity-based, and homomorphic cryptographic schemes using lattices, e.g., [Reg05, PW08, GPV08, Gen09, CHKP10, ABB10a] and many more. Part of their appeal stems from provable worst-case hardness guarantees (starting with the seminal work of Ajtai [Ajt96]), good asymptotic efficiency and parallelism, and apparent resistance to quantum attacks (unlike the classical problems of factoring integers or computing discrete logarithms).

Perhaps surprisingly, there has been comparatively less progress in using lattices for *symmetric* cryptography, e.g., message authentication codes, block ciphers, and the like, which are widely used in practice. While in principle most symmetric objects of interest can be obtained generically from any one-way function, and hence from lattices, these generic constructions are usually very inefficient, which puts them at odds with the high performance demands of most applications. In addition, generic constructions often use their underlying primitives (e.g., one-way functions) in an inherently inefficient and *sequential* manner. While most lattice-based primitives are relatively efficient and highly parallelizable in a practical sense (i.e., they can be evaluated by small, low-depth circuits), those advantages are completely lost when plugging them into generic sequential constructions. This motivates the search for specialized constructions of symmetric objects that have comparable efficiency and parallelism to their lower-level counterparts.

Our focus in this work is on *pseudorandom function* (PRF) families, a central object in symmetric cryptography first rigorously defined and constructed by Goldreich, Goldwasser, and Micali ("GGM") [GGM84].

*School of Computer Science, Georgia Institute of Technology. Email: abhishek.banerjee@cc.gatech.edu. Research supported in part by an ARC Fellowship.

†School of Computer Science, Georgia Institute of Technology. Email: cpeikert@cc.gatech.edu. This material is based upon work supported by the National Science Foundation under Grant CNS-0716786 and CAREER Award CCF-1054495, and by the Alfred P. Sloan Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

‡Efi Arazi School of Computer Science, IDC Herzliya. Email: alon.rosen@idc.ac.il. Research supported in part by BSF grant 2010296.

Given a PRF family, most central goals of symmetric cryptography (e.g., encryption, authentication, identification) have simple solutions that make efficient use of the PRF. Informally, a family of deterministic functions is pseudorandom if no efficient adversary, given adaptive oracle access to a randomly chosen function from the family, can distinguish it from a uniformly random function. The seminal GGM construction is based generically on any length-doubling pseudorandom generator (and hence on any one-way function), but it requires k *sequential* invocations of the generator when operating on k-bit inputs.

In contrast, by relying on a generic object called a "pseudorandom *synthesizer*," or directly on concrete number-theoretic problems (such as decision Diffie-Hellman, RSA, and factoring), Naor and Reingold [NR95, NR97] and Naor, Reingold, and Rosen [NRR00] (see also [LW09, BMR10]) constructed very elegant and more efficient PRFs, which can in principle be computed in parallel by low-depth circuits (e.g., in NC^2 or TC^0). However, achieving such low depth for their number-theoretic constructions requires extensive preprocessing and enormous circuits, so their results serve mainly as a proof of theoretical feasibility rather than practical utility.

In summary, thus far all parallelizable PRFs from commonly accepted cryptographic assumptions rely on exponentiation in large multiplicative groups, and the functions (or at least their underlying hard problems) can be broken by polynomial-time quantum algorithms. While lattices appear to be a natural candidate for avoiding these drawbacks, and there has been some partial progress in the form of *randomized* weak PRFs [ACPS09] and randomized MACs [Pie10, KPC+11], constructing an efficient, parallelizable (deterministic) PRF under lattice assumptions has, frustratingly, remained open for some time now.

1.1 Results and Techniques

In this work we give the first direct constructions of PRF families based on lattices, via the *learning with errors* (LWE) [Reg05] and *ring*-LWE [LPR10] problems, and some new variants. Our constructions are highly parallelizable in a *practical* sense, i.e., they can be computed by relatively *small* low-depth circuits, and the runtimes are also potentially practical. (However, their performance and key sizes are still far from those of heuristically designed functions like AES.) In addition, (at least) one of our constructions can be evaluated in the circuit class TC^0 (i.e., constant-depth, poly-sized circuits with unbounded fan-in and threshold gates), which asymptotically matches the shallowest known PRF constructions based on the decision Diffie-Hellman and factoring problems [NR97, NRR00].

As a starting point, we recall that in their work introducing *synthesizers* as a foundation for PRFs [NR95], Naor and Reingold described a synthesizer based on a simple, conjectured hard-to-learn function. At first glance, this route seems very promising for obtaining PRFs from lattices, using LWE as the hard learning problem (which is known to be as hard as worst-case lattice problems [Reg05, Pei09]). However, a crucial point is that Naor and Reingold's synthesizer uses a *deterministic* hard-to-learn function, whereas LWE's hardness depends essentially on adding *random, independent* errors to every output of a mod-q "parity" function. (Indeed, without any error, parity functions are trivially easy to learn.) Probably the main obstacle so far in constructing efficient lattice/LWE-based PRFs has been in finding a way to introduce (sufficiently independent) error terms into each of the exponentially many function outputs, while still keeping the function deterministic and its key size a fixed polynomial. As evidence, consider that recent constructions of weaker primitives such as symmetric authentication protocols [HB01, JW05, KSS06], randomized weak PRFs [ACPS09], and message-authentication codes [Pie10, KPC+11] from noisy-learning problems are all inherently *randomized* functions, where security relies on introducing fresh noise at every invocation. Unfortunately, this is not an option for deterministic primitives like PRFs.

Derandomizing LWE**.** To resolve the above-described issues, our first main insight is a way of partially "derandomizing" the LWE problem, i.e., generating the *errors* efficiently and deterministically, while preserving hardness. This technique immediately yields a deterministic synthesizer and hence a simple and parallelizable PRF, though with a few subtleties specific to our technique that we elaborate upon below.

Before we explain the derandomization idea, first recall the learning with errors problem $\mathsf{LWE}_{n,q,\alpha}$ in dimension n (the main security parameter) with modulus q and error rate α. We are given many independent pairs $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$, where each $\mathbf{a}_i$ is uniformly random, and the b_i are all either "noisy inner products" of the form $b_i = \langle \mathbf{a}_i, \mathbf{s} \rangle + e_i \bmod q$ for a random secret $\mathbf{s} \in \mathbb{Z}_q^n$ and "small" random error terms $e_i \in \mathbb{Z}$ of magnitude $\approx \alpha q$, or are uniformly random and independent of the $\mathbf{a}_i$. The goal of the (decision) LWE problem is to distinguish between these two cases, with any non-negligible advantage. In the *ring*-LWE problem [LPR10], we are instead given noisy ring products $b_i \approx a_i \cdot s$, where s and the a_i are random elements of a certain polynomial ring R_q (the canonical example being $R_q = \mathbb{Z}_q[z]/(z^n + 1)$ for n a power of 2), and the error terms are "small" in a certain basis of the ring; the goal again is to distinguish these from uniformly random pairs. While the dimension n is the main hardness parameter, the error rate α also plays a very important role in both theory and practice: as long as the "absolute" error αq exceeds $\sqrt{n}$ or so, (ring-)LWE is provably as hard as approximating conjectured hard problems on (ideal) lattices to within $\tilde{O}(n/\alpha)$ factors in the worst case [Reg05, Pei09, LPR10]. Moreover, known attacks using lattice basis reduction (e.g., [LLL82, Sch87]) or combinatorial/algebraic methods [BKW03, AG11] require time $2^{\tilde{\Omega}(n/\log(1/\alpha))}$, where the $\tilde{\Omega}(\cdot)$ notation hides polylogarithmic factors in n. We emphasize that without the error terms, (ring-)LWE would become trivially easy, and that all prior hardness results for LWE and its many variants (e.g., [Reg05, Pei09, GKPV10, LPR10, Pie10]) require random, independent errors.

Our derandomization technique for LWE is very simple: instead of adding a small random error term to each inner product $\langle \mathbf{a}_i, \mathbf{s} \rangle \in \mathbb{Z}_q$, we just deterministically *round* it to the nearest element of a sufficiently "coarse" public subset of $p \ll q$ well-separated values in $\mathbb{Z}_q$ (e.g., a subgroup). In other words, the "error term" comes solely from deterministically rounding $\langle \mathbf{a}_i, \mathbf{s} \rangle$ to a relatively nearby value. Since there are only p possible rounded outputs in $\mathbb{Z}_q$, it is usually easier to view them as elements of $\mathbb{Z}_p$ and denote the rounded value by $\lfloor \langle \mathbf{a}_i, \mathbf{s} \rangle \rceil_p \in \mathbb{Z}_p$. We call the problem of distinguishing such rounded inner products from uniform samples the *learning with rounding* ($\mathsf{LWR}_{n,q,p}$) problem. Note that the problem can be hard only if $q > p$ (otherwise no error is introduced), that the "absolute" error is roughly q/p, and that the "error rate" relative to q (i.e., the analogue of α in the LWE problem) is on the order of $1/p$.

We show that for appropriate parameters, $\mathsf{LWR}_{n,q,p}$ is at least as hard as $\mathsf{LWE}_{n,q,\alpha}$ for an error rate α proportional to $1/p$, giving us a worst-case hardness guarantee for LWR. In essence, the reduction relies on the fact that with high probability, we have $\lfloor \langle \mathbf{a}, \mathbf{s} \rangle + e \rceil_p = \lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rceil_p$ when e is small relative to q/p, while $\lfloor U(\mathbb{Z}_q) \rceil_p \approx U(\mathbb{Z}_p)$ where U denotes the uniform distribution. Therefore, given samples $(\mathbf{a}_i, b_i)$ of an unknown type (either LWE or uniform), we can simply round the b_i terms to generate samples of a corresponding type (LWR or uniform, respectively). (The formal proof is somewhat more involved, because it has to deal with the rare event that the error term changes the rounded value.) In the ring setting, the derandomization technique and hardness proof based on ring-LWE all go through without difficulty as well. While our proof needs both the ratio q/p and the inverse LWE error rate $1/\alpha$ to be slightly super-polynomial in n, the state of the art in attack algorithms indicates that as long as q/p is an integer (so that $\lfloor U(\mathbb{Z}_q) \rceil_p = U(\mathbb{Z}_p)$) and is at least $\Omega(\sqrt{n})$, LWR may be exponentially hard (even for quantum algorithms) for any $p = \mathrm{poly}(n)$, and superpolynomially hard when $p = 2^{n^\epsilon}$ for any $\epsilon < 1$.

We point out that in LWE-based cryptosystems, rounding to a fixed, coarse subset is a common method of removing noise and recovering the plaintext when decrypting a "noisy" ciphertext; here we instead use it to avoid having to introduce any random noise in the first place. We believe that this technique should be useful

in many other settings, especially in symmetric cryptography. For example, the LWR problem immediately yields a simple and practical pseudorandom generator that does not require extracting biased (e.g., Gaussian) random values from its input seed, unlike the standard pseudorandom generators based on the LWE or LPN (learning parity with noise) problems. In addition, the rounding technique and its implications for PRFs are closely related to the "modulus reduction" technique from a concurrent and independent work of Brakerski and Vaikuntanathan [BV11a] on fully homomorphic encryption from LWE, and a very recent follow-up work of Brakerski, Gentry, and Vaikuntanathan [BGV11]; see Section 1.3 below for a discussion and comparison.

LWR-based synthesizers and PRFs. Recall from [NR95] that a pseudorandom *synthesizer* is a two-argument function $S(\cdot,\cdot)$ such that, for random and independent sequences $x_1,\ldots,x_m$ and $y_1,\ldots,y_m$ of inputs (for any $m = \mathrm{poly}(n)$), the matrix of all m^2 values $z_{i,j} = S(x_i, y_j)$ is pseudorandom (i.e., computationally indistinguishable from uniform). A synthesizer can be seen as an (almost) length-*squaring* pseudorandom generator with good locality properties, in that it maps $2m$ random "seed" elements (the x_i and y_j) to m^2 pseudorandom elements, and any component of its output depends on only two components of the input seed.

Using synthesizers in a recursive tree-like construction, Naor and Reingold gave PRFs on k-bit inputs, which can be computed using a total of about k synthesizer evaluations, arranged nicely in only $\lg k$ levels (depth). Essentially, the main idea is that given a synthesizer $S(\cdot,\cdot)$ and two independent PRF instances F_0 and F_1 on t input bits each, one gets a PRF on $2t$ input bits, defined as

$$F(x_1 \cdots x_{2t}) = S\big(\, F_0(x_1 \cdots x_t)\,,\, F_1(x_{t+1} \cdots x_{2t})\,\big). \tag{1.1}$$

The base case of a 1-bit PRF can trivially be implemented by returning one of two random strings in the function's secret key. Using particular NC^1 synthesizers based on a variety of both concrete and general assumptions, Naor and Reingold therefore obtain k-bit PRFs in NC^2, i.e., having circuit depth $O(\log^2 k)$.

We give a very simple and computationally efficient $\mathsf{LWR}_{n,q,p}$-based synthesizer $S_{n,q,p}\colon \mathbb{Z}_q^n \times \mathbb{Z}_q^n \to \mathbb{Z}_p$, defined as

$$S_{n,q,p}(\mathbf{a}, \mathbf{s}) = \lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rceil_p. \tag{1.2}$$

(In this and what follows, products of vectors or matrices over $\mathbb{Z}_q$ are always performed modulo q.) Pseudorandomness of this synthesizer under LWR follows by a standard hybrid argument, using the fact that the $\mathbf{a}_i$ vectors given in the LWR problem are public. (In fact, the synthesizer outputs $S(\mathbf{a}_i, \mathbf{s}_j)$ are pseudorandom even given the $\mathbf{a}_i$.) To obtain a PRF using the tree construction of [NR95], we need the synthesizer output length to roughly match its input length, so we actually use the synthesizer $T_{n,q,p}(\mathbf{S}_1, \mathbf{S}_2) = \lfloor \mathbf{S}_1 \cdot \mathbf{S}_2 \rceil_p \in \mathbb{Z}_p^{n \times n}$ for $\mathbf{S}_i \in \mathbb{Z}_q^{n \times n}$. Note that the matrix multiplication can be done with a constant-depth, size-$O(n^2)$ arithmetic circuit over $\mathbb{Z}_q$. Or for better space and time complexity, we can instead use the ring-LWR synthesizer $S_{R,q,p}(s_1, s_2) = \lfloor s_1 \cdot s_2 \rceil_p$, since the ring product $s_1 \cdot s_2 \in R_q$ is the same size as $s_1, s_2 \in R_q$. The ring product can also be computed with a constant depth, size-$O(n^2)$ circuit over $\mathbb{Z}_q$, or in $O(\log n)$ depth and only $O(n \log n)$ scalar operations using Fast Fourier Transform-like techniques [LMPR08, LPR10].

