Scalable, quantum-accessible, and adaptive pseudorandom quantum state and pseudorandom function-like quantum state generators

Rishabh Batra[1][0000-0002-5829-2304], Zhili Chen[1][0009-0002-2155-5817], Rahul Jain[1,2,3][0000-0002-3649-6576], and YaoNan Zhang[1][0009-0009-3760-9729]

[1] Centre for Quantum Technologies, Singapore
{rishabhbatra,chen.zhili,zhang.yaonan}@u.nus.edu
[2] Department of Computer Science, National University of Singapore
rahul@comp.nus.edu.sg
[3] MajuLab, UMI 3654, Singapore

Abstract. We show new constructions for pseudorandom quantum states (PRS) and pseudorandom function-like quantum state (PRFS) generators satisfying **scalability**, which means the security parameter can be much larger than the number of qubits, **quantum accessibility**, which means the adversary can provide quantum input, and **adaptivity**, which means the adversary can query it adaptively.

We present an isometric procedure to prepare quantum states that can be arbitrarily random (i.e., the trace distance from the Haar-random state can be arbitrarily small for the true random case, or the distinguishing advantage can be arbitrarily small for the pseudorandom case). This naturally gives the first construction for scalable, quantum-accessible, and adaptive PRFS assuming quantum-secure one-way functions. Compared to prior PRFS works, we use a stronger definition of quantum accessibility in which the adversary can be ancilla-assisted, i.e., the input state may not be pure and could be entangled with other quantum registers. Thus, our result also gives the first (fully) quantum-accessible PRFS.

Our PRFS construction implies various primitives, including long-input PRFS, short-input PRFS, short-output PRFS, non-adaptive PRFS, and classically-accessible adaptive PRFS [AQY22,AGQY22]. This new construction may be helpful in simplifying the microcrypt zoo.

Keywords: Quantum cryptography · Pseudorandom quantum state.

1 Introduction . . . 2
2 Technical overview . . . 7
2.1 Random amplitudes quantum state . . . 7
2.2 (Pseudo)random quantum state . . . 8
2.3 (Pseudo)random function-like quantum state generator . . . 9
3 Preliminaries . . . 10
3.1 Notations . . . 10
3.2 Probability distributions . . . 11
3.3 Quantum cryptography . . . 12
4 Isometric asymptotically random state generator . . . 16
4.1 Random amplitudes procedure . . . 16
4.2 Random phases procedure . . . 18
4.3 Asymptotically random states . . . 19
5 Pseudorandom quantum state generators . . . 20
6 (Pseudo)random function-like quantum state generators . . . 20
A Alternative proof of ARS . . . 28
B Proof of asymptotically random to pseudorandom . . . 32
C Proof of the random amplitudes quantum state from Beta variables . . . 33
D Proof of that random phase gives standard Gaussian . . . 34
E Proof of the diamond-norm bound via operator norm . . . 34
F Proof of claim about the scalable pseudorandom state generator . . . 35
G Classical sampling algorithms . . . 36
H Technical lemmas on function properties . . . 47

1 Introduction

Randomness has played an indispensable role in various fields of computer science including cryptography, algorithms, and complexity theory. For practical applications, however, using true randomness may be very expensive and difficult to produce. Storing the complete description of a truly random object may be very inefficient. For example, to merely specify a binary truly random function on n-bit inputs, we require an exponentially large description.

An object is called pseudorandom if we can generate it efficiently, but an efficient distinguisher cannot distinguish it from a truly random object. Using pseudorandomness enables us to design efficient protocols that require only short random seeds. In classical cryptography, pseudorandomness has been fundamental for the construction of efficient protocols that are secure against attacks by efficient adversaries.

Let U_n represent the uniform distribution on n bits. A pseudorandom generator (PRG) is an efficiently computable function, $\mathrm{PRG} : \{0,1\}^n \to \{0,1\}^{l(n)}$ such that $l(n) > n$ and $\mathrm{PRG}(U_n)$ is computationally indistinguishable from $U_{l(n)}$. A one-way function (OWF) is a function that is efficiently computable and hard to invert. The existence of one-way functions is a widely used computational assumption in classical cryptography. A pseudorandom function family (PRF) is a collection of efficiently computable functions that are computationally indistinguishable from a truly random function. It is known that in the classical world, OWF, PRG, and PRF are equivalent to each other [Gol01].

Motivations

In the quantum world, however, the story is not as straightforward. Ji, Liu, and Song [JLS18] first introduced the notion of pseudorandom quantum states (PRS), which are efficiently generatable pure quantum states that are computationally indistinguishable from truly random (Haar-random) quantum states — even when

an adversary has access to multiple copies of the state. PRS can be thought of as a generalization of PRGs with quantum outputs. Similarly, pseudorandom function-like quantum state (PRFS) generators, first introduced by Ananth, Qian, and Yuen [AQY22], are the quantum analogue of PRF.

Morimae and Yamakawa [MY22] showed that quantum commitments can be based on PRS (or PRFS). It is known that PRS (and PRFS) can exist even if BQP = QMA (relative to a quantum oracle) [Kre21] or if P = NP (relative to a classical oracle) [KQST23], which does not allow for the existence of one-way functions (relative to these oracles). Hence, these are potentially weaker objects than OWF, from which we can construct quantum commitments, oblivious transfer, and multi-party computation [BCQ23,GLSV21,BCKM21,AQY22]. Therefore, there is a natural interest in novel and efficient constructions of PRS and PRFS.

The following are some desirable properties of the PRS and PRFS constructions:

- **Scalability:** A PRS (or PRFS) generator is *scalable* if the security parameter λ and the output size n (and the input size m for PRFS) can be chosen independently. In other words, the distinguishing advantage can be arbitrarily small as long as we choose a sufficiently large λ, regardless of the state size. Scalability is not automatic in the quantum setting. Classically, an n-bit (pseudo)random string allows the use of any k-bit substring as a k-bit (pseudo)random string with $k < n$. However, in the quantum case, even if we already have an n-qubit Haar-random state, the first k qubits (while tracing out the rest) are typically very different from a k-qubit Haar-random state. Therefore, knowing how to create a longer pseudorandom object does not automatically provide a shorter one. Bouaziz-Ermann and Muguruza [BEM24] show that quantum pseudorandomness cannot be shrunk in a black-box way.
- **Adaptivity:** For PRFS, it is desirable to have *adaptive* security, i.e., the adversary can query it adaptively. The first PRFS construction [AQY22] guaranteed only selective (non-adaptive) security.
- **Quantum accessible/superposition query:** A PRFS is quantum accessible if the adversary can provide quantum (superposition) input (potentially entangled with ancilla registers) into it. Ideally, a PRFS query should keep any global pure state (including in ancilla registers) a pure state; that is, there is no loss of information on performing the query. Such a PRFS generator would be an *isometry*.

Remark 1 (Isometry vs quantum-accessibility). We would like to clarify that the quantum accessible property is the same as the isometric property for PRFS. That is, ideal map W should be the isometry as mentioned in our Definition 18.

$$\sum \alpha_x |x\rangle \mapsto \sum \alpha_x |x\rangle |\psi_x\rangle .$$

If it is not, then some input pure superposition query state would become a mixed state after the action of the ideal map. This would lose (to an extent) the (distinguishing) advantage that the adversary wanted to derive using the pure superposition query.

As such, it is not needed that the efficient construction V should be an isometry; it just needs to be indistinguishable from W by a QPT adversary. However, V being an isometry may have additional advantages of its own. For example, in our construction of PRS, V is an isometry as opposed to the construction by [BS20]. This difference crucially allows us to extend it to a PRFS construction in a natural fashion by using separate randomness for each input x.

Our contribution

In this work, we present an isometric and easy-to-implement procedure to prepare quantum states that can be arbitrarily random (i.e., the trace distance from the Haar-random state can be arbitrarily small for the true random case, or the distinguishing advantage can be arbitrarily small for the pseudorandom case). This procedure provides **a new method for scalable and isometric PRS**.

Theorem 1 (Informal). *Assuming a quantum-secure PRF, there exists a scalable and isometric PRS.*

Using independent randomness for different inputs in our isometric state-generation procedure yields **the first construction for scalable, quantum-accessible, and adaptive PRFS**.

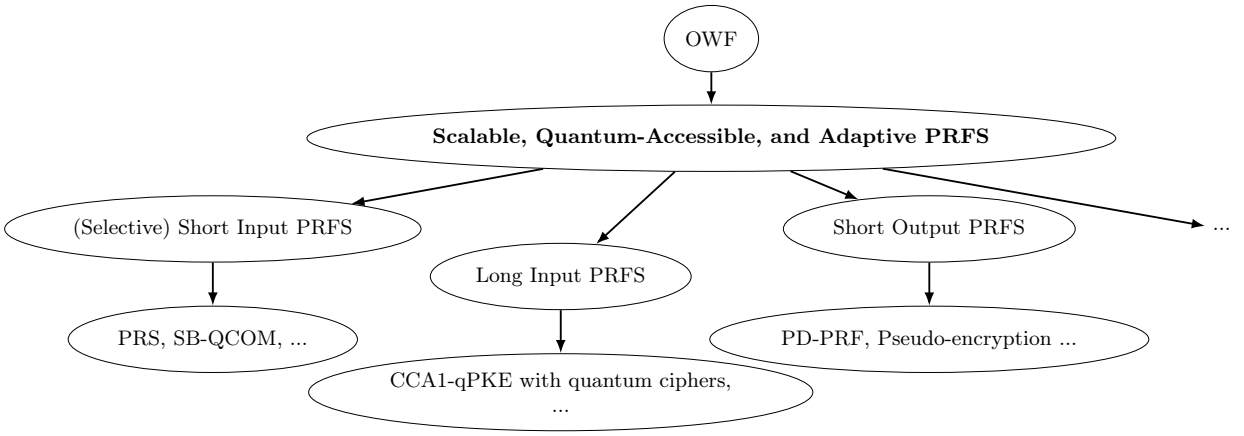


Fig. 1. Our PRFS construction implies various other primitives.