Using the recursive input-doubling construction from Equation (1.1) above, we get the following concrete PRF with input length $k = 2^d$. Let $q_d > q_{d-1} > \cdots > q_0 \geq 2$ be a chain of moduli where each q_j/q_{j-1} is a sufficiently large integer, e.g., $q_j = q^{j+1}$ for some $q \geq \sqrt{n}$. The secret key is a set of $2k$ matrices $\mathbf{S}_{i,b} \in \mathbb{Z}_{q_d}^{n \times n}$ for each $i \in \{1,\ldots,k\}$ and $b \in \{0,1\}$. Each pair $(\mathbf{S}_{i,0}, \mathbf{S}_{i,1})$ defines a 1-bit PRF $F_i(b) = \mathbf{S}_{i,b}$, and these are combined in a tree-like fashion according to Equation (1.1) using the appropriate synthesizers

$T_{n,q_j,q_{j-1}}$ for $j = d, \ldots, 1$. As a concrete example, when $k = 8$ (so $x = x_1 \cdots x_8$ and $d = 3$), we have

$$F_{\{\mathbf{S}_{i,b}\}}(x) \;=\; \left\lfloor \left\lfloor \lfloor \mathbf{S}_{1,x_1} \cdot \mathbf{S}_{2,x_2} \rceil_{q_2} \cdot \lfloor \mathbf{S}_{3,x_3} \cdot \mathbf{S}_{4,x_4} \rceil_{q_2} \right\rceil_{q_1} \cdot \left\lfloor \lfloor \mathbf{S}_{5,x_5} \cdot \mathbf{S}_{6,x_6} \rceil_{q_2} \cdot \lfloor \mathbf{S}_{7,x_7} \cdot \mathbf{S}_{8,x_8} \rceil_{q_2} \right\rceil_{q_1} \right\rfloor_{q_0} . \quad (1.3)$$

(In the ring setting, we just use random elements $s_{i,b} \in R_{q_d}$ in place of the matrices $\mathbf{S}_{i,b}$.) Notice that the function involves $d = \lg k$ levels of matrix (or ring) products, each followed by a rounding operation. In the exemplary case where $q_j = q^{j+1}$, the rounding operations essentially drop the "least-significant" base-q digit, so they can be implemented very easily in practice, especially if every q_j is a power of 2. The function is also amenable to all of the nice time/space trade-offs, seed-compression techniques, and incremental computation ideas described in [NR95].

In the security proof, we rely on the conjectured hardness of $\mathsf{LWR}_{q_j,q_{j-1}}$ for $j = d, \ldots, 1$. The strongest of these assumptions appears to be for $j = d$, and this is certainly the case when relying on our reduction from LWE to LWR. For the example parameters $q_j = q^{j+1}$ where $q \approx \sqrt{n}$, the dominating assumption is therefore the hardness of $\mathsf{LWR}_{q^{d+1},q^d}$, which involves a quasi-polynomial inverse error rate of $1/\alpha \approx q^d = n^{O(\lg k)}$. However, because the strongest assumptions are applied to the "innermost" layers of the function, it is unclear whether security actually *requires* such strong assumptions, or even whether the innermost layers need to be rounded at all. We discuss these issues further in Section 1.2 below.

Degree-k synthesizers and shallower PRFs. One moderate drawback of the above function is that it involves $\lg k$ levels of rounding operations, which appears to lower-bound the depth of any circuit computing the function by $\Omega(\lg k)$. Is it possible to do better?

Recall that in later works, Naor and Reingold [NR97] and Naor, Reingold, and Rosen [NRR00] gave direct, more efficient number-theoretic PRF constructions which, while still requiring exponentiation in large multiplicative groups, can in principle be computed in very shallow circuit classes like NC^1 or even TC^0. Their functions can be interpreted as "degree-k" (or k-argument) synthesizers for arbitrary $k = \mathrm{poly}(n)$, which immediately yield k-bit PRFs without requiring any composition. With this in mind, a natural question is whether there are direct LWE/LWR-based synthesizers of degree $k > 2$.

We give a positive answer to this question. Much like the functions of [NR97, NRR00], ours have a subset-product structure. We have public moduli $q \gg p$, and the secret key is a set of k matrices $\mathbf{S}_i \in \mathbb{Z}_q^{n \times n}$ (whose distributions may not necessarily be uniform; see below) for $i = 1, \ldots, k$, along with a uniformly random $\mathbf{a} \in \mathbb{Z}_q^n$.[1] The function $F = F_{\mathbf{a},\{\mathbf{S}_i\}} \colon \{0,1\}^k \to \mathbb{Z}_p^n$ is defined as the "rounded subset-product"

$$F_{\mathbf{a},\{\mathbf{S}_i\}}(x_1 \cdots x_k) = \left\lfloor \mathbf{a}^t \cdot \prod_{i=1}^{k} \mathbf{S}_i^{x_i} \right\rceil_p . \quad (1.4)$$

The ring variant is analogous, replacing $\mathbf{a}$ with uniform $a \in R_q$ (or R_q^*, the set of invertible elements) and each $\mathbf{S}_i$ by some $s_i \in R_q$. This function is particularly efficient to evaluate using the discrete Fourier transform, as is standard with ring-based primitives (see,e.g., [LMPR08, LPR10]). In addition, similarly to [NR97, NRR00], one can optimize the subset-product operation via pre-processing, and evaluate the function in TC^0. We elaborate on these optimizations in Section 5.2.

For the security analysis of construction (1.4), we have meaningful security proofs under various conditions on the parameters and computational assumptions, including standard LWE. In our LWE-based proof, two important issues are the distribution of the secret key components $\mathbf{S}_i$, and the choice of moduli q and p.

[1] To obtain longer function outputs, we can replace $\mathbf{a} \in \mathbb{Z}_q^n$ with a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ for any $m = \mathrm{poly}(n)$.

For the former, it turns out that our proof needs the $\mathbf{S}_i$ matrices to be *short*, i.e., their entries should be drawn from the LWE error distribution. (LWE is no easier to solve for such short secrets [ACPS09].) This appears to be an artifact of our proof technique, which can be viewed as a variant of our LWE-to-LWR reduction, enhanced to handle adversarial queries. Summarizing the approach, define

$$G(x) = G_{\mathbf{a},\{\mathbf{S}_i\}}(x) := \mathbf{a}^t \cdot \prod_i \mathbf{S}_i^{x_i}$$

to be the subset-product function inside the rounding operation of (1.4). The fact that $F = \lfloor G \rceil_p$ lets us imagine adding *independent error terms* to each distinct output of G, but *only as part of a thought experiment* in the proof. More specifically, we consider a related *randomized* function $\tilde{G} = \tilde{G}_{\mathbf{a},\{\mathbf{S}_i\}} \colon \{0,1\}^k \to \mathbb{Z}_q^n$ that computes the subset-product by multiplying by each $\mathbf{S}_i^{x_i}$ in turn, but then also adds a fresh error term immediately following each multiplication. Using the LWE assumption and induction on k, we can show that the randomized function $\tilde{G}$ is itself pseudorandom (over $\mathbb{Z}_q$), hence so is $\lfloor \tilde{G} \rceil_p$ (over $\mathbb{Z}_p$). Moreover, we show that for every queried input, with high probability $\lfloor \tilde{G} \rceil_p$ coincides with $\lfloor G \rceil_p = F$, because G and $\tilde{G}$ differ only by a cumulative error term that is small relative to q—this is where we need to assume that the entries of $\mathbf{S}_i$ are *small*. Finally, because $\lfloor \tilde{G} \rceil_p$ is a (randomized) pseudorandom function over $\mathbb{Z}_p$ that coincides with the deterministic function F on all queries, we can conclude that F is pseudorandom as well.

In the above-described proof strategy, the gap between G and $\tilde{G}$ grows *exponentially* in k, because we add a separate noise term following each multiplication by an $\mathbf{S}_i$, which gets enlarged when multiplied by all the later $\mathbf{S}_i$. So in order to ensure that $\lfloor \tilde{G} \rceil_p = \lfloor G \rceil_p$ on all queries, our LWE-based proof needs both the modulus q and inverse error rate $1/\alpha$ to exceed $n^{\Omega(k)}$. In terms of efficiency and security, this compares rather unfavorably with the quasipolynomial $n^{O(\lg k)}$ bound in the proof for our tree-based construction, though on the positive side, the direct degree-k construction has better circuit depth. However, just as with construction (1.3) it is unclear whether such strong assumptions and large parameters are actually *necessary* for security, or whether the matrices $\mathbf{S}_i$ really need to be short.

In particular, it would be nice if the function in (1.4) were secure if the $\mathbf{S}_i$ matrices were *uniformly random* over $\mathbb{Z}_q^{n \times n}$, because we could then recursively compose the function in a k-ary tree to rapidly extend its input length.[2] It would be even better to have a security proof for a smaller modulus q and inverse error rate $1/\alpha$, ideally both polynomial in n even for large k. While we have been unable to find such a security proof under standard LWE, we do give a very tight proof under a new, interactive "*related samples*" LWE/LWR assumption. Roughly speaking, the assumption says that LWE/LWR remains hard even when the sampled $\mathbf{a}_i$ vectors are related by adversarially chosen subset-products of up to k given random matrices (drawn from some known distribution). This provides some evidence that the function may indeed be secure for appropriately distributed $\mathbf{S}_i$, small modulus q, and large k. For further discussion, see Section 1.2.

PRFs via the GGM construction. The above constructions aim to minimize the depth of the circuit evaluating the PRF. However, if parallel complexity is not a concern, and one wishes to minimize the total amount of work per PRF evaluation (or the seed length), then the original GGM construction with an LWR-based pseudorandom generator may turn out to be even more efficient in practice.

Recall that the GGM construction makes generic use of any length-doubling pseudorandom generator $G : \{0,1\}^n \to \{0,1\}^{2n}$. The generator's output $G(s)$ is viewed as a pair $(G_0(s), G_1(s))$, where $|G_0(s)| = |G_1(s)| = n$. The key for a member of the PRF family is a seed s for G, and on input $x \in \{0,1\}^k$ the

[2] Note that we can always compose the degree-k function with our degree-2 synthesizers from above, but this would only yield a tree with 2-ary internal nodes.

function is defined as

$$F_s(x_1 \cdots x_k) = G_{x_k}(G_{x_{k-1}}(\cdots G_{x_1}(s) \cdots)). \tag{1.5}$$

As mentioned above, the LWR problem immediately yields a simple and practical pseudorandom generator that, in contrast to the generators obtained from the LWE or LPN problems, does not require extracting biased random error terms from its input seed. By plugging this generator into the GGM construction we immediately get a PRF whose evaluation involves precisely k sequential evaluations of the underlying generator.

The LWR-based generator that we have in mind is a function $G_{\mathbf{A}} : \mathbb{Z}_q^n \to \mathbb{Z}_p^m$, where the moduli $q \gg p$ and the (uniformly random) matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ are publicly known. Given a seed $\mathbf{s} \in \mathbb{Z}_q^n$, the generator is defined as

$$G_{\mathbf{A}}(\mathbf{s}) = \left\lfloor \mathbf{A}^t \cdot \mathbf{s} \right\rceil_p. \tag{1.6}$$

The generator's seed length (in bits) is $n \log_2 q$ and its output length is $m \log_2 p$, which gives an expansion rate of $(m \log_2 p)/(n \log_2 q) = (m/n) \log_q p$. For example, to obtain a length-doubling generator we may set $q = p^2 = 2^{2k} > n$ and $m = 4n$. (Other choices yielding different expansion rates are of course possible.) This choice of parameters has the additional benefit of admitting a practical implementation of the rounding and inner-product operations. Note also that when evaluating the resulting PRF, one can get the required part of $G_{\mathbf{A}}(\mathbf{s})$ by computing only the inner products of $\mathbf{s}$ with the corresponding columns of $\mathbf{A}$, not the entire product $\mathbf{A}^t \cdot \mathbf{s}$.

For an even faster implementation one may replace $G_{\mathbf{A}}$ by its analogous ring variant, obtained by replacing $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ with uniform $\mathbf{a} \in R_q^m$, and $\mathbf{s} \in \mathbb{Z}_q^n$ with uniform $s \in R_q$. As noted before, the ring variant is particularly efficient to evaluate using Fast Fourier Transform-like algorithms.

1.2 Discussion and Open Questions

The quasipolynomial $n^{O(\log k)}$ or exponential $n^{O(k)}$ moduli and inverse error rates used in our LWE-based security proofs are comparable to those used in recent fully homomorphic encryption (FHE) schemes (e.g., [Gen09, vGHV10, BV11b, BV11a, BGV11]), hierarchical identity-based encryption (HIBE) schemes (e.g., [CHKP10, ABB10a, ABB10b]), and other lattice-based constructions. However, there appears to be a major difference between our use of such strong assumptions, and that of schemes such as FHE/HIBE in the public-key setting. Constructions of the latter systems actually reveal LWE samples having very small error rates (which are needed to ensure correctness of decryption) to the attacker, and the attacker can break the cryptosystems by solving those instances. Therefore, the underlying assumptions and the true security of the schemes are essentially equivalent. In contrast, our PRF uses (small) errors *only as part of a thought experiment* in the security proof, not for any purpose in the operation of the function itself. This leaves open the possibility that our functions (or slight variants) remain secure even for much larger input lengths and smaller moduli than our proofs require. We conjecture that this is the case, even though we have not yet found security proofs (under standard assumptions) for these more efficient parameters. Certainly, determining whether there are effective cryptanalytic attacks is a very interesting and important research direction.