Theorem 2 (Informal). *Assuming a quantum-secure PRF, there exists a scalable, quantum accessible, and adaptive PRFS.*

Combining with the fact that quantum-secure OWF implies quantum-secure PRF [Zha21], our work shows that quantum-secure OWF implies scalable, quantum-accessible, and adaptive PRFS. Our PRFS construction implies various primitives, including long-input PRFS, short-input PRFS, short-output PRFS, non-adaptive PRFS, and classical-accessible adaptive PRFS [AQY22,AGQY22]. This gives a unified construction of PRFS, and may be helpful in simplifying the microcrypt zoo [4].

Since short-input PRFS implies SB-QCOM and quantum pseudo-encryption [AGQY22], long-input PRFS implies CCA1-qPKE with quantum ciphers [BGHD$^+$23], short-output PRFS implies PD-PRF [ALY23], and so on, our new construction can directly imply all these primitives. If a black-box separation between OWF and scalable PRFS can be shown in the future, this would suggest a potential cryptography world without OWFs, which encompasses a richer set of primitives than one starting from PRS.

Previous works and our novelty

1. **Novelty over Ji, et al. [JLS18] and Brakerski and Shmueli [BS19] (PRS)**: PRS was first introduced by [JLS18], and they provide the first construction by applying a random phase to each computational basis element:

$$|\phi_k\rangle = \frac{1}{\sqrt{N}} \sum_{z=0}^{N-1} \left(e^{2\pi i/N}\right)^{f_k(z)} |z\rangle \,.$$

   Later, [BS19] proved that the binary phase state is also pseudorandom:

$$|\phi_k\rangle = \frac{1}{\sqrt{N}} \sum_{z=0}^{N-1} (-1)^{f_k(z)} |z\rangle \,.$$

   These elegant constructions guarantee that t-copies of such a state are close to t-copies of the Haar-random state for polynomially bounded t when the number of qubits is sufficiently large. However, since these constructions are actually very different from the true Haar-random state, they are not scalable: There exists a size-dependent distance (distinguishing advantage) that cannot be reduced. Thus, the size n is required to be in $\Omega(\log \lambda)$ for security parameter λ. Our construction, on the other hand, is scalable and has no such restriction.

[4] https://sattath.github.io/microcrypt-zoo/.

2. **Novelty over Brakerski and Shmueli [BS20] (scalable PRS)**: The first scalable PRS generator construction is given by [BS20]. They prepare a state whose coefficients follow a normalized Gaussian random vector by *quantum rejection sampling*. This approach prevents them from obtaining an isometric PRS. Their construction does not give a pure state output and has some entangled junk that needs to be traced out, and hence, it is not clear how to extend such a PRS to the construction of a quantum-accessible PRFS.

   Our construction is also based on this well-known fact that the distribution of each coefficient in a Haar-random state follows a complex Gaussian distribution (up to normalization), but we use a very different approach. Our key step is similar to the Grover-Rudolph algorithm [GR02]: We run a Beta sampling algorithm in superposition and apply controlled rotations qubit by qubit. Then classical results relating Beta distribution, Gamma distribution, Chi-squared distribution, and Gaussian distribution give us the desired state. This gives us an isometric construction, and naturally leads to our PRFS construction.
3. **Novelty over Ananth et al. [AQY22,AGQY22] (PRFS)**: The initial PRFS construction introduced by [AQY22] guaranteed only selective security. Subsequently, Ananth, Gulati, Qian, and Yuen [AGQY22] gave new constructions for classically-accessible and quantum-accessible adaptive PRFS (secure against pure input), which are also not scalable. We present the first PRFS construction which is simultaneously quantum-accessible, adaptive, and scalable.
4. **Novelty over Lu et al. [LQS$^+$24] (PRSS)**: A new primitive called quantum pseudorandom scramblers (PRSS) was introduced by [LQS$^+$24], which also provides a scalable and isometric PRS generation procedure. In their Appendix, they show that PRSS can imply classically-accessible selective PRFS (without invoking OWF). They also claim adaptive and quantum-accessible security for logarithmic-size inputs, using the fact that superposition queries provide no additional advantages when the output state is known for every input string.

   Our focus differs: They show that their new primitive implies (limited) PRFS in a black-box way. We directly construct PRFS that satisfies all the desired properties from a quantum-secure PRF. The resulting scalable, quantum-accessible, and adaptive PRFS implies all prior limited PRFS primitives. Using independent randomness for different inputs is natural, but security against quantum queries is not automatic. We explicitly allow the adversary to use quantum ancilla registers (see more details below). We bound the *diamond-norm distance* via the operator-norm distance between the corresponding isometries, without a dimension-dependent loss in the input size.

   Subsequent to our work appearing on ArXiv, in a personal communication from the authors of [LQS$^+$24], it was brought to our notice that, similar to the construction in this work, assuming in addition a quantum-secure PRF, one can sample a key for each x by the PRF and obtain a scalable PRFS for arbitrarily long inputs. This construction is secure against adversaries using classical ancillary bits via straightforward triangle inequalities. However, it is currently unclear whether the construction remains secure when the ancillary qubits may be entangled with the input register. [5].
5. **Novelty over all of the above:** For a quantum-accessible pseudorandom object, we naturally expect the adversary to be ancilla-assisted, i.e., the input may not be pure, but entangled with some other register held by the adversary. The recent constructions of pseudorandom unitaries (PRUs) by Ma and Huang [MH25] indeed allow the adversary to use ancilla registers. No other work allows this in PRFS to the best of our knowledge. We give **the first (fully) quantum-accessible PRFS**.

   Since scalable PRU is still open, to the best of our knowledge, our result is the first quantum pseudorandom object that is scalable and secure against ancilla-assisted quantum adversaries.
6. **Difference with PRU:** PRU is only known to imply a PRFS with $m \leq n$. In the classical world, given a PRF ensemble $\{f_k : \{0,1\}^{l(\lambda)} \to \{0,1\}^{r(\lambda)}\}$, one can straightforwardly derive another PRF ensemble with different input/output lengths $\{f_k : \{0,1\}^{l'(\lambda)} \to \{0,1\}^{r'(\lambda)}\}$. Consequently, we typically do not distinguish between PRF and PRF with specific input/output lengths. However, this is not true for PRFS.

   Non-adaptive PRU was given by Metger et al. [MPSY24], and adaptive PRU by [MH25]. However, constructing a scalable PRU remains an open question. Since gluing small PRUs has recently become a

[5] Chuhan Lu, Minglong Qin, Fang Song, Penghui Yao, Mingnan Zhao. Personal communication.

popular technique for constructing low-depth PRU [SHH25,FPVY26], scalability appears to be a crucial property to improve the result [6].
Furthermore, even given a scalable PRU, it is unclear how to derive a scalable PRFS. This is similar to the classical case: A PRP over a small domain can be distinguished from PRF.

7. **Difference with Grover-Rudolph [GR02]:** To prepare the target state, we develop a new technique inspired by the Grover-Rudolph algorithm, yet distinct in its approach. In the Grover-Rudolph algorithm, if a quantum state with given coefficients is to be created, qubit-by-qubit rotations are done by computing the ratio between the sum of the coefficients. In our case, we are creating a random quantum state with normalized Gaussian vector coefficients by performing qubit-by-qubit rotations using Beta sampling algorithms, feeding prefixes in superposition as randomness seeds.
This provides a general framework for efficiently sampling a random quantum state, as long as the distribution of the ratio between partial coefficient sums is independent and efficiently samplable. This may prove useful for other applications in quantum cryptography and algorithms.

In Table 1 and Table 2, we compare our work with prior PRS and PRFS constructions.

Table 1. Comparison with prior PRS constructions.

Work by	PRS Construction	Scalable	Isometry
[JLS18]	Random phase	No	Yes
[BS19]	Binary phase	No	Yes
[BS20]	Quantum rejection sampling	Yes	No
[LQS$^+$24]	Kac's walk	Yes	Yes
This work	Qubit-by-qubit rotation	Yes	Yes

Table 2. Comparison with prior PRFS constructions.

Work by	Scalable	Adaptive	Quantum-accessible	Ancilla-assisted adversary
[AQY22]	No	No	No	Not applicable
[AGQY22]	No	Yes	Yes	Unknown
[LQS$^+$24](Lemma 16)	Yes	No	No	Not applicable
[LQS$^+$24](Lemma 17)	No	Yes	Yes	Unknown
This work	Yes	Yes	Yes	Yes

Future work

Using ideas similar to our construction of PRS and PRFS, it may be possible to construct more scalable quantum pseudorandom primitives, such as pseudorandom isometries and pseudorandom unitaries, which is a major open question.

As shown by Ananth et al. [AGQY22], using the technique of "verifiable tomography", they are able to obtain a bit-commitment protocol with classical communication from a scalable PRS. For this application, the scalability of the PRS is important. It is conceivable that, using similar arguments, the technique of verifiable tomography may help in getting classical communication analogues of some other primitives, e.g., MAC with quantum tags, qPKE with quantum ciphers, etc., from our scalable PRFS. These are currently obtained from long-input PRFS.

[6] Currently known constructions either require the size of each small PRU to be $\omega(\log n)$, or use log-size small PRUs but achieve only inverse-polynomial security.