Note that in our construction (1.4), if we draw the secret key components from the uniform (or error) distribution and allow k to be too large relative to q, then the function can become insecure via a simple attack (and our new "interactive" LWR assumption, which yields a tight security proof, becomes false). This is easiest to see for the ring-based function: representing each $s_i \in R_q$ by its vector of "Fourier coefficients" over $\mathbb{Z}_q^n$, each coefficient is 0 with probability about $1/q$ (depending on the precise distribution of s_i). Therefore, with noticeable probability the product of $k = O(q \log n)$ random s_i will have all-0 Fourier coefficients, i.e., will be $0 \in R_q$. In this case our function will return zero on the all-1s input, in violation

of the PRF requirement. (A similar but more complicated analysis can also be applied to the matrix-based function.) Of course, an obvious countermeasure is just to restrict the secret key components to be *invertible*; to our knowledge, this does not appear to have any drawback in terms of security. In fact, it is possible to show that the decision-(ring-)LWE problem remains hard when the secret is restricted to be invertible (and otherwise drawn from the uniform or error distribution), and this fact may be useful in further analysis of the function with more efficient parameters.

In summary, our work raises several interesting concrete questions, including:

- Is $\mathsf{LWR}_{n,q,p}$ really exponentially hard for $p = \mathrm{poly}(n)$ and sufficiently large integer $q/p = \mathrm{poly}(n)$? Are there stronger worst-case hardness guarantees than our current proof based on LWE?
- Is there a security proof for construction (1.4) (with $k = \omega(1)$) for $\mathrm{poly}(n)$-bounded moduli and inverse error rates, under a non-interactive assumption?
- In construction (1.4), is there a security proof (under a non-interactive assumption) for uniformly random $\mathbf{S}_i$? Is there any provable security advantage to using *invertible* $\mathbf{S}_i$?
- Our derandomization technique and LWR problem require working with moduli q greater than 2. Is there an efficient, parallel PRF construction based on the learning parity with noise (LPN) problem?

1.3 Other Related Work

Most closely related to the techniques in this work are two very recent results of Brakerski and Vaikuntanathan [BV11a] and a follow-up work of Brakerski, Gentry, and Vaikuntanathan [BGV11] on fully homomorphic encryption from LWE. In particular, the former work includes a "modulus reduction" technique for LWE-based cryptosystems, which maps a large-modulus ciphertext to a small-modulus one; this induces a shallower decryption circuit and allows the system to be "bootstrapped" into a fully homomorphic scheme using the techniques of [Gen09]. The modulus-reduction technique involves a rounding operation much like the one we use to derandomize LWE; while they use it on ciphertexts that are already "noisy," we apply it to noise-free LWE samples. Our discovery of the rounding/derandomization technique in the PRF context was independent of [BV11a]. In fact, the first PRF and security proof we found were for the direct degree-k construction defined in (1.4), not the synthesizer-based construction in (1.3). As another point of comparison, the "somewhat homomorphic" cryptosystem from [BV11a] that supports degree-k operations (along with all prior ones, e.g., [Gen09, vGHV10]) involves an inverse error rate of $n^{O(k)}$, much like the LWE-based proof for our degree-k synthesizer.

Building on the modulus reduction technique of [BV11a], Brakerski *et al.* [BGV11] showed that homomorphic cryptosystems can support certain degree-k functions using a much smaller modulus and inverse error rate of $n^{O(\log k)}$. The essential idea is to interleave the homomorphic operations with several "small" modulus-reduction steps in a tree-like fashion, rather than performing all the homomorphic operations followed by one "huge" modulus reduction. This very closely parallels the difference between our direct degree-k synthesizer and the Naor-Reingold-like [NR95] composed synthesizer defined in (1.3). Indeed, after we found construction (1.4), the result of [BGV11] inspired our search for a PRF having similar tree-like structure and quasipolynomial error rates. Given our degree-2 synthesizer, the solution turned out to largely be laid out in the work of [NR95]. We find it very interesting that the same quantitative phenomena arise in two seemingly disparate settings (PRFs and FHE).

1.4 Organization

The rest of the paper is organized as follows. In Section 2 we recall the necessary preliminaries regarding PRFs and the (ring-)LWE problem. In Section 3 we introduce the "learning with rounding" (LWR) problem and discuss its relationship with LWE. In Section 4 we describe LWR-based (degree-2) synthesizers and the PRFs that follow from them. In Section 5 we describe our direct degree-k synthesizer/PRF and its security proofs under the LWE and "subset-product" LWR problem.

2 Preliminaries

For a probability distribution X over a domain D, let X^n denote its n-fold product distribution over D^n. The uniform distribution over a finite domain D is denoted by $U(D)$. The discrete Gaussian probability distribution over $\mathbb{Z}$ with parameter $r > 0$, denoted $D_{\mathbb{Z},r}$, assigns probability proportional to $\exp(-\pi x^2/r^2)$ to each $x \in \mathbb{Z}$. It is possible to efficiently sample from this distribution (up to $\mathrm{negl}(n)$ statistical distance) via rejection [GPV08].

For any integer modulus $q \geq 2$, $\mathbb{Z}_q$ denotes the quotient ring of integers modulo q. We define a 'rounding' function $\lfloor \cdot \rceil_p \colon \mathbb{Z}_q \to \mathbb{Z}_p$, where $q \geq p \geq 2$ will be apparent from the context, as

$$\lfloor x \rceil_p = \lfloor (p/q) \cdot \bar{x} \rceil \bmod p, \tag{2.1}$$

where $\bar{x} \in \mathbb{Z}$ is any integer congruent to $x \bmod q$. We extend $\lfloor \cdot \rceil_p$ component-wise to vectors and matrices over $\mathbb{Z}_q$, and coefficient-wise (with respect to the "power basis") to the quotient ring R_q defined in the next subsection. Note that we can use any other common rounding method, like the floor $\lfloor \cdot \rfloor$, or ceiling $\lceil \cdot \rceil$ functions, in Equation 2.1 above, with only minor changes to our proofs. In implementations, it may be advantageous to use the floor function $\lfloor \cdot \rfloor$ when q and p are both powers of some common base b (e.g., 2). In this setting, computing $\lfloor \cdot \rfloor_p$ is equivalent to dropping the least-significant digit(s) in base b.

2.1 Pseudorandom Functions

The main security parameter through this paper is n, and all algorithms (including the adversary) are implicitly given the security parameter n in unary. We write $\mathrm{negl}(n)$ to denote an arbitrary *negligible* function in n, one that vanishes faster than the inverse of any polynomial. We say that a probability is *overwhelming* if it is $1 - \mathrm{negl}(n)$.

We consider adversaries interacting as part of probabilistic experiments called *games*. For an adversary $\mathcal{A}$ and two games H_0, H_1 with which it can interact, $\mathcal{A}$'s *distinguishing advantage* (implicitly, as a function of n) is $\mathbf{Adv}_{H_0,H_1}(\mathcal{A}) := |\Pr[\mathcal{A} \text{ accepts in } H_0] - \Pr[\mathcal{A} \text{ accepts in } H_1]|$.

Definition 2.1 (Computational Indistinguishability)**.** We say that games H_0, H_1 are *computationally indistinguishable*, written $H_0 \overset{c}{\approx} H_1$, if $\mathbf{Adv}_{H_0,H_1}(\mathcal{A}) = \mathrm{negl}(n)$ for any probabilistic polynomial-time $\mathcal{A}$.

By the triangle inequality, $\overset{c}{\approx}$ is a transitive relation over any $\mathrm{poly}(n)$-length sequence of games. If $H_0 \overset{c}{\approx} H_1$ and $\mathcal{S}$ is any probabilistic polynomial-time algorithm, then the outputs of $\mathcal{S}$ playing in games H_0 and H_1 (respectively) are also computationally indistinguishable.

Definition 2.2 (Pseudorandom functions)**.** Let A and B be finite sets, and let $\mathcal{F} = \{F_i \colon A \to B\}$ be a function family, endowed with an efficiently sampleable distribution (more precisely, $\mathcal{F}$, A and B are all indexed by the security parameter n). We say that $\mathcal{F}$ is a *pseudorandom function* (PRF) family if the following two games are computationally indistinguishable:

1. Choose a function $F \leftarrow \mathcal{F}$ and give the adversary adaptive oracle access to $F(\cdot)$.

2. Choose a uniformly random function $U\colon A \to B$ and give the adversary adaptive oracle access to $U(\cdot)$.

To efficiently simulate access to a uniformly random function $U\colon A \to B$, one may think of a process in which the adversary's queries are "lazily" answered with independently and randomly chosen elements in B, while keeping track of the answers so that queries made more than once are answered consistently.

2.2 (Ring) Learning With Errors

We recall the learning with errors (LWE) problem due to Regev [Reg05] and its ring analogue by Lyubashevsky, Peikert, and Regev [LPR10]. For positive integer dimension n (the security parameter) and modulus $q \geq 2$, a probability distribution χ over $\mathbb{Z}$, and a vector $\mathbf{s} \in \mathbb{Z}_q^n$, define the LWE distribution $A_{\mathbf{s},\chi}$ to be the distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ obtained by choosing a vector $\mathbf{a} \leftarrow \mathbb{Z}_q^n$ uniformly at random, an error term $e \leftarrow \chi$, and outputting $(\mathbf{a}, b = \langle \mathbf{a}, \mathbf{s} \rangle + e \bmod q)$. We use the following "normal form" of the *decision*-$\mathsf{LWE}_{n,q,\chi}$ problem, which is to distinguish (with advantage non-negligible in n) between any desired number $m = \mathrm{poly}(n)$ of independent samples $(\mathbf{a}_i, b_i) \leftarrow A_{\mathbf{s},\chi}$ where $\mathbf{s} \leftarrow \chi^n \bmod q$ is chosen from the (folded) error distribution, and the same number of samples from the uniform distribution $U(\mathbb{Z}_q^n \times \mathbb{Z}_q)$. This form of the problem is as hard as the one where $\mathbf{s} \in \mathbb{Z}_q^n$ is chosen uniformly at random [ACPS09].

We extend the LWE distribution to $w \geq 1$ secrets, defining $A_{\mathbf{S},\chi}$ for $\mathbf{S} \in \mathbb{Z}_q^{n \times w}$ to be the distribution obtained by choosing $\mathbf{a} \leftarrow \mathbb{Z}_q^n$, an error vector $\mathbf{e}^t \leftarrow \chi^w$, and outputting $(\mathbf{a}, \mathbf{b}^t = \mathbf{a}^t\mathbf{S} + \mathbf{e}^t \bmod q)$. By a standard hybrid argument, distinguishing such samples (for $\mathbf{S} \leftarrow \chi^{n \times w}$) from uniformly random is as hard as decision-$\mathsf{LWE}_{n,q,\chi}$, for any $w = \mathrm{poly}(n)$. It is often convenient to group many (say, m) sample pairs together in matrices. This allows us to express the LWE problem as: distinguish any desired number of pairs $(\mathbf{A}^t, \mathbf{B}^t = \mathbf{A}^t\mathbf{S} + \mathbf{E} \bmod q) \in \mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^{m \times w}$, for the same $\mathbf{S}$, from uniformly random.

For certain moduli q and (discrete) Gaussian error distributions χ, the decision-LWE problem is as hard as the search problem, where the goal is to find $\mathbf{s}$ given samples from $A_{\mathbf{s},\chi}$ (see, e.g., [Reg05, Pei09, ACPS09, MM11], and [MP11] for the mildest known requirements on q, which include the case where q is a power of 2). In turn, for $\chi = D_{\mathbb{Z},r}$ with $r = \alpha q \geq 2\sqrt{n}$, the search problem is as hard as approximating worst-case lattice problems to within $\tilde{O}(n/\alpha)$ factors; see [Reg05, Pei09] for precise statements.[3]

Ring-LWE. For simplicity of exposition, we use the following special case of the ring-LWE problem. (Our results can be extended to the more general form defined in [LPR10].) Throughout the paper we let R denote the cyclotomic polynomial ring $R = \mathbb{Z}[z]/(z^n + 1)$ for n a power of 2. (Equivalently, R is the ring of integers $\mathbb{Z}[\omega]$ for $\omega = \exp(\pi i/n)$.) For any integer modulus q, define the quotient ring $R_q = R/qR$. An element of R can be represented as a polynomial (in z) of degree less than n having integer coefficients; in other words, the "power basis" $\{1, z, \ldots, z^{n-1}\}$ is a $\mathbb{Z}$-basis for R. Similarly, it is a $\mathbb{Z}_q$-basis for R_q.

For a modulus q, a probability distribution χ over R, and an element $s \in R_q$, the ring-LWE (RLWE) distribution $A_{s,\chi}$ is the distribution over $R_q \times R_q$ obtained by choosing $a \in R_q$ uniformly at random, an error term $x \leftarrow \chi$, and outputting $(a, b = a \cdot s + x \bmod qR)$. The normal form of the decision-$\mathsf{RLWE}_{R,q,\chi}$ problem is to distinguish (with non-negligible advantage) between any desired number $m = \mathrm{poly}(n)$ of independent samples $(a_i, b_i) \leftarrow A_{s,\chi}$ where $s \leftarrow \chi \bmod q$, and the same number of samples drawn from the uniform

[3] It is important to note that the original hardness result of [Reg05] for search-LWE is for a *continuous* Gaussian error distribution, which when rounded naïvely to the nearest integer does not produce a true discrete Gaussian $D_{\mathbb{Z},r}$. Fortunately, a suitable randomized rounding method does so [Pei10].

distribution $U(R_q \times R_q)$. We will use the error distribution χ over R where each coefficient (with respect to the power basis) is chosen independently from the discrete Gaussian $D_{\mathbb{Z},r}$ for some $r = \alpha q \geq \omega(\sqrt{n \log n})$.