Another interesting topic is the relationship between OWF and scalable PRFS (see Figure 1). As mentioned before, if a black-box separation can be shown, this would suggest a potential cryptography world without OWF, which encompasses a richer set of primitives than one starting from PRS. On the other hand, if we can show that OWF and scalable PRFS are equivalent, this would significantly reshape our understanding of quantum cryptographic primitives.

2 Technical overview

An asymptotically random state (ARS) generator is a statistical notion of PRS that generates states that are statistically indistinguishable from Haar-random states but requires (quantum) access to an exponential amount of randomness. Then, replacing the exponentially large random string with a quantum-secure pseudorandom function (PRF), a PRS construction naturally follows from an ARS construction.

We now provide an outline of how we achieve a scalable ARS using a $\mathbb{C} \to \mathbb{C}^{2^n}$ isometric procedure, given quantum access to a random function. The output state is exactly the Haar-random state in the idealized setting with infinite precision and a perfect sampling algorithm, and has negligible error with finite precision. We use a similar technique to construct scalable and adaptive asymptotically random function-like quantum state (ARFS) generators and PRFS generators.

2.1 Random amplitudes quantum state

From Fact 12 we know that for a Haar-random state, its coefficients follow the distribution of the normalized standard Gaussian random vector (Definition 11). We want to construct a state

$$|\phi\rangle = \frac{1}{\sqrt{\sum_{z\in[N]} A_z^2 + B_z^2}} \sum_{z\in[N]} (A_z + iB_z)\,|z\rangle\,,$$

where $N = 2^n$ the size of Hilbert space, and $A_z, B_z \sim \mathcal{N}(0,1)$ i.i.d. To do this, we first consider the amplitudes $\sqrt{C_z} = \sqrt{A_z^2 + B_z^2}$ and create the following state (we call it the random amplitudes quantum state):

$$|\xi\rangle = \frac{1}{\sqrt{\sum_{z\in[N]} C_z}} \sum_{z\in[N]} \sqrt{C_z}\,|z\rangle\,.$$

Each C_z follows the χ_2^2 distribution if $A_z, B_z \sim \mathcal{N}(0,1)$. Once we can prepare such a state, we can obtain the Haar-random state by applying a random phase $e^{2\pi i U_z}$, where $U_z \sim \mathcal{U}(0,1)$, to each $|z\rangle$ (Lemma 2):

$$|\phi\rangle = \frac{1}{\sqrt{\sum_{z\in[N]} C_z}} \sum_{z\in[N]} \sqrt{C_z} e^{2\pi i U_z}\,|z\rangle\,.$$

Merging all the $|z\rangle$ starting with 0 and all the $|z\rangle$ starting with 1, respectively, we can represent $|\xi\rangle$ as

$$|\xi\rangle = \frac{1}{\sqrt{W_0 + W_1}}(\sqrt{W_0}\,|0\rangle\,|\xi_0\rangle + \sqrt{W_1}\,|1\rangle\,|\xi_1\rangle),$$

where $W_0 = \sum_{z:z[0]=0} C_z$ and $W_1 = \sum_{z:z[0]=1} C_z$. If each $C_i \sim \chi_2^2$, then $W_0, W_1 \sim \chi_{2^n}^2$. This shows a very straightforward way to make the "weight" of the first qubit correct: Sample w_0, w_1 from $\chi_{2^n}^2$, and rotate the first qubit:

$$|0\rangle \to \frac{1}{\sqrt{w_0 + w_1}}(\sqrt{w_0}\,|0\rangle + \sqrt{w_1}\,|1\rangle).$$

We can continue to make the "weight" of the first two qubits correct. Let $W_{ab} = \sum_{z:z[0,1]=ab} C_z$ for $a, b \in \{0,1\}$. Similarly, we represent $|\xi_0\rangle$ as

$$|\xi_0\rangle = \frac{1}{\sqrt{W_{00} + W_{01}}}(\sqrt{W_{00}}\,|0\rangle\,|\xi_{00}\rangle + \sqrt{W_{01}}\,|1\rangle\,|\xi_{01}\rangle),$$

and represent $|\xi_1\rangle$ as

$$|\xi_1\rangle = \frac{1}{\sqrt{W_{10}+W_{11}}}(\sqrt{W_{10}}\,|0\rangle\,|\xi_{10}\rangle + \sqrt{W_{11}}\,|1\rangle\,|\xi_{11}\rangle).$$

Since each $C_z \sim \chi^2_2$, we have $W_{ab} \sim \chi^2_{2^{n-1}}$. We can sample $w_{00}, w_{01}, w_{10}, w_{11}$ from $\chi^2_{2^{n-1}}$ under the condition that $w_{00}+w_{01}=w_0$ and $w_{10}+w_{11}=w_1$, and rotate the second qubit controlled on the first qubit:

$$\begin{aligned}&\frac{1}{\sqrt{w_0+w_1}}(\sqrt{w_0}\,|0\rangle + \sqrt{w_1}\,|1\rangle)\,|0\rangle \\ &\to \frac{1}{\sqrt{w_{00}+w_{01}+w_{10}+w_{11}}}(\sqrt{w_{00}}\,|00\rangle + \sqrt{w_{01}}\,|01\rangle + \sqrt{w_{10}}\,|10\rangle + \sqrt{w_{11}}\,|11\rangle).\end{aligned}$$

Fact 4 tells us if $W_0, W_1 \sim \chi^2_{2^n}$ independently, then $W_0/(W_0+W_1) \sim \mathrm{Beta}(2^{n-1}, 2^{n-1})$ and is independent of W_0+W_1. This makes the above task easier. The overall process (for making the "weight" of the first two qubits correct) is as follows:

1. Initialize $|0\rangle^{\otimes n}$.
2. Sample $b \leftarrow \mathrm{Beta}(2^{n-1}, 2^{n-1})$ and rotate the first qubit:

$$|0\rangle \to \sqrt{b}\,|0\rangle + \sqrt{1-b}\,|1\rangle\,.$$

3. Sample $b_0, b_1 \leftarrow \mathrm{Beta}(2^{n-2}, 2^{n-2})$ and rotate the second qubit controlled on the first qubit:

$$|00\rangle \to |0\rangle\,(\sqrt{b_0}\,|0\rangle + \sqrt{1-b_0}\,|1\rangle),$$

$$|10\rangle \to |1\rangle\,(\sqrt{b_1}\,|0\rangle + \sqrt{1-b_1}\,|1\rangle).$$

Doing this bit by bit, we can obtain the random amplitudes (see Lemma 1 for details). If we directly apply the above procedure, we need to sample 2^t Beta random variables and do 2^t controlled rotations for the t-th qubit, but we can do this in superposition, which makes the procedure efficient. This is inspired by the Grover-Rudolph algorithm [GR02]. While the Grover-Rudolph algorithm uses each prefix to compute the conditional probability mass, we use the prefix as a random seed to sample an independent Beta random variable.

Assume that we have a deterministic classical Beta sampling algorithm $\mathcal{A}_B(k,z)$ which samples $b \leftarrow \mathrm{Beta}(2^k, 2^k)$ with random seed z (with finite precision). We use its reversible quantum version $U_B^{(k)} : |z\rangle\,|0\rangle \mapsto |z\rangle\,|b_z\rangle$ to compute an independent Beta random variable b_z (with finite precision) for each prefix z and store in an ancilla qubit, and then apply a rotation controlled on this ancilla qubit. These are described in Algorithm 1. We need two things for such $\mathcal{A}_B(k,z)$: A random function to convert the random seed z to randomness r_z, and a Beta sampling algorithm which uses randomness r_z.

In the Appendix, we show a classical sampling algorithm we used for the random amplitudes procedure. While this algorithm is well-known and widely used in practice (for example, in the NumPy library), we provide a rigorous proof of its correctness and show that the error is negligible when truncating to finite precision.

2.2 (Pseudo)random quantum state

Once we have the above random amplitudes state, denoted as $|\xi\rangle = \sum_{z=0}^{2^n-1} \alpha_z\,|z\rangle$, we can get an ARS by applying a random phase to each $|z\rangle$. We compute a uniformly random $u_z \in [2^\lambda]$ for each z on an ancilla

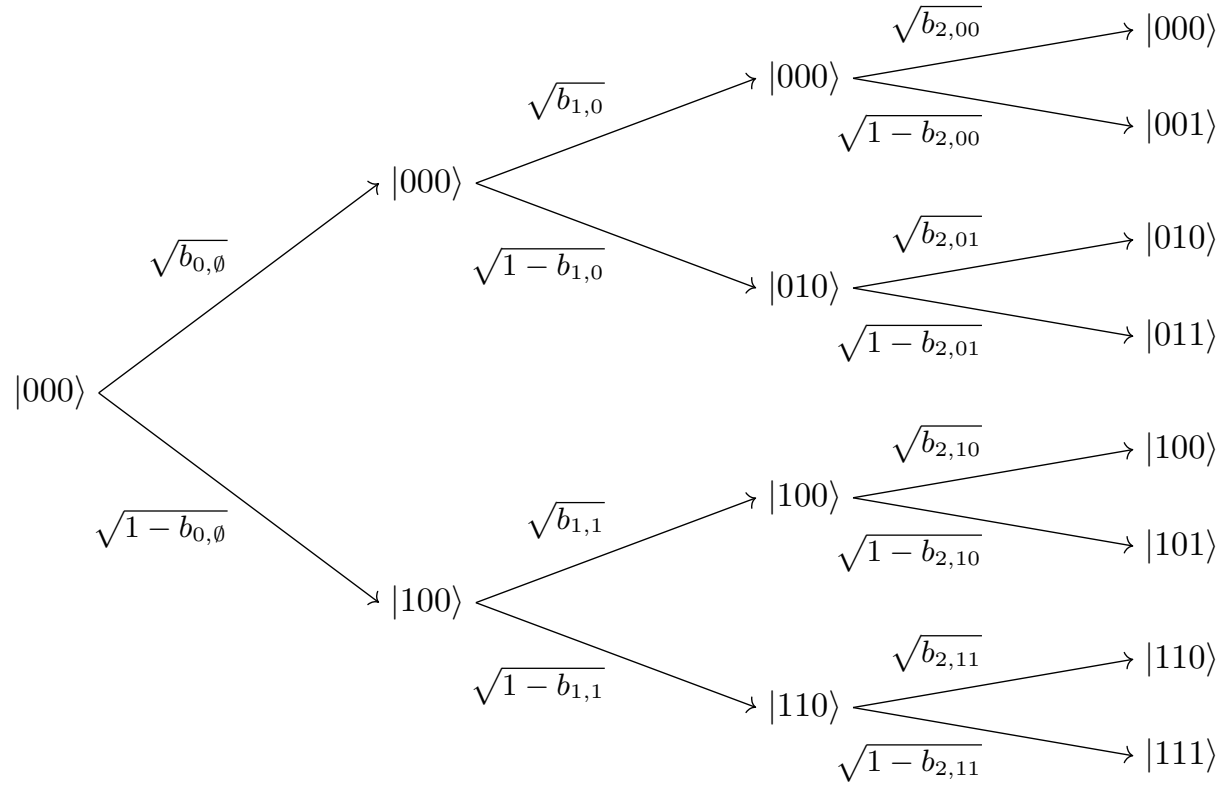


Fig. 2. Random amplitudes procedure for 3 qubits.

register, then apply the phase shift gate $P(2\pi/2^k)$ on k-th qubit of u_z for all k, and finally uncompute u_z.

$$\begin{aligned}
|\xi\rangle|0\rangle = \sum_{z=0}^{2^n-1} \alpha_z |z\rangle|0\rangle &\to \sum_{z=0}^{2^n-1} \alpha_z |z\rangle|u_z\rangle \\
&\to \sum_{z=0}^{2^n-1} \alpha_z |z\rangle \left(e^{2\pi i u_z/2^\lambda}|u_z\rangle\right) \\
&\to \sum_{z=0}^{2^n-1} \alpha_z e^{2\pi i u_z/2^\lambda}|z\rangle|0\rangle .
\end{aligned}$$

To compute these u_z's, we need quantum access to a random function.

In the idealized setting ($\lambda \to \infty$), the coefficient vector $(\alpha_z e^{2\pi i u_z/2^\lambda})_z$ is a normalized complex standard Gaussian random vector that corresponds to a Haar-random state. However, it requires considerable effort (deferred to the Appendix) to show that, with finite precision, the trace distance is negligible. Thus, for any adversary who queries this oracle polynomially many times, the distinguishing advantage is negligible (Theorem 4). Finally, by replacing the random function with a quantum-secure PRF, we construct a PRS (Theorem 5).

2.3 (Pseudo)random function-like quantum state generator

The next step is to modify the above ARS algorithm to accept an additional quantum state input $|\phi\rangle = \sum_x \alpha_x |x\rangle$ (More generally, the input could be $\rho = \mathrm{Tr}_E[\sum_x \alpha_x |x\rangle_X |\theta_x\rangle_E]$). Now, the random function f

takes not only the original input but also x as input. In other words, for classical input $|x\rangle$, $f(\cdot)$ is replaced by $f(x\|\cdot)$. Since the additional input state may be entangled with other registers, we need to bound the **diamond-norm distance**. We do this via the operator-norm distance between the corresponding isometries. Their block-diagonal structure makes this operator norm the maximum of the state-vector distances over x, so the bound has no dimension-dependent loss in the input size m. Let the output state of such a modified ARS procedure be $|\psi_x\rangle$ on input $|x\rangle$; then, for a superposition input $|\phi\rangle = \sum_x \alpha_x |x\rangle$, the entire output state becomes $\sum_x \alpha_x |x\rangle |\psi_x\rangle$.

Since we know that $f(x\|\cdot)$ for each x is also an independent random function, each $|\psi_x\rangle$ is an independent Haar-random state, in the idealized setting. For finite precision, we choose the rounded Beta sampler's output precision independently of its statistical accuracy, so the union bound over x affects only the latter and does not increase the state-generation precision parameter. This isometry $\sum_x \alpha_x |x\rangle \mapsto \sum_x \alpha_x |x\rangle |\psi_x\rangle$ is a truly random function-like quantum state generator. For finite precision and an imperfect sampling algorithm, the error in diamond norm distance is negligible, which bounds the maximal distinguishing probability of two quantum channels. Thus, for any adversary who queries this oracle polynomially many times, the distinguishing advantage is negligible (Theorem 6). Finally, by replacing the random function with a quantum-secure PRF, we construct a PRFS (Theorem 7).

Organization

In Section 3, we introduce the preliminaries and notations required for this work. In Section 4, we show an isometric algorithm whose outputs can be arbitrarily close to Haar-random states, given (quantum) access to a truly random function. In Section 5, we replace the truly random function with a quantum-secure pseudorandom function in the above algorithm, and achieve a scalable pseudorandom quantum state generator. In Section 6, we modify the above algorithm to accept an additional input state and achieve a scalable and adaptive (pseudo)random function-like quantum state generator. In the Appendix, we show a classical algorithm for sampling from the rounded Beta distribution (with negligible statistical distance), which is used in the above algorithm, and provide proofs of some lemmas and theorems.

3 Preliminaries

In this section, we introduce various notations, functions, and definitions we will require for our proofs.

3.1 Notations

Given $n \in \mathbb{N}^+$, we denote $\{0, 1, \ldots, n-1\}$ by $[n]$. For $x \in \{0,1\}^n$, we use $x[t]$ to denote the t-th bit of x, and use $x[1, \cdots, t]$ to denote the first t bits of x. Specifically, if $t = 0$, then $x[1, \cdots, t]$ is the empty string.

For a real number x, we use $\lfloor x \rfloor$ for rounding down; i.e., the largest integer that does not exceed x. We use $\lfloor x \rceil$ for rounding to the nearest integer (with rounding half up). For a complex number z, we denote the real part and the imaginary part of z by $\Re(z)$ and $\Im(z)$, respectively.

For a finite set $\mathcal{X}$, the notation $x \leftarrow \mathcal{X}$ signifies that the value x is sampled uniformly randomly from $\mathcal{X}$. Given a distribution D over $\mathcal{X}$, we use the notation $X \sim D$ to indicate that X is a random variable distributed according to D. In this paper, we always use uppercase letters for random variables and lowercase letters for their realizations/observed values, e.g., $X \sim D$ and $x \leftarrow D$. We use $\mathcal{U}(a, b)$ for the continuous uniform distribution on (a, b), and we use $X \sim \mathcal{U}_n$ to denote that X is a uniformly random variable over $\{0,1\}^n$. We usually use ":=" for assignment, but for clarity, we use "$\leftarrow$" when assigning the output of a probabilistic algorithm, e.g., $k \leftarrow \mathrm{KeyGen}(1^\lambda)$, as this is equivalent to sampling from the distribution of outputs.

A function $\varepsilon(\lambda)$ is negligible if for all constant $c > 0$, $\varepsilon(\lambda) < \lambda^{-c}$ for large enough λ. We denote the collection of polynomially bounded functions in λ by $\mathsf{poly}(\lambda)$ and denote the collection of negligible functions in λ by $\mathsf{negl}(\lambda)$. For $n(\lambda)$ a function of the security parameter λ, we may use the shorthand $n = n(\lambda)$ for brevity after it is defined.

Given sets $\mathcal{X}$ and $\mathcal{Y}$, denote $\mathcal{Y}^{\mathcal{X}}$ as the set of all functions $\{ f : \mathcal{X} \to \mathcal{Y} \}$.

Let $\mathcal{H}$ be a Hilbert space. We denote $S(\mathcal{H})$ as the set of unit vectors/pure states in $\mathcal{H}$. We use $L(\mathcal{H})$ to denote the set of linear operators from $\mathcal{H}$ to $\mathcal{H}$. We use $D(\mathcal{H})$ to denote the set of density operators acting on $\mathcal{H}$. Given an operator $P \in L(\mathcal{H})$, its trace norm is denoted by $\|P\|_1 \stackrel{\text{def}}{=} \operatorname{Tr}\left(\sqrt{P^\dagger P}\right)$. For a vector $\boldsymbol{v} \in \mathcal{H}$, we denote its Euclidean norm as $\|\boldsymbol{v}\|_2$. For a linear map $A : \mathcal{H}_A \to \mathcal{H}_B$, its operator norm is denoted by $\|A\|_{\text{op}} \stackrel{\text{def}}{=} \max_{\|\boldsymbol{v}\|_2=1} \|A\boldsymbol{v}\|_2$.

A linear map $V : \mathbb{C}^n \to \mathbb{C}^m$ is an *isometry* if $V^\dagger V = \mathrm{id}_n$. An isometry V is a *unitary* when $n = m$.

3.2 Probability distributions

Definition 1 (Statistical distance). *Given two distributions D_X and D_Y over a finite set S, denote random variables $X \sim D_X$ and $Y \sim D_Y$. The statistical distance of D_X and D_Y is defined as*

$$\Delta(D_X, D_Y) \stackrel{def}{=} \frac{1}{2} \sum_{s \in S} |\Pr[X = s] - \Pr[Y = s]| = \max_{T \subseteq S} |\Pr[X \in T] - \Pr[Y \in T]| .$$

Definition 2 (Normal distribution). *Given $\mu \in \mathbb{R}$ and $\sigma \in \mathbb{R}^+$, the normal (Gaussian) distribution $\mathcal{N}(\mu, \sigma^2)$ is a real-valued continuous probability distribution, with PDF*

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/(2\sigma^2)}, x \in \mathbb{R}.$$

The *standard normal distribution* is the special case $\mathcal{N}(0, 1)$.