For a prime modulus $q = 1 \bmod 2n$ and the error distribution χ described above, the decision-RLWE problem is as hard as the search problem, via a reduction that runs in time $q \cdot \mathrm{poly}(n)$ [LPR10]. In turn, the search problem is as hard as quantumly approximating worst-case problems on ideal lattices.[4]

To bound products of samples drawn from the error distribution χ over R, we recall a useful result from [LPR10].

Lemma 2.3. *Let χ be the distribution over the ring R where each coefficient (with respect to the power basis) is chosen independently from $D_{\mathbb{Z},r}$ for some $r > 0$, and let $t = \omega(\sqrt{\log n})$ denote any function that grows asymptotically faster than $\sqrt{\log n}$. Then in the product of $k \geq 1$ independent samples drawn from χ, every coefficient is bounded in magnitude by $(r\sqrt{n} \cdot t)^k/\sqrt{n}$, except with $\exp(-\Omega(t^2)) = \mathrm{negl}(n)$ probability.*

2.3 Subgaussian Distributions and Random Matrices

A random variable X over $\mathbb{R}$ (or its distribution) with $\mathbb{E}[X] = 0$ is *subgaussian* with parameter $r > 0$ if it has Gaussian tails, i.e., for all $t > 0$, $\Pr[|X| > t] \leq 2\exp(-\pi(t/r)^2)$.[5] In particular, $D_{\mathbb{Z},r}$ is subgaussian with parameter r [Ban95]. Here we recall a useful result from the non-asymptotic theory of random matrices [Ver11], which bounds the largest singular value (sometimes called the *spectral norm*) $s_1(\mathbf{X}) := \max_{\mathbf{u} \neq \mathbf{0}} \|\mathbf{X}\mathbf{u}\|/\|\mathbf{u}\|$ of a matrix with independent subgaussian entries.

Lemma 2.4. *Let $\mathbf{X} \in \mathbb{R}^{n \times m}$ be a random matrix (or vector) whose entries are drawn independently from (not necessarily identical) subgaussian distributions with common parameter r. There exists a universal constant $C > 0$ such that $s_1(\mathbf{X}) \leq r \cdot C(\sqrt{m} + \sqrt{n})$ except with probability at most $2^{-\Omega(m+n)}$.*

3 The Learning With Rounding Problem

We now define the "learning with rounding" (LWR) problem and its ring analogue, which are like "derandomized" versions of the usual (ring)-LWE problems, in that the error terms are chosen deterministically.

Definition 3.1. Let $n \geq 1$ be the main security parameter and moduli $q \geq p \geq 2$ be integers.

- For a vector $\mathbf{s} \in \mathbb{Z}_q^n$, define the LWR distribution $L_{\mathbf{s}}$ to be the distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_p$ obtained by choosing a vector $\mathbf{a} \leftarrow \mathbb{Z}_q^n$ uniformly at random, and outputting $(\mathbf{a}, b = \lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rceil_p)$.
- For $s \in R_q$ (defined in Section 2.2), define the ring-LWR (RLWR) distribution L_s to be the distribution over $R_q \times R_p$ obtained by choosing $a \leftarrow R_q$ uniformly at random and outputting $(a, b = \lfloor a \cdot s \rceil_p)$.

For a given distribution over $\mathbf{s} \in \mathbb{Z}_q^n$ (e.g., the uniform distribution), the decision-LWR$_{n,q,p}$ problem is to distinguish (with advantage non-negligible in n) between any desired number of independent samples $(\mathbf{a}_i, b_i) \leftarrow L_{\mathbf{s}}$, and the same number of samples drawn uniformly and independently from $\mathbb{Z}_q^n \times \mathbb{Z}_p$. The decision-RLWR$_{R,q,p}$ problem is defined analogously.

[4] More accurately, to prove that the search problem is hard for an a priori *unbounded* number of RLWE samples, the worst-case connection from [LPR10] requires the error distribution's parameters to themselves be chosen at random from a certain distribution. Our constructions are easily modified to account for this subtlety, but for simplicity, we ignore this issue and assume hardness for a fixed, public error distribution.

[5] This simple definition will suffice for our purposes, because we will always use mean-zero distributions. For a more general definition that applies to any distribution, see [MP11].

Note that we have defined LWR exclusively as a decision problem, as this is the only form of the problem we will need. By a simple (and by now standard) hybrid argument, the (ring-)LWR problem is no easier, up to a $\mathrm{poly}(n)$ factor in advantage, if we reuse each public $\mathbf{a}_i$ across several independent secrets. That is, distinguishing samples $(\mathbf{a}_i, \lfloor \langle \mathbf{a}_i, \mathbf{s}_1 \rangle \rceil_p, \ldots, \lfloor \langle \mathbf{a}_i, \mathbf{s}_\ell \rangle \rceil_p) \in \mathbb{Z}_q^n \times \mathbb{Z}_p^\ell$ from uniform, where each $\mathbf{s}_j \in \mathbb{Z}_q^n$ is chosen independently for any $\ell = \mathrm{poly}(n)$, is at least as hard as decision-LWR for a single secret $\mathbf{s}$. An analogous statement also holds for ring-LWR.

3.1 Reduction from LWE

We now show that for appropriate parameters, decision-LWR is at least as hard as decision-LWE. We say that a probability distribution χ over $\mathbb{R}$ (more precisely, a family of distributions χ_n indexed by the security parameter n) is *B-bounded* (where $B = B(n)$ is a function of n) if $\Pr_{x \leftarrow \chi}[|x| > B] \leq \mathrm{negl}(n)$. Similarly, a distribution over the ring R is B-bounded if the marginal distribution of every coefficient (with respect to the power basis) of an $x \leftarrow \chi$ is B-bounded.

Theorem 3.2. *Let χ be any efficiently sampleable B-bounded distribution over $\mathbb{Z}$, and let $q \geq p \cdot B \cdot n^{\omega(1)}$. Then for any distribution over the secret $\mathbf{s} \in \mathbb{Z}_q^n$, solving decision-$\mathsf{LWR}_{n,q,p}$ is at least as hard as solving decision-$\mathsf{LWE}_{n,q,\chi}$ for the same distribution over $\mathbf{s}$. The same holds true for $\mathsf{RLWR}_{R,q,p}$ and $\mathsf{RLWE}_{R,q,\chi}$, for any B-bounded χ over R.*

We note that although our proof uses a super-polynomial $q = n^{\omega(1)}$, as long as $q/p \geq \sqrt{n}$ is an integer, the LWR problem appears to be exponentially hard (in n) for any $p = \mathrm{poly}(n)$, and super-polynomially hard for $p \leq 2^{n^\epsilon}$ for any $\epsilon < 1$, given the state of the art in noisy learning algorithms [BKW03, AG11] and lattice reduction algorithms [LLL82, Sch87]. We also note that in our proof, we do not require the error terms drawn from χ in the LWE samples to be independent; we just need them all to have magnitude bounded by B with overwhelming probability.

Proof of Theorem 3.2. We give a detailed proof for the LWR case; the one for RLWR proceeds essentially identically. The main idea behind the reduction is simple: given pairs $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ which are distributed either according to an LWE distribution $A_{\mathbf{s},\chi}$ or are uniformly random, we translate them into the pairs $(\mathbf{a}_i, \lfloor b_i \rceil_p) \in \mathbb{Z}_q^n \times \mathbb{Z}_p$, which we show will be distributed according to the LWR distribution $L_{\mathbf{s}}$ (with overwhelming probability) or uniformly random, respectively. Proving this formally takes some care, however. We proceed via a sequence of games.

Game H_0. This is the real attack game against the LWR distribution. That is, we choose $\mathbf{s}$ and upon request generate and give the attacker independent samples from $L_{\mathbf{s}}$.

Game H_1. Here the attack is against a 'rounded' version of the LWE distribution $A_{\mathbf{s},\chi}$. That is, we first choose $\mathbf{s}$. Then each time the attacker requests a sample, we generate a pair $(\mathbf{a}, b)$ distributed according to $A_{\mathbf{s},\chi}$ (that is, choose $\mathbf{a} \leftarrow \mathbb{Z}_q^n$ and $b = \langle \mathbf{a}, \mathbf{s} \rangle + x$ for $x \leftarrow \chi$), and return the pair $(\mathbf{a}, \lfloor b \rceil_p)$, but with one exception: we define a 'bad event' BAD to be

$$\mathsf{BAD} := \lfloor b + [-B, B] \rceil_p \neq \{ \lfloor b \rceil_p \} .$$

That is, BAD indicates whether b is "too close" to some value in $\mathbb{Z}_q$ having a different rounded value. (In other words, rounding the sample $(\mathbf{a}, b)$ from $A_{\mathbf{s},\chi}$ may give a different result than the corresponding sample

$(\mathbf{a}, \lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rceil_p)$ from the $L_{\mathbf{s}}$ distribution.) If BAD occurs on any of the attacker's requests for a sample, we immediately abort the game.

If BAD does not occur for a particular sample $(\mathbf{a}, b)$, then we have $\lfloor b \rceil_p := \lfloor \langle \mathbf{a}, \mathbf{s} \rangle + x \rceil_p = \lfloor \langle \mathbf{a}, \mathbf{s} \rangle \rceil_p$ with overwhelming probability over the choice of $x \leftarrow \chi$, because χ is B-bounded. It immediately follows that for any (potentially unbounded) attacker $\mathcal{A}$,

$$\mathbf{Adv}_{H_0,H_1}(\mathcal{A}) \leq \Pr[\mathsf{BAD} \text{ occurs in } H_1 \text{ with attacker } \mathcal{A}] + \mathrm{negl}(n). \tag{3.1}$$

We do not directly bound the probability of BAD occurring in H_1, instead deferring it to the analysis of the next game, where we can show that it is indeed negligible.

Game H_2. Here whenever the attacker requests a sample, we choose $(\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ uniformly at random and give $(\mathbf{a}, \lfloor b \rceil_p)$ to the attacker, subject to the same "bad event" and abort condition as described in the game H_1 above. Under the decision-LWE assumption and by the fact that BAD can be tested efficiently given b, a straightforward reduction implies that $\mathbf{Adv}_{H_1,H_2}(\mathcal{A}) \leq \mathrm{negl}(n)$ for any efficient attacker $\mathcal{A}$. For the same reason, it also follows that

$$|\Pr[\mathsf{BAD} \text{ occurs in } H_1] - \Pr[\mathsf{BAD} \text{ occurs in } H_2]| \leq \mathrm{negl}(n).$$

Now for each uniform b, $\Pr[\mathsf{BAD} \text{ occurs on } b \text{ in } H_2] \leq (2B+1) \cdot p/q = \mathrm{negl}(n)$, by assumption on q. It follows by a union bound over all the samples, and Equation (3.1), that

$$\Pr[\mathsf{BAD} \text{ occurs in } H_1 \text{ with } \mathcal{A}] \leq \mathrm{negl}(n) \quad \Rightarrow \quad \mathbf{Adv}_{H_0,H_1}(\mathcal{A}) = \mathrm{negl}(n).$$

Game H_3. This game is similar to the game H_2, with pairs $(\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ being chosen uniformly at random and BAD being defined similarly. However, in this game we always return $(\mathbf{a}, \lfloor b \rceil_p)$ to the attacker, even when BAD occurs. By the analysis above, we have that for any (potentially unbounded) attacker $\mathcal{A}$,

$$\mathbf{Adv}_{H_2,H_3}(\mathcal{A}) \leq \Pr[\mathsf{BAD} \text{ occurs in } H_3 \text{ with } \mathcal{A}] = \Pr[\mathsf{BAD} \text{ occurs in } H_2 \text{ with } \mathcal{A}] = \mathrm{negl}(n).$$

Game H_4. In this game we give the attacker samples drawn uniformly from $\mathbb{Z}_q^n \times \mathbb{Z}_p$. The statistical distance between $U(\mathbb{Z}_q^n \times \mathbb{Z}_p)$ and $U(\mathbb{Z}_q^n) \times \lfloor U(\mathbb{Z}_q) \rceil_p$ is at most $p/q = \mathrm{negl}(n)$ by assumption on q, so by a union bound over all the $\mathrm{poly}(n)$ samples, we have $\mathbf{Adv}_{H_3,H_4}(\mathcal{A}) = \mathrm{negl}(n)$ for any efficient attacker $\mathcal{A}$.

Finally, by the triangle inequality, we have $\mathbf{Adv}_{H_0,H_4}(\mathcal{A}) = \mathrm{negl}(n)$ for any efficient adversary $\mathcal{A}$, which completes the proof. Essentially the same proof works for the RLWR problem as well. □

4 Synthesizer-Based PRFs

We now describe the LWR-based synthesizer and our construction of a PRF from it. We first define a *pseudorandom synthesizer*, slightly modified from the definition proposed by Naor and Reingold [NR95].

Let $S : A \times A \to B$ be a function (where A and B are finite domains, which along with S are implicitly indexed by the security parameter n) and let $X = (x_1, \ldots, x_k) \in A^k$ and $Y = (y_1, \ldots, y_\ell) \in A^\ell$ be two sequences of inputs. Then $\mathbf{C}_S(X, Y) \in B^{k \times \ell}$ is defined to be the matrix with $S(x_i, y_j)$ as its (i, j)th entry. (Here $\mathbf{C}$ stands for combinations.)