Fact 1 (Box-Muller transform [BM58]). *Suppose $X \sim \mathcal{U}(0, 1)$ and $Y \sim \mathcal{U}(0, 1)$ independently. Let*

$$Z_0 = \sqrt{-2 \ln X} \cos(2\pi Y),$$

and

$$Z_1 = \sqrt{-2 \ln X} \sin(2\pi Y).$$

Then $Z_0 \sim \mathcal{N}(0, 1)$ and $Z_1 \sim \mathcal{N}(0, 1)$ independently.

Fact 2 (Mill's Inequality [CB02, p. 123]). *Let $X \sim \mathcal{N}(0, 1)$, then*

$$\Pr[|X| > t] \leq \sqrt{\frac{2}{\pi}} \frac{e^{-\frac{t^2}{2}}}{t} < \frac{e^{-\frac{t^2}{2}}}{t}.$$

Definition 3 (Gamma distribution). *Given $\alpha, \theta \in \mathbb{R}^+$, the Gamma distribution, denoted by $\operatorname{Gamma}(\alpha, \theta)$, is a continuous probability distribution with PDF*

$$f(x) = \frac{1}{\Gamma(\alpha)\theta^\alpha} x^{\alpha-1} e^{-x/\theta}, x \in [0, \infty),$$

where $\Gamma(\alpha)$ is the Gamma function, that is,

$$\Gamma(\alpha) = \int_0^\infty t^{\alpha-1} e^{-t}\, dt.$$

Fact 3 (Stirling's formula [DS12, Theorem 5.7.4]).

$$\Gamma(\alpha) = \sqrt{\frac{2\pi}{\alpha}} \left(\frac{\alpha}{e}\right)^\alpha \left(1 + O\left(\frac{1}{\alpha}\right)\right).$$

Definition 4 (Beta distribution). *Given $\alpha, \beta \in \mathbb{R}^+$, the Beta distribution, denoted by* $\mathrm{Beta}(\alpha, \beta)$*, is a continuous probability distribution with PDF*

$$f(x) = \frac{1}{\mathrm{B}(\alpha, \beta)} x^{\alpha-1}(1-x)^{\beta-1}, x \in [0,1],$$

where $\mathrm{B}(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ *is the Beta function.*

Fact 4 ([DS12, Theorem 5.8.4]). *For two random variables X, Y,*

$$X \sim \mathrm{Gamma}(\alpha, \theta), Y \sim \mathrm{Gamma}(\beta, \theta) \text{ independently}$$
$$\Leftrightarrow X/(X+Y) \sim \mathrm{Beta}(\alpha, \beta), X+Y \sim \mathrm{Gamma}(\alpha+\beta, \theta) \text{ independently.}$$

Definition 5 (Chi-squared distribution). *Given $k \in \mathbb{N}$, the Chi-squared distribution χ^2_k is the distribution of the sum of the squares of k independent standard normal random variables.*

Fact 5 ([DS12, Section 8.2]). *The Chi-squared distribution is a special case of the Gamma distribution. Specifically,* $\chi^2_k = \mathrm{Gamma}(\alpha = k/2, \theta = 2)$.

Fact 6 (Transformation theorem, one-dimensional case [DS12, Theorem 3.8.4]). *Denote the PDF of the random variable X as $f_X(x)$. For a monotonic and differentiable function h, the PDF of $Y = h(X)$, denoted as $f_Y(y)$, is:*

$$f_Y(y) = \frac{f_X(x)}{|h'(x)|},$$

where $x = h^{-1}(y)$, $h'(x)$ is the derivative of $h(x)$.

Fact 7 (Rejection sampling [vN51]). *Rejection sampling is a well-known method to sample from a target distribution X with probability density $f(x)$, given the ability to sample from another distribution Y with probability density $g(x)$. Let M be a constant such that $f(x) \le Mg(x)$ for all x, then we can sample x from Y and accept it with probability $\frac{f(x)}{Mg(x)}$. The distribution of the accepted samples is exactly X, and the success probability is $\frac{1}{M}$.*

3.3 Quantum cryptography

Single-qubit gates. In this paper, we use the phase shift gate

$$P(\theta) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{bmatrix},$$

and the y-axis rotation gate

$$R_Y(\theta) = \begin{bmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix}.$$

Definition 6 (Trace distance). *The trace distance of two quantum states $\rho_0, \rho_1 \in D(\mathcal{H})$ is defined as*

$$\mathrm{TD}(\rho_0, \rho_1) \stackrel{def}{=} \frac{1}{2} \|\rho_0 - \rho_1\|_1 = \frac{1}{2} \mathrm{Tr}\left(\sqrt{(\rho_0 - \rho_1)^\dagger (\rho_0 - \rho_1)}\right).$$

Fact 8 (Data processing [Wil17]). *Let $\rho_0, \rho_1 \in D(\mathcal{H})$ and Φ be a CPTP map. Then,*

$$\mathrm{TD}(\Phi(\rho_0), \Phi(\rho_1)) \le \mathrm{TD}(\rho_0, \rho_1).$$

Fact 9 ([Wil17, Theorem 9.3.1]). *Let $|\psi\rangle, |\varphi\rangle$ be pure states. Then*

$$\mathrm{TD}(|\psi\rangle\langle\psi|, |\varphi\rangle\langle\varphi|) = \sqrt{1 - |\langle\psi|\varphi\rangle|^2}.$$

Fact 10 ([Wil17, Section 9.1.4]). *Let $\rho_0, \rho_1 \in D(\mathcal{H})$ be two quantum states. For a state drawn uniformly from the set $\{\rho_0, \rho_1\}$, the optimal distinguishing probability is*

$$\frac{1}{2} + \frac{1}{2} \operatorname{TD}(\rho_0, \rho_1).$$

Definition 7 (Diamond-norm distance [Wil17, Section 9.1.6]). *Let $\mathcal{E}, \mathcal{F} : L(\mathcal{H}_A) \to L(\mathcal{H}_B)$ be quantum channels. The diamond-norm distance is defined as*

$$\|\mathcal{E} - \mathcal{F}\|_\diamond \stackrel{def}{=} \sup_N \max_{\rho_{R_N A}} \|(\mathrm{id}_{R_N} \otimes \mathcal{E}_{A\to B})(\rho_{R_N A}) - (\mathrm{id}_{R_N} \otimes \mathcal{F}_{A\to B})(\rho_{R_N A})\|_1 ,$$

where $\rho_{R_N A} \in D(\mathcal{H}_{R_N} \otimes \mathcal{H}_A)$ and R_N is the reference system with dimension N.

The diamond-norm distance is the measure of distinguishability between channels $\mathcal{E}$ and $\mathcal{F}$.

Definition 8 (Choi representation [Wat18, Section 2.2.2]). *Let $\mathcal{E} : L(\mathcal{H}_A) \to L(\mathcal{H}_B)$ be a quantum channel, where $\mathcal{H}_A = \mathbb{C}^N$. The Choi representation of $\mathcal{E}$ is the operator*

$$J(\mathcal{E}) \stackrel{def}{=} \sum_{i,j\in[N]} \mathcal{E}(|i\rangle\langle j|) \otimes |i\rangle\langle j|.$$

Fact 11 ([Wat18, Section 3.4]). *Let $\mathcal{E}, \mathcal{F} : L(\mathcal{H}_A) \to L(\mathcal{H}_B)$ be quantum channels. Then*

$$\|\mathcal{E} - \mathcal{F}\|_\diamond \le \|J(\mathcal{E}) - J(\mathcal{F})\|_1 .$$

Definition 9 (Haar measure). *The Haar measure on $S(\mathcal{H})$ is the probability measure that is invariant under the action of every unitary operator, and we denote it by $\mu(S(\mathcal{H}))$. In this paper, we simplify $\mu(S(\mathbb{C}^{2^n}))$ to μ_n, which is the Haar measure over the n-qubit pure states set.*

Definition 10 (Haar-random states). *A Haar-random state $|\psi\rangle \in S(\mathcal{H})$ is a quantum state chosen uniformly at random from $S(\mathcal{H})$, according to the Haar measure $\mu(S(\mathcal{H}))$.*

Definition 11 (Standard Gaussian random vector). *Let $A_i \sim \mathcal{N}(0, \frac{1}{2})$ and $B_i \sim \mathcal{N}(0, \frac{1}{2})$ independently for all $i \in [N]$. We say that the complex random vector*

$$\boldsymbol{v} = (A_0 + iB_0, A_1 + iB_1, \ldots, A_{N-1} + iB_{N-1}) \in \mathbb{C}^N$$

is a standard (complex) Gaussian random vector.

In this paper, when we say standard Gaussian random vector, we always mean the standard complex Gaussian random vector. For simplicity, we slightly abuse terminology by also referring to the case $A_i, B_i \sim \mathcal{N}(0, \sigma^2)$ for any fixed $\sigma \in \mathbb{R}_{>0}$ as a standard Gaussian random vector, since they become equivalent after normalization.