Definition 4.1 (Pseudorandom Synthesizer)**.** We say that a function $S : A \times A \to B$ is a *pseudorandom synthesizer* if it is polynomial-time computable, and if for every $\mathrm{poly}(n)$-bounded $k = k(n)$, $\ell = \ell(n)$,

$$\mathbf{C}_S\big(\, U(A^k)\,,\, U(A^\ell)\,\big) \stackrel{c}{\approx} U\big(B^{k\times\ell}\big).$$

That is, the matrix $\mathbf{C}_S(X, Y)$ for uniform and independent $X \leftarrow A^k$, $Y \leftarrow A^\ell$ is computationally indistinguishable from a uniformly random k-by-ℓ matrix over B.

4.1 Synthesizer Constructions

We now describe synthesizers whose security is based on the (ring-)LWR problem.

Definition 4.2 ((Ring-)LWR Synthesizer)**.** For moduli $q > p \geq 2$, the LWR synthesizer is the function $S_{n,q,p} \colon \mathbb{Z}_q^n \times \mathbb{Z}_q^n \to \mathbb{Z}_p$ defined as

$$S_{n,q,p}(\mathbf{x}, \mathbf{y}) = \lfloor \langle \mathbf{x}, \mathbf{y} \rangle \rceil_p.$$

The RLWR synthesizer is the function $S_{R,q,p} \colon R_q \times R_q \to R_p$ defined as

$$S_{R,q,p}(x, y) = \lfloor x \cdot y \rceil_p.$$

Theorem 4.3. *Assuming the hardness of decision-*$\mathsf{LWR}_{n,q,p}$ *(respectively, decision-*$\mathsf{RLWR}_{R,q,p}$*) for a uniformly random secret, the function* $S_{n,q,p}$ *(respectively,* $S_{R,q,p}$*) given in Definition 4.2 above is a pseudorandom synthesizer.*

It follows generically from this theorem that the function $T_{n,q,p} \colon \mathbb{Z}_q^{n\times n} \times \mathbb{Z}_q^{n\times n} \to \mathbb{Z}_p^{n\times n}$, defined as $T_{n,q,p}(\mathbf{X}, \mathbf{Y}) = \lfloor \mathbf{X} \cdot \mathbf{Y} \rceil_p$, is also a pseudorandom synthesizer, since by the definition of matrix multiplication, we only incur a factor of n increase in the length of the input sequences. This is the synthesizer that we use below in the construction of a PRF.

Proof of Theorem 4.3. Let $\ell, k = \mathrm{poly}(n)$ be arbitrary. Let $X = (\mathbf{x}_1, \ldots, \mathbf{x}_k)$ and $Y = (\mathbf{y}_1, \ldots, \mathbf{y}_\ell)$ be uniformly random and independent sequences of $\mathbb{Z}_q^n$-vectors. Assuming the hardness of "multiple secrets" version of decision-$\mathsf{LWR}_{n,q,p}$ (see the remark following Definition 3.1), we have that the tuples

$$\big(\mathbf{x}_i, \lfloor \langle \mathbf{x}_i, \mathbf{y}_1 \rangle \rceil_p, \ldots, \lfloor \langle \mathbf{x}_i, \mathbf{y}_\ell \rangle \rceil_p\big) \in \mathbb{Z}_q^n \times \mathbb{Z}_p^\ell$$

for $i = 1, \ldots, k$ are computationally indistinguishable from uniform and independent. That is,

$$\big((\mathbf{x}_i)_{i\in[k]}, \mathbf{C}_S(X, Y)\big) \stackrel{c}{\approx} U(\mathbb{Z}_q^{n\times k} \times \mathbb{Z}_p^{k\times\ell}).$$

From this stronger fact, we have that $\mathbf{C}_S(X, Y) \stackrel{c}{\approx} U(\mathbb{Z}_p^{k\times\ell})$, as desired. Essentially the same proof works for the RLWR synthesizer as well. □

4.2 The PRF Construction

Definition 4.4 ((Ring-)LWR PRF)**.** For parameters $n \in \mathbb{N}$, input length $k = 2^d \geq 1$, and moduli $q_d \geq q_{d-1} \geq \ldots \geq q_0 \geq 2$, the LWR family $\mathcal{F}^{(j)}$ for $0 \leq j \leq d$ is defined inductively to consist of functions from $\{0,1\}^{2^j}$ to $\mathbb{Z}_{q_{d-j}}^{n\times n}$. We define $\mathcal{F} = \mathcal{F}^{(d)}$.

- For $j = 0$, a function $F \in \mathcal{F}^{(0)}$ is indexed by $\mathbf{S}_b \in \mathbb{Z}_{q_d}^{n\times n}$ for $b \in \{0,1\}$, and is defined simply as $F_{\{\mathbf{S}_b\}}(x) = \mathbf{S}_x$. We endow $\mathcal{F}^{(0)}$ with the distribution where the $\mathbf{S}_b$ are uniform and independent.

- For $j \geq 1$, a function $F \in \mathcal{F}^{(j)}$ is indexed by some $F_0, F_1 \in \mathcal{F}^{(j-1)}$, and is defined as

$$F_{F_0,F_1}(x_0, x_1) = T^{(j)}\big(\, F_0(x_0)\,,\, F_1(x_1)\,\big)$$

  where $|x_0| = |x_1| = 2^{j-1}$ and $T^{(j)} = T_{n,q_{d-j+1},q_{d-j}}$ is the appropriate synthesizer. We endow $\mathcal{F}^{(j)}$ with the distribution where F_0 and F_1 are chosen independently from $\mathcal{F}^{(j-1)}$.

More explicitly, $F \in \mathcal{F}$ is indexed by a set of matrices $\{\mathbf{S}_{i,b}\}$ (where $i \in [k]$, $b \in \{0,1\}$), and for input $x = x_1 \cdots x_k$ is defined as

$$F_{\{\mathbf{S}_{i,b}\}}(x) \;=\; \Big\lfloor \cdots \Big\lfloor \lfloor \mathbf{S}_{1,x_1} \cdot \mathbf{S}_{2,x_2} \rceil_{q_{d-1}} \cdot \lfloor \mathbf{S}_{3,x_3} \cdot \mathbf{S}_{4,x_4} \rceil_{q_{d-1}} \Big\rceil_{q_{d-2}} \cdots \lfloor \mathbf{S}_{k-1,x_{k-1}} \cdot \mathbf{S}_{k,x_k} \rceil_{q_{d-1}} \cdots \Big\rceil_{q_0} .$$

The ring-LWR family $\mathcal{RF}^{(j)}$ is defined similarly to consist of functions from $\{0,1\}^{2^j}$ to $R_{q_{d-j}}$, where in the base case ($j = 0$) we replace each $\mathbf{S}_b$ with a uniformly random $s_b \in R_{q_d}$, and in the inductive case ($j \geq 1$) we use the ring-LWR synthesizer $S^{(j)} = S_{R,q_{d-j+1},q_{d-j}}$.

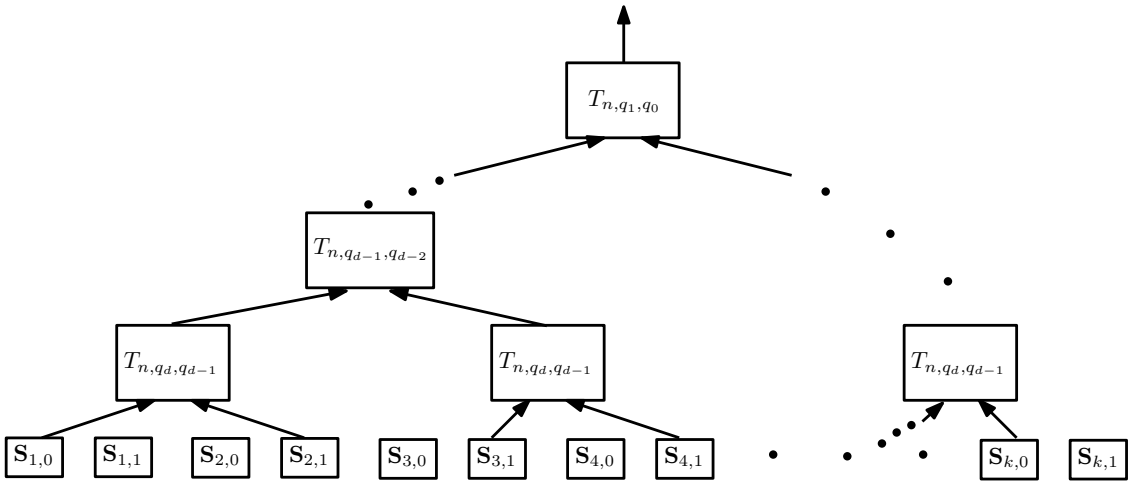

Figure 1: The synthesizer-based PRF evaluated on the input $0111\ldots0$

We remark that the recursive LWR-based construction above does not have to use *square* matrices; any legal dimensions would be acceptable with no essential change to the security proof. Square matrices appear to give the best combination of seed size, computational efficiency, and input/output lengths.

4.3 Efficiency

Consider a function in either one of the families $\mathcal{F}$ or $\mathcal{RF}$ from Definition 4.4. Computing the function at any given point $x \in \{0,1\}^k$ can be done in a tree-like fashion using a tree of depth $d = \lg k$, where each node of the tree corresponds to an evaluation of an appropriate synthesizer. Each synthesizer involves a single matrix (or ring) product mod q_{d-j+1}, followed by a rounding step. Here we discuss implementations of the synthesizers, describing both the simplest practical methods along with depth-optimized parallel solutions (which rely on preprocessing and use larger circuits). In summary, the synthesizers can be computed by small, low-depth arithmetic circuits; moreover, in principle they can be implemented in TC^0, the class of

constant-depth, $\mathrm{poly}(n)$-sized circuits with unbounded fan-in and threshold gates (which is a subset of NC^1). Therefore, the PRFs can be implemented in TC^1, which matches the best constructions from [NR95].

In a practical sequential implementation, we can use any fast matrix multiplication algorithm (e.g., Strassen's), and we can multiply ring elements (in the standard power basis) in $O(n \log n)$ scalar operations mod q (see, e.g., [LPR10]). In a practical parallel implementation, we can compute a matrix multiplication in the natural way using a size-$O(n^2)$, depth-2 arithmetic circuit over $\mathbb{Z}_q$, where the first layer of multiplication gates have fan-in 2 and the second layer of addition gates has fan-in n. The same is true for a product of ring elements in R_q, since it can be expressed as a matrix-vector product: multiplication by any fixed element $a \in R_q$ is a linear transformation.

For computing a synthesizer $T_{n,q,p}$ in TC^0, we note that a matrix product consists of n^2 parallel inner products of n-dimensional vectors, which each involve a multi-sum of (binary) scalar products modulo q. The subsequent rounding step simply amounts to dropping some of the least-significant digits if q and p are both powers of the same small base, or more generally, multiplying by p/q (under suitable precision) and truncating. Both operations can be performed in TC^0, for any $q = 2^{\mathrm{poly}(n)}$ [RT92].

Interestingly, once we allow for threshold gates, there seems to be no asymptotic improvement in depth for the ring-based synthesizer $S_{R,q,p}$. This is because threshold circuits enable binary matrix product to be computed in constant depth, and the depth of computing the PRF is anyway dominated by d, the depth of the tree. The gains in efficiency obtained by using a ring-based construction will be much more pronounced in the case of the degree-k synthesizers described in Section 5. We discuss these gains in detail in Section 5.2.

We remark that Naor and Reingold [NR95] describe several nice optimizations and additional features of their synthesizer-based PRFs, including compression of the secret key and faster amortized computation for a sequence of related inputs. Our functions are amenable to all these techniques as well.

4.4 Security Proof

The security proof for our PRF hinges on the fact that the functions $T^{(j)} = T_{n,q_{d-j+1},q_{d-j}}$ are synthesizers for appropriate choices of the moduli. In fact, the proof is essentially identical to Naor and Reingold's [NR95] for their PRF construction from pseudorandom synthesizers; the only reason we cannot use their theorem exactly as stated is because they assume that the synthesizer output is exactly the same size as its two inputs, which is not quite the case with our synthesizer due to the modulus reduction. This is a minor detail that does not change the proof in any material way; it only limits the number of times we may compose the synthesizer, and hence the input length of the PRF.

Theorem 4.5. *Assuming that $T^{(j)} = T_{n,q_{d-j+1},q_{d-j}}$ is a pseudorandom synthesizer for every $j \in [d]$ (in particular, assuming the hardness of decision-$\mathsf{LWR}_{n,q_{d-j+1},q_{d-j}}$), the LWR family $\mathcal{F}$ from Definition 4.4 is a pseudorandom function family.*

The same holds for the ring-LWR family $\mathcal{RF}$, assuming that $S^{(j)} = S_{R,q_{d-j+1},q_{d-j}}$ is a pseudorandom synthesizer for every $j \in [d]$ (in particular, assuming the hardness of decision-$\mathsf{RLWR}_{R,q_{d-j+1},q_{d-j}}$).

Proof. We give a detailed proof for the family $\mathcal{F}$; the one for $\mathcal{RF}$ proceeds essentially identically. We prove that each $\mathcal{F}^{(j)}$ is a pseudorandom function family by induction for $j = 0, \ldots, d$. The case $j = 0$ is trivial by construction of $\mathcal{F}^{(0)}$. Assuming the inductive hypothesis on $\mathcal{F}^{(j-1)}$ for some $j \geq 1$, we prove the claim for $\mathcal{F}^{(j)}$ via the following series of games.

Game H_0. This is the PRF attack game against $\mathcal{F}^{(j)}$: we choose an $F \leftarrow \mathcal{F}^{(j)}$, i.e., choose $F_0, F_1 \leftarrow \mathcal{F}^{(j-1)}$ independently, and give the attacker oracle access to $F_{F_0,F_1}(x_0, x_1) = T^{(j)}(F_0(x_0), F_1(x_1))$, where as always $|x_0| = |x_1| = 2^{j-1}$.