Fact 12 ([Wat18, Section 7.2.1]). *Given a standard Gaussian random vector $\boldsymbol{v} \in \mathbb{C}^N$, the normalized random vector $\boldsymbol{v}/\|\boldsymbol{v}\|_2$ is a Haar-random state.*

Definition 12 (Quantum oracle). *A quantum oracle $\mathcal{O}$ is a quantum map. We denote a quantum circuit $\mathcal{A}$ with a black-box access to $\mathcal{O}$ by $\mathcal{A}^{\mathcal{O}}$ (each invocation of $\mathcal{O}$ is counted unit time). For oracles $\mathcal{O}_1$ and $\mathcal{O}_2$ and a number j we denote $\mathcal{A}^{\mathcal{O}_1,\mathcal{O}_2,j}$ as a circuit which does j black-box access to $\mathcal{O}_1$ followed by several black-box access to $\mathcal{O}_2$.*

For a function $f : \mathcal{X} \mapsto \mathcal{Y}$, we denote its unitary oracle

$$U_f \stackrel{def}{=} \sum_{x\in\mathcal{X}, y\in\mathcal{Y}} |x\rangle |y \oplus f(x)\rangle \langle x| \langle y| .$$

Definition 13 (Quantum-secure one-way functions [Zha21]). *A one-way function is a function $F : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are implicitly functions of a security parameter λ. We require that F can be evaluated by a classical algorithm in time polynomial in λ.*

A one-way function F is secure if no efficient quantum adversary $\mathcal{A}$ can invert F. Formally, for any efficient quantum adversary A, there exists a negligible function $\varepsilon = \varepsilon(\lambda)$ such that

$$\Pr_{x \leftarrow \mathcal{X}}[F(\mathcal{A}(F(x))) = F(x)] < \varepsilon.$$

Definition 14 (Quantum-secure pseudorandom functions (QS-PRFs) [Zha21]). *A PRF is a function $\mathsf{PRF} : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$, where $\mathcal{K}$ is the key-space, and $\mathcal{X}$ and $\mathcal{Y}$ are the domain and range. $\mathcal{K}$, $\mathcal{X}$ and $\mathcal{Y}$ are implicitly functions of the security parameter λ. We require that PRF can be evaluated by a classical algorithm in time polynomial in λ. We write $y = \mathsf{PRF}_k(x)$.*

A pseudorandom function PRF is quantum-secure if no efficient quantum adversary $\mathcal{A}$ making quantum queries can distinguish between a truly random function and the function PRF_k for a random k. That is, for every such $\mathcal{A}$, there exists a negligible function $\varepsilon = \varepsilon(\lambda)$ such that

$$\left| \Pr_{k \leftarrow \mathcal{K}}[\mathcal{A}^{U_{\mathsf{PRF}_k}}(1^\lambda) = 1] - \Pr_{f \leftarrow \mathcal{Y}^{\mathcal{X}}}[\mathcal{A}^{U_f}(1^\lambda) = 1] \right| < \varepsilon.^7$$

Definition 15 (Scalable quantum-secure pseudorandom function generator). *Let λ be the security parameter. A pair of algorithms $(\mathrm{KeyGen}, \mathrm{Eval})$ is called a scalable adaptive quantum-secure pseudorandom function generator if for any $n(\lambda), m(\lambda) \in \mathsf{poly}(\lambda)$, the following holds:*

1. *Key generation:* $\mathrm{KeyGen}(1^n, 1^m, 1^\lambda)$ *is a PPT algorithm which outputs a key k.*
2. *Evaluation:* $\mathrm{Eval}(1^n, 1^\lambda, k, x \in \{0,1\}^m)$ *is PT algorithm that outputs $f_k(x) \in \{0,1\}^n$.*
3. *Adaptive pseudorandomness: For any (non-uniform) QPT distinguisher $\mathcal{A}$*

$$\left| \Pr_{k \leftarrow \mathrm{KeyGen}(1^n, 1^m, 1^\lambda)} \left[\mathcal{A}^{U_{f_k}}(1^\lambda) = 1\right] - \Pr_{f \leftarrow \mathcal{Y}^{\mathcal{X}}} \left[\mathcal{A}^{U_f}(1^\lambda) = 1\right] \right| = \mathsf{negl}(\lambda),$$

where $\mathcal{X} = \{0,1\}^m, \mathcal{Y} = \{0,1\}^n$.

Fact 13 ([Zha21, Corollary 1.3]). *Assuming quantum-secure one-way functions (Definition 13) exist, quantum-secure pseudorandom functions (Definition 14) exist.*

In [Zha21], Zhandry proves Fact 13 by essentially showing that the GGM construction for PRFs due to Goldreich, Goldwasser, and Micali [GGM86] is quantum-secure when instantiated with a quantum-secure length-doubling PRG. The GGM construction can be adapted to yield arbitrary quantum-secure PRF with polynomially bounded input and output lengths, assuming the existence of quantum-secure polynomial-stretch PRGs. As the observation made by Aaronson and Christiano [Aar09,AC12], such PRGs exist under the assumption that quantum-secure OWFs exist. Consequently, Zhandry's result implicitly implies the following fact.

Fact 14 ([Zha21]). *Assuming quantum-secure one-way functions (Definition 13) exist, scalable quantum-secure pseudorandom function generators (Definition 15) exist.*

Definition 16 (Pseudorandom quantum state (PRS's) [JLS18]). *Let λ be the security parameter. Let $\mathcal{H}$ be a Hilbert space and $\mathcal{K}$ the key space, both parameterized by λ. A keyed family of quantum states $\{|\phi_k\rangle \in S(\mathcal{H})\}_{k \in \mathcal{K}}$ is pseudorandom, if the following two conditions hold:*

1. *Efficient generation: There is a polynomial-time quantum algorithm G that generates state $|\phi_k\rangle$ on input k. That is, for all $k \in \mathcal{K}$, $G(k) = |\phi_k\rangle$.*

[7] Although in Zhandry's original definition the adversary $\mathcal{A}$ takes no input, we add this input 1^λ to make it clear that an efficient quantum adversary means the running time is polynomially bounded in the security parameter λ.

2. *Pseudorandomness: Any polynomially many copies of $|\phi_k\rangle$ with the same random $k \in \mathcal{K}$ is computationally indistinguishable from the same number of copies of a Haar-random state. More precisely, for any efficient quantum algorithm $\mathcal{A}$ and any $t \in \mathsf{poly}(\lambda)$,*

$$\left| \Pr_{k \leftarrow \mathcal{K}} \left[\mathcal{A}(|\phi_k\rangle^{\otimes t}) = 1 \right] - \Pr_{|\psi\rangle \leftarrow \mu} \left[\mathcal{A}(|\psi\rangle^{\otimes t}) = 1 \right] \right| = \mathsf{negl}(\lambda),$$

where μ is the Haar measure on $S(\mathcal{H})$.

Definition 17 (Scalable pseudorandom quantum state (PRS) generator). *Let λ be the security parameter. A pair of algorithms* $(\mathrm{KeyGen}, \mathrm{CircuitGen})$ *is a scalable pseudorandom quantum state generator if for any $n(\lambda) \in \mathsf{poly}(\lambda)$ [8] the following holds:*

1. *Key generation:* $\mathrm{KeyGen}(1^n, 1^\lambda)$ *is a PPT algorithm which outputs a key k.*
2. *Circuit generation:* $\mathrm{CircuitGen}(1^n, 1^\lambda, k)$ *is a PT algorithm whose output is a quantum circuit for an isometry $V_k : \mathbb{C} \to \mathbb{C}^{2^n}$.*
3. *Pseudorandomness: For any (non-uniform) QPT distinguisher $\mathcal{A}$,*

$$\left| \Pr_{k \leftarrow \mathrm{KeyGen}(1^n, 1^\lambda)} \left[\mathcal{A}^{V_k}(1^\lambda) = 1 \right] - \Pr_{|\psi\rangle \leftarrow \mu_n} \left[\mathcal{A}^{W_{|\psi\rangle}}(1^\lambda) = 1 \right] \right| = \mathsf{negl}(\lambda),$$

where $W_{|\psi\rangle} : \mathbb{C} \to \mathbb{C}^{2^n}$ is the isometry $W_{|\psi\rangle} \stackrel{def}{=} |\psi\rangle$.

Claim. Let $(\mathrm{KeyGen}, \mathrm{CircuitGen})$ be a scalable pseudorandom quantum state generator as in Definition 17, then for any $n(\lambda) \in \mathsf{poly}(\lambda)$, it gives a family of n-qubit pseudorandom quantum states as in Definition 16.

The proof is provided in Section F.

Definition 18 (Scalable and adaptive pseudorandom function-like quantum state (PRFS) generator). *Let λ be the security parameter. A pair of algorithms* $(\mathrm{KeyGen}, \mathrm{CircuitGen})$ *is a scalable adaptive pseudorandom function-like state generator if for any $n(\lambda), m(\lambda) \in \mathsf{poly}(\lambda)$, the following holds:*

1. *Key generation:* $\mathrm{KeyGen}(1^n, 1^m, 1^\lambda)$ *is a PPT algorithm which outputs a key k.*
2. *Circuit generation:* $\mathrm{CircuitGen}(1^n, 1^m, 1^\lambda, k)$ *is a PT algorithm whose output is a quantum circuit for an isometry $V_k : \mathbb{C}^{2^m} \to \mathbb{C}^{2^{n+m}}$.*
3. *Adaptive pseudorandomness: For any (non-uniform) QPT distinguisher $\mathcal{A}$,*

$$\left| \Pr_{k \leftarrow \mathrm{KeyGen}(1^n, 1^m, 1^\lambda)} \left[\mathcal{A}^{V_k}(1^\lambda) = 1 \right] - \Pr_{\{|\psi_x\rangle \leftarrow \mu_n\}_{x \in \{0,1\}^m}} \left[\mathcal{A}^{W_{\{|\psi_x\rangle\}_x}}(1^\lambda) = 1 \right] \right| = \mathsf{negl}(\lambda),$$

where $W_{\{|\psi_x\rangle\}_x} : \mathbb{C}^{2^m} \to \mathbb{C}^{2^{n+m}}$ is the isometry

$$W_{\{|\psi_x\rangle\}_x} \stackrel{def}{=} \sum_{x=0}^{2^m-1} \left(|x\rangle |\psi_x\rangle \right) \langle x| .$$

[8] Technically, $n(\lambda)$ can be any function in our construction. The indistinguishability still holds even if $n(\lambda)$ is superpolynomial, since the distinguisher $\mathcal{A}$ is only allowed to run in polynomial time in λ. However, we do not discuss this case since it is irrelevant for practical cryptography schemes.

4 Isometric asymptotically random state generator

In this section, we show an isometric procedure which, given quantum oracle access to a random function, efficiently generates a quantum state that can be arbitrarily close to a Haar-random state statistically. More specifically, we show how to construct a unitary $\mathrm{RS}_{n,\lambda}^{U_f}$ such that $\{\mathrm{RS}_{n,\lambda}^{U_f}|0\rangle^{\otimes n}\}_f$ is statistically close to Haar-random states , with $\mathsf{negl}(\lambda)$ distance.

The following theorem, as the main result of this section, is a direct corollary of Theorem 4.

Theorem 3. *For every $\lambda \in \mathbb{N}^+$ security parameter, $n(\lambda) \in \mathsf{poly}(\lambda)$ number of qubits, there exists a unitary $\mathrm{RS}_{n,\lambda}^{U_f}$ which can be implemented efficiently given quantum oracle access to U_f, where $f : \mathcal{X} \to \mathcal{Y}$, $\mathcal{X} = \{0,1\}^{n+1}$ and $\mathcal{Y} = \{0,1\}^{\mathsf{poly}(n+\lambda)}$. For any (non-uniform, quantum-output) quantum algorithm $\mathcal{A}$ that can access an oracle at most $\mathsf{poly}(\lambda)$ times,*

$$\mathrm{TD}\left(\mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[\mathcal{A}^{V_f}(1^\lambda)\right], \mathop{\mathbb{E}}_{|\psi\rangle\leftarrow\mu_n}\left[\mathcal{A}^{W_{|\psi\rangle}}(1^\lambda)\right]\right) = \mathsf{negl}(\lambda),$$

where

- *$V_f : \mathbb{C} \to \mathbb{C}^{2^n}$ is the isometry $V_f \stackrel{def}{=} \mathrm{RS}_{n,\lambda}^{U_f}|0\rangle^{\otimes n}$.*
- *$W_{|\psi\rangle} : \mathbb{C} \to \mathbb{C}^{2^n}$ is the isometry $W_{|\psi\rangle} \stackrel{def}{=} |\psi\rangle$.*

4.1 Random amplitudes procedure

Consider a standard Gaussian random vector (Definition 11)

$$v = (A_0 + iB_0, A_1 + iB_1, \ldots, A_{N-1} + iB_{N-1}) \in \mathbb{C}^N,$$

where $A_i, B_i \sim \mathcal{N}(0,1)$ independently. The squared norm of each item $C_i = A_i^2 + B_i^2$ follows $\chi_2^2 = \mathrm{Gamma}(1,2)$ (Fact 5).

Definition 19 (Random amplitudes quantum state). *Let random variables $C_i \sim \chi_2^2$ i.i.d. The random state*

$$|\xi\rangle = \frac{1}{\sqrt{\sum_{i=0}^{N-1} C_i}} \sum_{i=0}^{N-1} \sqrt{C_i}\,|i\rangle$$

is called a random amplitudes quantum state.

The first and most critical step in our ARS / PRS algorithm is to prepare such a random amplitudes quantum state.

Lemma 1. *For any $n \in \mathbb{N}^+$, given $B_{t,z} \sim \mathrm{Beta}(2^{n-t-1}, 2^{n-t-1})$ all independent, where $t \in [n], z \in \{0,1\}^t$, the state prepared by the following procedure is an n-qubit random amplitudes quantum state.*

1. *Initialize with $|0\rangle^{\otimes n}$.*
2. *For $k := 1$ to n*
   - *Map the first k qubits from $|z\rangle|0\rangle$ to $|z\rangle\left(\sqrt{B_{k-1,z}}|0\rangle + \sqrt{1-B_{k-1,z}}|1\rangle\right)$ for each $z \in \{0,1\}^{k-1}$. i.e., apply the unitary*

$$U_k : |z\rangle|0\rangle \mapsto |z\rangle\left(\sqrt{B_{k-1,z}}|0\rangle + \sqrt{1-B_{k-1,z}}|1\rangle\right)$$

   on the first k qubits.

(Earlier, we have shown the example for $n = 3$ in Figure 2.)

The proof is provided in Section C.

Let $\mathcal{X} = \{0,1\}^{n+1}$ and $\mathcal{Y} = \{0,1\}^{\mathsf{poly}(n+\lambda)}$. Given the unitary oracle $U_f : |x\rangle|y\rangle \mapsto |x\rangle|y \oplus f(x)\rangle$ corresponding to a classical random function $f : \mathcal{X} \to \mathcal{Y}$, we now show a unitary algorithm $\mathrm{RA}_{n,\lambda}^{U_f}$ to implement the above random amplitudes procedure efficiently. That is, $|\xi_f\rangle = \mathrm{RA}_{n,\lambda}^{U_f}|0\rangle^{\otimes n}$ is ($\mathsf{negl}(\lambda)$-close to) a random amplitudes quantum state when $f \leftarrow \mathcal{Y}^{\mathcal{X}}$.

Define $B_f^{(t)}$ to be the unitary that computes (with $(n+\lambda)$-bit precision)

$$\theta_{t,z} = \frac{\arccos\sqrt{\mathcal{A}_{RB}(1^{n+\lambda}, 2^{n-t-1}, f(2^t+z))}}{2\pi} \in (0,1),$$

where $\mathcal{A}_{RB}$ is the sampling algorithm in Lemma 6 to sample from $\mathrm{Beta}_{R(2^{-n-\lambda})}(2^{n-t-1}, 2^{n-t-1})$. That is,

$$B_f^{(t)} : |z\rangle|0\rangle \mapsto |z\rangle|\theta_{t,z}\rangle.$$

Algorithm 1 Random amplitudes procedure $\mathrm{RA}_{n,\lambda}^{U_f}$

Input: n qubits, denoted by q_i

1: Prepare an $(n+\lambda)$-qubit ancilla register q_r, initialized to $|0\rangle^{\otimes(n+\lambda)}$
2: **for** $t := 0$ to $n-1$ **do**
3: Apply $B_f^{(t)}$ on $q_i[1,\cdots,t]$ and q_r
4: Apply controlled rotation on $q_i[t+1]$, controlled on q_r
5: Apply $B_f^{(t)}$ on $q_i[1,\cdots,t]$ and q_r
6: **end for**
7: Output q_i.

The implementation details of $B_f^{(t)}$ from U_f are shown in Figure 3. The implementation details of controlled rotation $R : |0\rangle|\theta\rangle \mapsto (\cos(2\pi\theta)|0\rangle + \sin(2\pi\theta)|1\rangle)|\theta\rangle$ are shown in Figure 4. Notice that each controlled-$R_Y(2\pi/2^i)$ gate will rotate $2\pi/2^{i+1}$ if the corresponding bit of θ is 1, so R indeed rotates $2\pi\theta$ in the end. The entire procedure for Algorithm 1 is shown in Figure 5.

In each iteration, the second $B_f^{(t)}$ uncomputes q_r, so $\mathrm{RA}_{n,\lambda}^{U_f}$ is indeed a unitary.

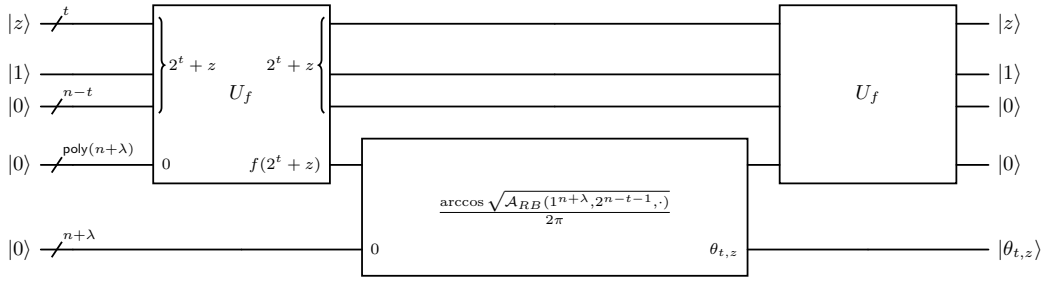


Fig. 3. $B_f^{(t)} : |z\rangle|0\rangle \mapsto |z\rangle|\theta_{t,z}\rangle$

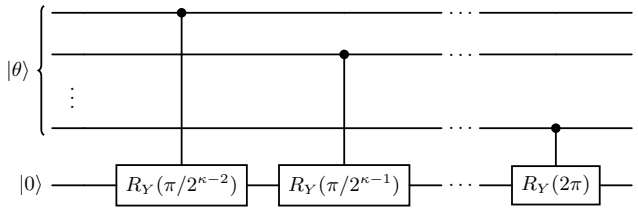


Fig. 4. Controlled rotation $R : |\theta\rangle |0\rangle \mapsto |\theta\rangle (\cos(2\pi\theta) |0\rangle + \sin(2\pi\theta) |1\rangle)$, where $\theta \in (0, 1)$, presented by κ qubits, is the angle of turn.