Game H_1. We replace F_0, F_1 above with truly uniform functions. Specifically, we (lazily) choose two uniform and independent functions $U_0, U_1 \colon \{0,1\}^{2^{j-1}} \to \mathbb{Z}_{q_{d-j+1}}^{n \times n}$. On each query $x = (x_0, x_1) \in \{0,1\}^{2^j}$, we return $T^{(j)}(U_0(x_0), U_1(x_1))$. By a trivial reduction using the inductive hypothesis that $\mathcal{F}^{(j-1)}$ is a PRF family (so F_0, F_1 are computationally indistinguishable from U_0, U_1 given query access), this game is computationally indistinguishable from H_0.

Game H_2. We give the attacker oracle access to a (lazily defined) uniform function $U : \{0,1\}^{2^j} \to \mathbb{Z}_{q_{d-j}}^{n \times n}$.

We claim that games H_0 and H_1 are computationally indistinguishable, because $T^{(j)}$ is a synthesizer by hypothesis. Suppose that an efficient adversary $\mathcal{A}$ makes at most $Q = \mathrm{poly}(n)$ total queries. We design an efficient simulator $\mathcal{S}$ which, given input $(\mathbf{Z}_{i,j})_{i,j \in [Q]} \in (\mathbb{Z}_{q_{d-j}}^{n \times n})^{Q \times Q}$ where either $\mathbf{Z}_{i,j} = T^{(j)}(\mathbf{X}_i, \mathbf{Y}_j)$ for some uniformly random and independent $\mathbf{X}_i, \mathbf{Y}_j \in \mathbb{Z}_{q_{d-j+1}}^{n \times n}$ for $i, j \in [Q]$, or each $\mathbf{Z}_{i,j}$ is uniformly random and independent, simulates game H_1 or H_2, respectively. Because the two types of inputs to $\mathcal{S}$ are computationally indistinguishable by assumption on $T^{(j)}$ (and $\mathcal{S}$ is efficient), it follows that games H_1 and H_2 are indistinguishable as well.

$\mathcal{S}$ works as follows: starting from $i = j = 1$, on each query $x = (x_0, x_1) \in \{0,1\}^{2^j}$ from $\mathcal{A}$, look up whether x_0 (respectively, x_1) is already associated with an index $\hat{\imath}$ (resp., $\hat{\jmath}$); if not, associate it with the current value of i (resp., j) and increment that variable. Return the associated matrix $\mathbf{Z}_{\hat{\imath},\hat{\jmath}}$ to $\mathcal{A}$. It is clear by inspection that the behavior of $\mathcal{S}$ is as claimed above.

We conclude that game H_0 is computationally indistinguishable from game H_2, i.e., that $\mathcal{F}^{(j)}$ is a pseudorandom function family, as desired. □

5 Direct PRF Constructions

Here we present another, potentially more efficient construction of a pseudorandom function family whose security is based on the intractibility of the LWE problem.

5.1 Constructions

Definition 5.1 ((Ring-)LWE degree-k PRF)**.** For parameters $n \in \mathbb{N}$, moduli $q \geq p \geq 2$, positive integer $m = \mathrm{poly}(n)$, and input length $k \geq 1$, the family $\mathcal{F}$ consists of functions from $\{0,1\}^k$ to $\mathbb{Z}_p^{m \times n}$. A function $F \in \mathcal{F}$ is indexed by some $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{S}_i \in \mathbb{Z}^{n \times n}$ for each $i \in [k]$, and is defined as

$$F(x) = F_{\mathbf{A},\{\mathbf{S}_i\}}(x_1 \cdots x_k) := \left\lfloor \mathbf{A}^t \cdot \prod_{i=1}^{k} \mathbf{S}_i^{x_i} \right\rceil_p . \tag{5.1}$$

We endow $\mathcal{F}$ with the distribution where $\mathbf{A}$ is chosen uniformly at random, and below we consider a number of natural distributions for the $\mathbf{S}_i$.

The ring-based family $\mathcal{RF}$ is defined similarly to consist of functions from $\{0,1\}^k$ to R_p, where we replace $\mathbf{A}$ with uniformly random $a \in R_q$ and each $\mathbf{S}_i$ with some $s_i \in R$.

5.2 Efficiency

Consider a function $F \in \mathcal{F}$ as in Definition 5.1. Computing the function involves a subset-product of matrices. Generally speaking, matrix multi-product does not appear to be computable in TC^0 (if it were, then TC^0 would equal NC^1 [MP00]). However, in our case the matrices are known in advance (the variable input

is the subset, so it may be possible to reduce the depth of the computation via preprocessing, using ideas from [RT92]. As described in Section 4.3, both binary matrix product and rounding can be implemented with simple depth-2 arithmetic circuits, and hence in TC^0, so at worst F can be computed in TC^1 by computing the subset product in a tree-like fashion, followed by a final rounding step.

The ring variant of Construction 5.1 appears to be more efficient to evaluate, both in practice and in terms of the best theoretical depth. Consider a function $F \in \mathcal{RF}$ as in Definition 5.1. As is standard with ring-based primitives (see, e.g., [LMPR08, LPR10]), one could store the ring elements $a, s_1, \ldots, s_k$ as vectors in $\mathbb{Z}_q^n$ using the discrete Fourier transform or "Chinese remainder" representation modulo q (that is, by evaluating a and the s_i as polynomials at the n roots of $z^n + 1$ modulo q), so that multiplication of two ring elements just corresponds to a coordinate-wise product of their vectors. Then to evaluate the function, one would just compute a subset-product of the appropriate vectors, then interpolate the result to the power-basis representation, using essentially an n-dimensional Fast Fourier Transform over $\mathbb{Z}_q$, in order to perform the rounding operation. For the interesting case of $k = \omega(\log n)$, the sequential runtime of this method is dominated by the kn scalar multiplications in $\mathbb{Z}_q$ to compute the subset-product; in parallel, the arithmetic depth (over $\mathbb{Z}_q$) is $O(\log(nk))$. Alternatively, the subset-product part of the function might be computed even faster by storing the *discrete logs*, with respect to some arbitrary generator g of $\mathbb{Z}_q^*$, of the Fourier coefficients of a and s_i.[6] The subset-product then becomes a subset-sum, followed by exponentiation modulo q, or even just a table lookup if q is relatively small. Assuming that additions mod $q - 1$ are significantly less expensive than multiplications mod q, the sequential runtime of this method is dominated by the $O(n \log n)$ scalar operations in the FFT, and the parallel arithmetic depth is again $O(\log n)$.

In terms of theoretical depth, the multi-product of vectors can be performed in TC^0, as can the Fast Fourier Transform and rounding steps [RT92]. This implies that the entire function can be computed in TC^0, matching (asymptotically) the shallowest known PRFs based on the DDH and factoring problems [NR97, NRR00].

5.3 Security Proof Under LWE

Our first theorem says that when the entries of the $\mathbf{S}_i$ are "small," i.e., chosen from a suitable LWE error distribution, the degree-k construction is a PRF under a suitable LWE assumption.

Theorem 5.2. *Let $\chi = D_{\mathbb{Z},r}$ for some $r > 0$, and let $q \geq p \cdot k(Cr\sqrt{n})^k \cdot n^{\omega(1)}$ for a suitable universal constant C. Endow the family $\mathcal{F}$ from Definition 5.2 with the distribution where each $\mathbf{S}_i$ is drawn independently from $\chi^{n \times n}$. Then assuming the hardness of decision-$\mathsf{LWE}_{n,q,\chi}$, the family $\mathcal{F}$ is pseudorandom.*

An analogous theorem holds for the ring-based family $\mathcal{RF}$, under decision-RLWE.

Theorem 5.3. *Let χ be the distribution over the ring R where each coefficient (with respect to the power basis) is chosen independently from $D_{\mathbb{Z},r}$ for some $r > 0$, and let $q \geq p \cdot k(r\sqrt{n} \cdot \omega(\sqrt{\log n}))^k \cdot n^{\omega(1)}$. Endow the family $\mathcal{RF}$ from Definition 5.2 with the distribution where each s_i is drawn independently from χ. Then assuming the hardness of decision-$\mathsf{RLWE}_{n,q,\chi}$, the family $\mathcal{RF}$ is pseudorandom.*

We first prove Theorem 5.2 for the standard LWE construction.

Proof of Theorem 5.2. To aid the proof, it helps to define a family $\mathcal{G}$ of functions $G\colon \{0,1\}^k \to \mathbb{Z}_q^{n \times n}$, which are simply the unrounded counterparts of the functions in $\mathcal{F}$. That is, for $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{S}_i \in \mathbb{Z}^{n \times n}$ for $i \in [k]$, we define $G_{\mathbf{A},\{\mathbf{S}_i\}}(x_1 \cdots x_k) := \mathbf{A}^t \cdot \prod_{i=1}^k \mathbf{S}_i^{x_i}$. We endow $\mathcal{G}$ with the same distribution over $\mathbf{A}$ and the $\mathbf{S}_i$ as $\mathcal{F}$ has.

[6] If necessary, one would also store binary mask vectors indicating which Fourier coefficients are zero, and hence not in $\mathbb{Z}_q^*$.

We proceed via a sequence of games, much like in the proof of Theorem 3.2. First as a "thought experiment" we define a new family $\tilde{\mathcal{G}}$ of functions from $\{0,1\}^k$ to $\mathbb{Z}_q^{m \times n}$. This family is a counterpart to $\mathcal{G}$, but with two important differences: it is a PRF family *without* any rounding (and hence, with rounding as well), but each function in the family has an exponentially large key. Alternatively, one may think of the functions in $\tilde{\mathcal{G}}$ as *randomized* functions with small keys. Then we show that with overwhelming probability, the rounding of $\tilde{G} \leftarrow \tilde{\mathcal{G}}$ agrees with the rounding of the corresponding $G \in \mathcal{G}$ on all the attacker's queries, because the outputs of the two functions are relatively close. It follows that the rounding of $G \leftarrow \mathcal{G}$ (i.e., $F \leftarrow \mathcal{F}$) cannot be distinguished from a uniformly random function, as desired.

More formally, we define the following games:

Game H_0. This is the real PRF attack game against the family $\mathcal{F}$: we choose an $F \leftarrow \mathcal{F}$ (so $F(\cdot) = \lfloor G(\cdot) \rceil_p$ for $G \leftarrow \mathcal{G}$), and the attacker has oracle access to $F(\cdot)$.

Game H_1. Here we instead choose $\tilde{G} \leftarrow \tilde{\mathcal{G}}$, where the family $\tilde{\mathcal{G}}$ is given in Definition 5.4 below. The choice of $\tilde{G}$ induces a corresponding $G \in \mathcal{G}$ having the same distribution as in H_0. (This is simply because the key of G is just a portion of the key of $\tilde{G}$.) To be precise, we choose $\tilde{G}$ "lazily" as the attacker makes queries, because the description of $\tilde{G}$ has exponential size; see the remarks following Definition 5.4 for details.

The attacker has oracle access to $\lfloor \tilde{G}(\cdot) \rceil_p$, but with one exception: on query x, define the "bad event" BAD_x for that query to be

$$\left\lfloor \tilde{G}(x) + [-B, B]^{m \times n} \right\rceil_p \neq \{ \lfloor \tilde{G}(x) \rceil_p \},$$

where $B = k(Cr\sqrt{n})^k$. That is, BAD_x indicates whether any entry of $\tilde{G}(x) \in \mathbb{Z}_q^{m \times n}$ is "too close" to another element of $\mathbb{Z}_q$ that rounds to a different value in $\mathbb{Z}_p$. Note that a $y \in \mathbb{Z}_q$ is "too close" in this sense if and only if $\lfloor (\bar{y} - B) \cdot \frac{p}{q} \rceil \neq \lfloor (\bar{y} + B) \cdot \frac{p}{q} \rceil \in \mathbb{Z}$, where $\bar{y} \in \mathbb{Z}$ is any integer congruent to $y \bmod q$, so BAD_x can be efficiently detected given only the value of $\tilde{G}(x)$. If BAD_x occurs any of the attacker's queries, then the game immediately aborts.

In Lemma 5.5 below, we show that for every fixed $x \in \{0,1\}^k$, with overwhelming probability over the choice of $\tilde{G} \leftarrow \tilde{\mathcal{G}}$ and the induced $G \in \mathcal{G}$, it is the case that $G(x) \in \tilde{G}(x) + [-B, B]^{m \times n} \bmod q$. Hence $\lfloor G(x) \rceil_p = \lfloor \tilde{G}(x) \rceil_p$ so long as BAD_x does not occur, and the attacker's queries are answered exactly as they are in H_0, subject to the game not aborting. It follows that for any (potentially unbounded) attacker $\mathcal{A}$,

$$\mathbf{Adv}_{H_0,H_1}(\mathcal{A}) \leq \Pr[\text{some } \mathsf{BAD}_x \text{ occurs in } H_1 \text{ with attacker } \mathcal{A}] + \mathrm{negl}(n). \tag{5.2}$$

We do not directly bound the probability that some BAD_x occurs in H_1, but instead defer to the analysis of the next game, where we can show that it is indeed negligible.

Game H_2. Here we choose U to be a uniformly random function from $\{0,1\}^k$ to $\mathbb{Z}_q^{m \times n}$ (defined "lazily" as the attacker makes queries). The attacker has oracle access to $\lfloor U(\cdot) \rceil_p$, with the same "bad event" and abort condition as in H_1, but defined relative to U instead of $\tilde{G}$.

In Theorem 5.6 below, we show that under the LWE assumption from the theorem statement, no efficient adversary can distinguish (given oracle access) between $\tilde{G} \leftarrow \tilde{\mathcal{G}}$ and a uniformly random function U. Because the BAD_x event in H_1 (respectively, H_2) for a query x can be tested efficiently given query access to $\tilde{G}$ (resp., U), a trivial simulation implies that for any efficient attacker $\mathcal{A}$, we have $\mathbf{Adv}_{H_1,H_2}(\mathcal{A}) \leq \mathrm{negl}(n)$. For the same reasons, it also follows by a straightforward simulation that for any efficient attacker $\mathcal{A}$,

$$|\Pr[\text{some } \mathsf{BAD}_x \text{ occurs in } H_1 \text{ with } \mathcal{A}] - \Pr[\text{some } \mathsf{BAD}_x \text{ occurs in } H_2 \text{ with } \mathcal{A}]| \leq \mathrm{negl}(n).$$

In H_2, because U is a uniformly random function, for any particular query x the probability that BAD_x occurs is bounded by $(2B+1) \cdot p/q = \mathrm{negl}(n)$, by assumption on q. By a union bound over all $\mathrm{poly}(n)$ queries of an efficient $\mathcal{A}$, and then applying Equation (5.2), we therefore have that

$$\Pr[\text{some } \mathsf{BAD}_x \text{ occurs in } H_1 \text{ with } \mathcal{A}] = \mathrm{negl}(n) \quad \Rightarrow \quad \mathbf{Adv}_{H_0,H_1}(\mathcal{A}) = \mathrm{negl}(n).$$

Game H_3. Here we still choose a uniformly random function U and give the attacker oracle access to $\lfloor U(\cdot) \rceil_p$. For each query x we define the event BAD_x as in game H_2, but still answer the query and continue with the game even if BAD_x occurs. From the above analysis of H_2 it follows that for any (potentially unbounded) attacker $\mathcal{A}$ making $\mathrm{poly}(n)$ queries, we have

$$\mathbf{Adv}_{H_2,H_3}(\mathcal{A}) \leq \Pr[\text{some } \mathsf{BAD}_x \text{ occurs in } H_2 \text{ with } \mathcal{A}] = \mathrm{negl}(n).$$

Finally, observe that $\lfloor U(\cdot) \rceil_p$ is a truly random function from $\{0,1\}^k$ to $\mathbb{Z}_p^{m \times n}$, up to the bias involved in rounding the uniform distribution on $\mathbb{Z}_q$ to $\mathbb{Z}_p$. Because $q \geq p \cdot n^{\omega(1)}$, this bias is negligible (and there is no bias if p divides q).

By the triangle inequality, it follows that for any efficient $\mathcal{A}$, we have $\mathbf{Adv}_{H_0,H_3}(\mathcal{A}) = \mathrm{negl}(n)$, and this completes the proof. □

We now define the family $\tilde{\mathcal{G}}$ used in the proof of Theorem 5.2.

Definition 5.4. For parameters n, q, m, k and error distribution χ (over $\mathbb{Z}$) as in Definition 5.1, the family $\tilde{\mathcal{G}}^{(i)}$ for $0 \leq i \leq k$ is defined inductively to consist of functions from $\{0,1\}^i$ to $\mathbb{Z}_q^{m \times n}$; we define $\tilde{\mathcal{G}} = \tilde{\mathcal{G}}^{(k)}$.

- For $i = 0$, a function $\tilde{G} \in \tilde{\mathcal{G}}^{(0)}$ is indexed by some $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, and is defined simply as $\tilde{G}_{\mathbf{A}}(\varepsilon) = \mathbf{A}^t$. We endow $\tilde{\mathcal{G}}^{(0)}$ with the distribution where $\mathbf{A}$ is chosen uniformly at random.
- For $i \geq 1$, a function $\tilde{G} \in \tilde{\mathcal{G}}^{(i)}$ is indexed by some $\tilde{G}' \in \tilde{\mathcal{G}}^{(i-1)}$, plus an $\mathbf{S}_i \in \mathbb{Z}^{n \times n}$ and error matrices $\mathbf{E}_{x'} \in \mathbb{Z}^{m \times n}$ for each $x' \in \{0,1\}^{i-1}$ (where $\{0,1\}^0$ is the singleton set $\{\varepsilon\}$). For $x = (x', x_i) \in \{0,1\}^i$ where $|x'| = i-1$, the function is defined as

  $$\tilde{G}(x) = \tilde{G}_{\tilde{G}', \mathbf{S}_i, \{\mathbf{E}_{x'}\}}(x', x_i) := \tilde{G}'(x') \cdot \mathbf{S}_i^{x_i} + x_i \cdot \mathbf{E}_{x'} \bmod q.$$

  We endow $\tilde{\mathcal{G}}^{(i)}$ with the distribution where $\tilde{G}' \leftarrow \tilde{\mathcal{G}}^{(i-1)}$, and all the entries of $\mathbf{S}_i$ and every $\mathbf{E}_{x'}$ are chosen independently from χ.

Note that a function $\tilde{G} \in \tilde{\mathcal{G}}$ is fully specified by $\mathbf{A}$, $\{\mathbf{S}_i\}_{i \in [k]}$, and *exponentially* (in k) many error matrices $\mathbf{E}_{x_1 \cdots x_{i-1}}$ for all $x \in \{0,1\}^k$ and $i \in [k]$; these error matrices are what prevents $\tilde{\mathcal{G}}$ itself from being used as a PRF family. However, as needed in the proof of Theorem 5.2 (game H_1), the error matrices can be chosen "lazily," since the value of $\tilde{G}(x)$ depends only on $\mathbf{A}$, $\{\mathbf{S}_i\}$, and $\mathbf{E}_{x_1 \cdots x_{i-1}}$ for $i \in [k]$. For a function $\tilde{G} = \tilde{G}_{\mathbf{A}, \{\mathbf{S}_i\}, \{\mathbf{E}_{x'}\}} \in \tilde{\mathcal{G}}$, we define its *induced* function in the family $\mathcal{G}$ to be $G = G_{\mathbf{A}, \{\mathbf{S}_i\}}$. Note that for $\tilde{G} \leftarrow \tilde{\mathcal{G}}$, the induced function G has the same marginal distribution as if it had been chosen from $\mathcal{G}$ directly.

The following lemma is used in the analysis of game H_1.

Lemma 5.5. *Let $x \in \{0,1\}^k$ be arbitrary. Then except with $2^{-\Omega(n)}$ probability over the choice of $\tilde{G} = \tilde{G}_{\mathbf{A}, \{\mathbf{S}_i\}, \{\mathbf{E}_{x'}\}} \leftarrow \tilde{\mathcal{G}}$ and its induced function $G = G_{\mathbf{A}, \{\mathbf{S}_i\}} \in \mathcal{G}$, we have*

$$G(x) \in \tilde{G}(x) + [-B, B]^{m \times n} \bmod q$$

for some $B = k \cdot (Cr\sqrt{n})^k$, where C is a universal constant.

Proof. Observe that

$$\begin{aligned}\tilde{G}(x_1 \cdots x_k) &= (\cdots((\mathbf{A}^t \cdot \mathbf{S}_1^{x_1} + x_1 \cdot \mathbf{E}_\varepsilon) \cdot \mathbf{S}_2^{x_2} + x_2 \cdot \mathbf{E}_{x_1}) \cdots) \cdot \mathbf{S}_k^{x_k} + x_k \cdot \mathbf{E}_{x_1 \cdots x_{k-1}} \bmod q \\ &= \underbrace{\mathbf{A}^t \cdot \prod_{i=1}^{k} \mathbf{S}_i^{x_i}}_{G(x)} + x_1 \cdot \mathbf{E}_\varepsilon \cdot \prod_{i=2}^{k} \mathbf{S}_i^{x_i} + x_2 \cdot \mathbf{E}_{x_1} \cdot \prod_{i=3}^{k} \mathbf{S}_i^{x_i} + \cdots + x_k \cdot \mathbf{E}_{x_1 \cdots x_{k-1}} \bmod q.\end{aligned}$$

Now by Lemma 2.4, except with probability $2^{-\Omega(n)}$, for every $i \in [k]$ we have $s_1(\mathbf{S}_i) \le O(r\sqrt{n})$ and $\|\mathbf{e}\| \le O(r\sqrt{n})$ for every row $\mathbf{e}$ of the error matrices $\mathbf{E}_{x_1 \cdots x_{i-1}}$. Therefore, each row of the k cumulative error matrices $\mathbf{E}_{x_1 \cdots x_{i-1}} \cdot \prod_{j=i+1}^{k} \mathbf{S}_j^{x_j}$ (for $i \in [k]$) has Euclidean length at most $O(r\sqrt{n})^k$, and so its entries are bounded by the same quantity in magnitude. The claim follows. □

Theorem 5.6. *Under the* LWE *assumption from the statement of Theorem 5.2, the family $\tilde{\mathcal{G}}$ of functions from $\{0,1\}^k$ to $\mathbb{Z}_q^{m \times n}$ is pseudorandom.*

In the proof we will need the following intermediate function families.

Definition 5.7. For n, q, m, and χ as in Definition 5.1, and an integer $i \ge 1$, the family $\mathcal{H}^{(i)}$ consists of functions from $\{0,1\}^i$ to $\mathbb{Z}_q^{m \times n}$. A function H from the family is indexed by some $\mathbf{S}_i \in \mathbb{Z}^{n \times n}$ and matrices $\mathbf{A}_{x'} \in \mathbb{Z}_q^{n \times m}, \mathbf{E}_{x'} \in \mathbb{Z}^{m \times n}$ for each $x' \in \{0,1\}^{i-1}$ (where $\{0,1\}^0 = \{\varepsilon\}$). It is defined as

$$H(x) = H_{\mathbf{S}_i, \{\mathbf{A}_{x'}\}, \{\mathbf{E}_{x'}\}}(x', x_i) := \mathbf{A}_{x'}^t \cdot \mathbf{S}_i^{x_i} + x_i \cdot \mathbf{E}_{x'} \bmod q,$$

where $|x'| = i - 1$. We endow $\mathcal{H}$ with the distribution where each $\mathbf{A}_{x'}$ is uniformly random and independent, and all the entries of $\mathbf{S}_i$ and $\mathbf{E}_{x'}$ are chosen independently from χ. We remark that an $H \leftarrow \mathcal{H}^{(i)}$ can be chosen "lazily" in the natural way.

Proof of Theorem 5.6. We prove that each family $\tilde{\mathcal{G}}^{(i)}$ is pseudorandom by induction on i, from 0 to k. The base case of $i = 0$ is trivial by construction. For $i \ge 1$, we prove the claim by the following series of games.

Game H_0. We (lazily) choose a $\tilde{G} \leftarrow \tilde{\mathcal{G}}^{(i)}$ and give the attacker oracle access to $\tilde{G}(\cdot)$.

Game H_1. We (lazily) choose an $H \leftarrow \mathcal{H}^{(i)}$ (defined above) and give the attacker oracle access to $H(\cdot)$.

We claim that $H_0 \overset{c}{\approx} H_1$ under the inductive hypothesis that $\tilde{\mathcal{G}}^{(i-1)}$ is a PRF family. To prove this, we design an efficient simulator $\mathcal{S}$ that is given oracle access to a function $F\colon \{0,1\}^{i-1} \to \mathbb{Z}_q^{m \times n}$, where F is either $\tilde{G}' \leftarrow \tilde{\mathcal{G}}^{(i-1)}$ or a uniformly random function, and $\mathcal{S}$ emulates either game H_0 or H_1 (respectively) to an attacker. The simulator $\mathcal{S}$ first chooses an $\mathbf{S}_i \leftarrow \chi^{n \times n}$, and on each query $x = (x', x_i)$ from the attacker where $|x'| = i - 1$, $\mathcal{S}$ queries its oracle to get $\mathbf{A}_{x'}^t = F(x')$, chooses an $\mathbf{E}_{x'} \leftarrow \chi^{m \times n}$ (if it has not already been defined by a previous query), and returns $\mathbf{A}_{x'}^t \cdot \mathbf{S}_i^{x_i} + x_i \cdot \mathbf{E}_{x'}$ to the attacker. It is clear by the definitions of $\tilde{\mathcal{G}}^{(i)}$ and $\mathcal{H}^{(i)}$ that if F is some $\tilde{G}' \leftarrow \tilde{\mathcal{G}}^{(i-1)}$, then $\mathcal{S}$ emulates access to $\tilde{G}_{\tilde{G}', \mathbf{S}_i, \{\mathbf{E}_{x'}\}} \in \tilde{\mathcal{G}}^{(i)}$ with the appropriate distribution, whereas if F is a uniformly random function, then $\mathcal{S}$ emulates access to $H \leftarrow \mathcal{H}^{(i)}$.

Game H_2. We (lazily) choose a uniformly random function $U\colon \{0,1\}^i \to \mathbb{Z}_q^{m\times n}$ and give the attacker oracle access to $U(\cdot)$.

We claim that $H_1 \overset{c}{\approx} H_2$ under the decision-LWE assumption from Theorem 5.2. To prove this, we design an efficient simulator $\mathcal{S}$ that is given access to an oracle $\mathcal{O}$ that outputs arbitrarily many pairs $(\mathbf{A}^t, \mathbf{B}^t) \in \mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^{m\times n}$, drawn either as a group of samples $(\mathbf{A}^t, \mathbf{B}^t = \mathbf{A}^t\mathbf{S} + \mathbf{E} \bmod q)$ from the LWE distribution $A_{\mathbf{S},\chi}$ (for the same $\mathbf{S} \leftarrow \chi^{n\times n}$) or from the uniform distribution, and $\mathcal{S}$ emulates either game H_1 or H_2 (respectively) to an attacker. Under the decision-LWE assumption, this will establish the claim. The simulator $\mathcal{S}$ answers queries $x = (x', x_i)$ where $|x'| = i - 1$ in the following way: if x' has never been queried before, then it draws a new sample $(\mathbf{A}^t_{x'}, \mathbf{B}^t_{x'})$ from $\mathcal{O}$ and stores it, otherwise it looks up the already stored $(\mathbf{A}^t_{x'}, \mathbf{B}^t_{x'})$. It then returns $\mathbf{A}^t_{x'}$ if $x_i = 0$, and $\mathbf{B}^t_{x'}$ if $x_i = 1$. It is clear by inspection and the definition of $\mathcal{H}^{(i)}$ that $\mathcal{S}$ has the claimed behavior given the two types of oracles $\mathcal{O}$.

By the triangle inequality, we have $H_0 \overset{c}{\approx} H_2$, i.e., $\tilde{\mathcal{G}}^{(i)}$ is a pseudorandom function family. □

We now analyze the ring-LWE construction.

Proof sketch for Theorem 5.3. The proof proceeds almost identically to the proof of Theorem 5.2, so we only outline the few small differences. We define the function families $\mathcal{RG}$ and $\mathcal{R}\tilde{\mathcal{G}}$ in exactly the same fashion as the families $\mathcal{G}$ and $\tilde{\mathcal{G}}$, respectively, with $a \in R_q$, $s_i \in R$ and $e_{x'} \in R$ substituting $\mathbf{A}$, $\mathbf{S}_i$ and $\mathbf{E}_{x'}$ respectively. In the games, the bad event BAD_x occurs if any *coefficient* of $R\tilde{G}(x) \in R_q$ (for $R\tilde{G} \leftarrow \mathcal{R}\tilde{\mathcal{G}}$) is "too close" to another element in $\mathbb{Z}_q$ having a different rounded value, where "too close" is defined using the interval $[-B, B]$ for $B = k(r\sqrt{n} \cdot \omega(\sqrt{\log n}))^k/\sqrt{n}$. For this bound B, the analogue of Lemma 5.5 (which bounds the cumulative error terms, i.e., the difference $R\tilde{G}(x) - RG(x)$) follows immediately from Lemma 2.3. Finally, pseudorandomness of the family $\mathcal{R}\tilde{\mathcal{G}}$ follows analogously to the proof of Theorem 5.6, via families $\mathcal{RH}^{(i)}$ defined similarly to $\mathcal{H}^{(i)}$. □

Remark 5.8. By almost identical proofs, a similar subset-product-like construction

$$F_{\mathbf{A},\{\mathbf{S}_{i,b}\}}(x_1 \cdots x_k) = \left\lfloor \mathbf{A}^t \cdot \prod_{i=1}^{k} \mathbf{S}_{i,x_i} \right\rceil_p, \tag{5.3}$$

for uniform $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and matrices $\mathbf{S}_{i,b} \in \mathbb{Z}^{n\times n}$ (for $i \in [k], b \in \{0,1\}$), and the analogous function in the ring setting, are also PRF families for the same parameters and distributions as in Theorem 5.2 and Theorem 5.3. (These functions are analogous to the factoring-based PRF of [NRR00].) While the secret keys are about twice as large as their counterparts' from Definition 5.1, these functions are more "symmetric," which may be important in practice (e.g., to prevent timing attacks).

5.4 Security Proof Under Interactive LWR

We now present an "interactive" LWR assumption and prove that under this assumption, the degree-k construction from Definition 5.1 is a PRF under an appropriate distribution of the $\mathbf{S}_i$. The advantage of this proof is that it allows us to prove security for a small modulus q and inverse error rate (both small polynomials in n), and it also works for uniformly random (or uniform invertible) matrices $\mathbf{S}_i$, among other distributions. For example, this allows us to compose the degree-k construction with itself (or with any other PRF) in a k-ary tree. The drawback to our proof is that it relies on a stronger assumption that is harder to evaluate or falsify, because it allows the adversary to make queries to its challenger.

Definition 5.9 (k-subset-product LWR)**.** Let $q \geq p$ be integer moduli. We describe a pair of games, which are parameterized by integers $k \geq 1$ and $m = \mathrm{poly}(n)$, and a distribution ψ over $\mathbb{Z}_q^{n \times n}$ (e.g., the uniform distribution). In both games, we choose $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ uniformly at random and $\mathbf{S}_i \leftarrow \psi$ independently for each $i \in [k]$, then give $\mathbf{A}$ and $\mathbf{S}_i$ for $i \in [k-1]$ to the attacker. We then allow the attacker to adaptively make queries to a function $H\colon \{0,1\}^{k-1} \to \mathbb{Z}_p^{m \times n}$. In the first game, the function H is defined to be

$$H(x) := \left\lfloor \mathbf{A}^t \cdot \prod_{i=1}^{k-1} \mathbf{S}_i^{x_i} \cdot \mathbf{S}_k \right\rceil_p ;$$

in the second game, H is a uniformly random function. The *k-subset-product* LWR *problem*, denoted k-LWR$_{q,p,m,\psi}$, is to distinguish between these two games with an advantage non-negligible in n. The k-subset-product ring-LWR problem is defined analogously. (A subset-product version of (ring-)LWE is also easy to formulate, where instead of rounding we add random and independent error terms to each answer.)

We make a few simple observations about the k-LWR problem. First note that $\mathbf{B}_x^t = \mathbf{A}^t \cdot \prod_{i=1}^{k-1} \mathbf{S}_i^{x_i}$ is the part of the product that changes for each new query. Since $\mathbf{A}$ and all the $\mathbf{S}_i$ for $i \in [k-1]$ are given to the attacker, it can compute each $\mathbf{B}_x^t$ on its own, and its goal is to determine whether the challenger is returning rounded products $\lfloor \mathbf{B}_x^t \cdot \mathbf{S}_k \rceil_p$ or uniformly random and independent values. In effect, the k-LWR problem is therefore to solve LWR when the sampled $\mathbf{A}$ matrices are related by adversarially chosen subset-products of given random matrices $\mathbf{S}_i$. To avoid an efficient attack (as outlined in the introduction), the distribution ψ should be chosen so that the product of many $\mathbf{S}_i \leftarrow \psi$ does not significantly reduce the entropy of $\mathbf{A}^t \prod_i \mathbf{S}_i$. It appears that restricting ψ to invertible elements is most effective for this purpose.

We also observe that 1-LWR$_{q,p,m,\psi}$ is just the standard LWR$_{q,p}$ problem given m samples, where the secret matrix $\mathbf{S}$ is chosen from ψ. The problems form a hierarchy over k, that is, k-LWR$_{q,p,m,\psi}$ no harder than $(k-1)$-LWR$_{q,p,m,\psi}$, by a reduction that just prepends 0 to all queries, and withholds $\mathbf{S}_1$ from the attacker.

Theorem 5.10. *Endow the family $\mathcal{F}$ from Definition 5.2 with the distribution where each $\mathbf{S}_i$ is drawn from some distribution ψ. Then, assuming that k-LWR$_{q,p,m,\psi}$ problem is hard, the family $\mathcal{F}$ is pseudorandom.*

Unlike our inductive proof of Theorem 5.6, which transitions from the PRF family to a random function by "dropping" the secret key components $\mathbf{S}_i$ from $i = 1$ to k, the proof of Theorem 5.10 drops them from $i = k$ down to 1. This prevents the error terms from growing with k (because the errors are not compounded by multiplication with other $\mathbf{S}_i$), which is what allows us to use a small modulus q if we so desire. However, this style of proof also seems to require an interactive assumption, so that a simulator can answer queries involving the component $\mathbf{S}_i$ that is being dropped between adjacent games.

Proof of Theorem 5.10. We prove this by induction over k. For $k = 0$, the claim follows trivially by construction. For $k \geq 1$, we again proceed via a series of games.

Game H_0. This is the real PRF attack game against the family $\mathcal{F}$: we choose an $F \leftarrow \mathcal{F}$, and the attacker has oracle access to $F(\cdot)$.

Game H_1. We choose $F \leftarrow \mathcal{F}$. For attacker queries of the form $x = x_1 \ldots x_{k-1}1$, we return uniformly random and independent value (consistent with prior answers), and for queries of the form $x = x_1 \ldots x_{k-1}0$, we return $F(x) = \left\lfloor \mathbf{A}^t \cdot \prod_{i=1}^{k-1} \mathbf{S}_i^{x_i} \right\rceil_p$.

We claim that $H_0 \overset{c}{\approx} H_1$ by a straightforward reduction assuming the hardness of k-LWR. As proof, we construction a simulator $\mathcal{S}$ that interacts with an oracle $\mathcal{O}$ that implements one of the two games from the k-LWR problem, and emulates either H_0 or H_1 respectively. The simulator is first given some matrices $\mathbf{A}$ and $\mathbf{S}_i$ for $i \in [k-1]$. It then answers attacker queries $x = (x', 0) \in \{0,1\}^k$ by returning $\lfloor \mathbf{A}^t \cdot \prod_{i=1}^{k-1} \mathbf{S}_i^{x_i} \rceil_p$, and answers queries $x = (x', 1) \in \{0,1\}^k$ by returning $\mathcal{O}(x')$ to the attacker. It is clear by inspection that the behavior of $\mathcal{S}$ is as claimed.

Game H_2. We lazily choose a uniformly random function $U \colon \{0,1\}^k \to \mathbb{Z}_p^{m \times n}$ and give the attacker oracle access to $U(\cdot)$.

We claim that $H_1 \overset{c}{\approx} H_2$ by the inductive hypothesis. This is because in game H_1, queries ending in 1 are already answered uniformly, while queries ending in 0 are answered according to a function drawn from the family $\mathcal{F}$ of degree $(k-1)$. This family is pseudorandom by the inductive hypothesis, and the fact that $(k-1)$-LWR is no easier than k-LWR.

This completes the induction and the proof. □

Acknowledgments. We thank Oded Regev for interesting discussions.

References

[ABB10a] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In *EUROCRYPT*, pages 553–572. 2010.

[ABB10b] S. Agrawal, D. Boneh, and X. Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In *CRYPTO*, pages 98–115. 2010.

[ACPS09] B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *CRYPTO*, pages 595–618. 2009.

[AG11] S. Arora and R. Ge. New algorithms for learning in presence of errors. In *ICALP (1)*, pages 403–415. 2011.

[Ajt96] M. Ajtai. Generating hard instances of lattice problems. *Quaderni di Matematica*, 13:1–32, 2004. Preliminary version in STOC 1996.

[Ban95] W. Banaszczyk. Inequalites for convex bodies and polar reciprocal lattices in R^n. *Discrete & Computational Geometry*, 13:217–231, 1995.

[BGV11] Z. Brakerski, C. Gentry, and V. Vaikuntanathan. Fully homomorphic encryption without bootstrapping. Cryptology ePrint Archive, Report 2011/277, 2011. `http://eprint.iacr.org/`.

[BKW03] A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM*, 50(4):506–519, 2003.

[BMR10] D. Boneh, H. W. Montgomery, and A. Raghunathan. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In *ACM Conference on Computer and Communications Security*, pages 131–140. 2010.

[BV11a] Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *FOCS*. 2011. To appear.

[BV11b] Z. Brakerski and V. Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *CRYPTO*, pages 505–524. 2011.

[CHKP10] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, pages 523–552. 2010.

[Gen09] C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178. 2009.

[GGM84] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986. Preliminary version in FOCS 1984.

[GKPV10] S. Goldwasser, Y. T. Kalai, C. Peikert, and V. Vaikuntanathan. Robustness of the learning with errors assumption. In *ICS*, pages 230–240. 2010.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206. 2008.

[HB01] N. J. Hopper and M. Blum. Secure human identification protocols. In *ASIACRYPT*, pages 52–66. 2001.

[JW05] A. Juels and S. A. Weis. Authenticating pervasive devices with human protocols. In *CRYPTO*, pages 293–308. 2005.

[KPC+11] E. Kiltz, K. Pietrzak, D. Cash, A. Jain, and D. Venturi. Efficient authentication from hard learning problems. In *EUROCRYPT*, pages 7–26. 2011.

[KSS06] J. Katz, J. S. Shin, and A. Smith. Parallel and concurrent security of the HB and HB$^+$ protocols. *J. Cryptology*, 23(3):402–421, 2010. Preliminary version in Eurocrypt 2006.

[LLL82] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, December 1982.

[LMPR08] V. Lyubashevsky, D. Micciancio, C. Peikert, and A. Rosen. SWIFFT: A modest proposal for FFT hashing. In *FSE*, pages 54–72. 2008.

[LPR10] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In *EUROCRYPT*, pages 1–23. 2010.

[LW09] A. B. Lewko and B. Waters. Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In *ACM Conference on Computer and Communications Security*, pages 112–120. 2009.

[MM11] D. Micciancio and P. Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In *CRYPTO*, pages 465–484. 2011.

[MP00] C. Mereghetti and B. Palano. Threshold circuits for iterated matrix product and powering. *ITA*, 34(1):39–46, 2000.

[MP11] D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller, 2011. Manuscript.

[NR95] M. Naor and O. Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. *J. Comput. Syst. Sci.*, 58(2):336–375, 1999. Preliminary version in FOCS 1995.

[NR97] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004. Preliminary version in FOCS 1997.

[NRR00] M. Naor, O. Reingold, and A. Rosen. Pseudorandom functions and factoring. *SIAM J. Comput.*, 31(5):1383–1404, 2002. Preliminary version in STOC 2000.

[Pei09] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *STOC*, pages 333–342. 2009.

[Pei10] C. Peikert. An efficient and parallel Gaussian sampler for lattices. In *CRYPTO*, pages 80–97. 2010.

[Pie10] K. Pietrzak. Subspace LWE, 2010. Manuscript. Last retrieved from `http://homepages.cwi.nl/~pietrzak/publications/SLWE.pdf` on 28 June 2011.

[PW08] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In *STOC*, pages 187–196. 2008.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):1–40, 2009. Preliminary version in STOC 2005.

[RT92] J. H. Reif and S. R. Tate. On threshold circuits and polynomial computation. *SIAM J. Comput.*, 21(5):896–908, 1992.

[Sch87] C.-P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theor. Comput. Sci.*, 53:201–224, 1987.

[Ver11] R. Vershynin. Introduction to the non-asymptotic analysis of random matrices, January 2011. Available at `http://www-personal.umich.edu/~romanv/papers/non-asymptotic-rmt-plain.pdf`, last accessed 4 Feb 2011.

[vGHV10] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 24–43. 2010.