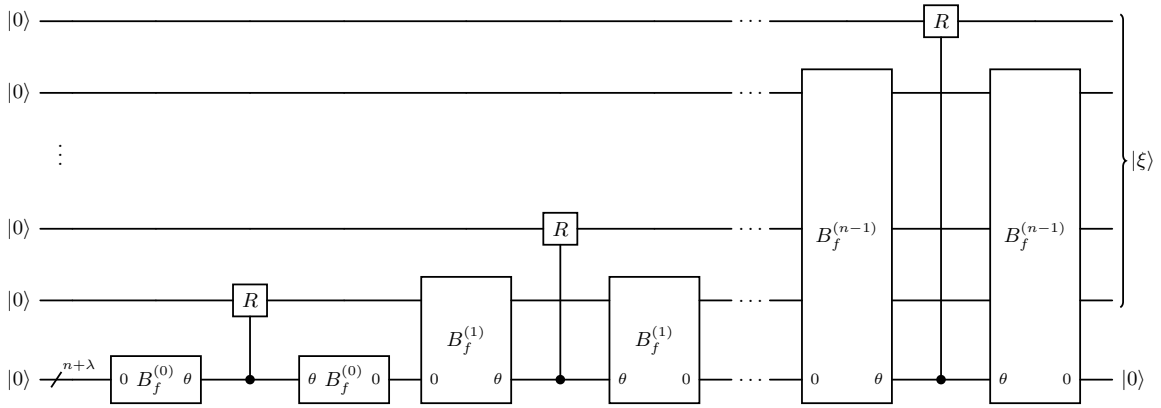


Fig. 5. The random amplitudes procedure for ARS/PRS.

4.2 Random phases procedure

Given quantum oracle access to a random classical function $f : \{0,1\}^{n+1} \to \{0,1\}^{\mathsf{poly}(n+\lambda)}$, now we know how to prepare a random amplitudes state, denoted as $|\xi\rangle = \sum_{z=0}^{2^n-1} \alpha_z |z\rangle$. The next step is to apply an independent random phase for each $\alpha_z |z\rangle$.

Lemma 2. *Let $C \sim \chi_2^2, U \sim \mathcal{U}(0,1)$ independently, then both real and imaginary parts of $\sqrt{C}e^{2\pi iU}$ follow standard normal distribution independently*

The proof is provided in Section D.

We now show a unitary algorithm $\mathrm{RP}_{n,\lambda}^{U_f}$ to implement the random phases procedure efficiently. To simplify the notation, for $f : \{0,1\}^{n+1} \to \{0,1\}^{\mathsf{poly}(n+\lambda)}$ we define the function $\check{f} : \{0,1\}^n \to \{0,1\}^\lambda$ as

$$\check{f}(z) = f(2^n + z)[1, \ldots, \lambda].$$

Here we add 2^n to ensure that the input of f is different from what has already been used in the random amplitudes procedure.

Algorithm 2 Random phases procedure $\mathrm{RP}_{n,\lambda}^{U_f}$

Input: n qubits, denoted by q_i
1: Prepare a λ-qubit ancilla register q_r, initialized to $|0\rangle^{\otimes\lambda}$
2: Apply $U_{\check{f}}$ on q_i and q_r.
3: **for** $k := 1$ to λ **do**
4: Apply phase shift gate $P(2\pi/2^k)$ on $q_r[k]$.
5: **end for**
6: Apply $U_{\check{f}}$ on q_i and q_r.
7: Output q_i.

Let the input be $\sum_{z=0}^{2^n-1}\alpha_z|z\rangle$. After applying the first $U_{\check{f}}$, the state is $\sum_{z=0}^{2^n-1}\alpha_z|z\rangle|\check{f}(z)\rangle$. Since each phase shift gate $P(2\pi/2^k)$ applies a phase $2\pi/2^k$ to $|\check{f}(z)\rangle$ when the k-th bit of $\check{f}(z)$ is 1, in total they apply a phase $2\pi\check{f}(z)/2^\lambda$ to $|\check{f}(z)\rangle$, and the entire state becomes $\sum_{z=0}^{2^n-1}\alpha_x|z\rangle e^{2\pi i\check{f}(z)/2^\lambda}|\check{f}(z)\rangle$. After uncomputing the ancilla register in the last step, the output is $\sum_{z=0}^{2^n-1}\alpha_z e^{2\pi i\check{f}(z)/2^\lambda}|z\rangle$.

4.3 Asymptotically random states

By combining the random amplitudes procedure Algorithm 1 and the random phases procedure Algorithm 2, we define our unitary random state procedure

$$\mathrm{RS}_{n,\lambda}^{U_f} \stackrel{\text{def}}{=} \mathrm{RP}_{n,\lambda}^{U_f}\,\mathrm{RA}_{n,\lambda}^{U_f}. \tag{1}$$

Fact 12, Lemma 1, and Lemma 2 imply that the ideal output state (infinite precision and a perfect sampling algorithm) of the random amplitudes procedure followed by the random phases procedure is a Haar-random state. We now show that the distance is negligible in λ when considering the actual algorithm output.

Definition 20. *Let $n \in \mathbb{N}^+$ be the number of qubits. For parameters $\mathbf{B} = (b_{t,z})_{t\in[n],z\in\{0,1\}^t}$ and $\mathbf{U} = (u_z)_{z\in\{0,1\}^n}$, where $b_{t,z} \in [0,1]$ and $u_z \in [0,1)$, we define $|\psi_{\mathbf{B},\mathbf{U}}\rangle$ as the quantum state prepared by the following procedure:*

1. *Initialize with $|0\rangle^{\otimes n}$.*
2. *For $t := 0$ to $n-1$*
   - *Map the first $t+1$ qubits from $|z\rangle|0\rangle$ to $|z\rangle(\sqrt{b_{t,z}}|0\rangle + \sqrt{1-b_{t,z}}|1\rangle)$ for each $z \in \{0,1\}^t$. i.e., apply the unitary*
   $$U^{(t)} : |z\rangle|0\rangle \mapsto |z\rangle(\sqrt{b_{t,z}}|0\rangle + \sqrt{1-b_{t,z}}|1\rangle)$$
   on the first $t+1$ qubits.
3. *Map $|z\rangle$ to $e^{2\pi i u_z}|z\rangle$ for each $z \in \{0,1\}^n$. i.e., apply the unitary*
$$U_P : |z\rangle \mapsto e^{2\pi i u_z}|z\rangle$$
on all n qubits.

Lemma 3. *Let $n \in \mathbb{N}^+$ the number of qubits. If*

$$\mathbf{B} = (b_{t,z})_{t\in[n],z\in\{0,1\}^t}, \mathbf{U} = (u_z)_{z\in\{0,1\}^n},$$
$$\mathbf{B}' = (b'_{t,z})_{t\in[n],z\in\{0,1\}^t}, \mathbf{U}' = (u'_z)_{z\in\{0,1\}^n}$$

satisfy $b_{t,z}, b'_{t,z} \in [0,1], u_z, u'_z \in [0,1)$, $|b_{t,z} - b'_{t,z}| < \delta_1$ and $|u_z - u'_z| < \delta_2$ for all t, z, then

$$\mathrm{TD}(|\psi_{\mathbf{B},\mathbf{U}}\rangle\langle\psi_{\mathbf{B},\mathbf{U}}|, |\psi_{\mathbf{B}',\mathbf{U}'}\rangle\langle\psi_{\mathbf{B}',\mathbf{U}'}|) < \sqrt{2^n n\delta_1} + 2\pi\delta_2.$$

Proof. This result follows directly from Lemma 4 with $m = 0$.

Theorem 4. *For every $n \in \mathbb{N}^+$ number of qubits, $\lambda \in \mathbb{N}^+$ security parameter, let $\mathcal{X} = \{0,1\}^{n+1}$, $\mathcal{Y} = \{0,1\}^{\mathsf{poly}(n+\lambda)}$, and $\mathrm{RS}_{n,\lambda}^{U_f}$ be defined as in eq. (1). For any (non-uniform, quantum-output) quantum algorithm $\mathcal{A}$ that can access an oracle at most l times,*

$$\mathrm{TD}\left(\mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[\mathcal{A}^{V_f}(1^\lambda)\right], \mathop{\mathbb{E}}_{|\psi\rangle\leftarrow\mu_n}\left[\mathcal{A}^{W_{|\psi\rangle}}(1^\lambda)\right]\right) < (1+\sqrt{2\pi})l\sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi l 2^{-\lambda} + 2^{-\lambda},$$

where

- *$V_f : \mathbb{C} \to \mathbb{C}^{2^n}$ is the isometry $V_f \stackrel{def}{=} \mathrm{RS}_{n,\lambda}^{U_f} |0\rangle^{\otimes n}$.*
- *$W_{|\psi\rangle} : \mathbb{C} \to \mathbb{C}^{2^n}$ is the isometry $W_{|\psi\rangle} \stackrel{def}{=} |\psi\rangle$.*

Proof. This result follows directly from Theorem 6 with $m = 0$, using eq. (3). For clarity, we also provide an independent proof in Section A.

5 Pseudorandom quantum state generators

Theorem 4 shows an ARS generator which generates asymptotically random states from real random functions. By replacing the real random functions with quantum-secure pseudorandom functions, we get a pseudorandom quantum state generator.

Theorem 5. *Let $(\mathrm{KeyGen}_{\mathsf{PRF}}, \mathrm{Eval}_{\mathsf{PRF}})$ be a scalable quantum-secure pseudorandom function generator as in Definition 15. The following is a scalable pseudorandom quantum state generator:*

- *$\mathrm{KeyGen}(1^n, 1^\lambda)$: Let $k \leftarrow \mathrm{KeyGen}_{\mathsf{PRF}}(1^{\mathsf{poly}(n+\lambda)}, 1^{n+1}, 1^\lambda)$. Output k.*
- *$\mathrm{CircuitGen}(1^n, 1^\lambda, k)$: Let function $\mathsf{PRF}_k : \{0,1\}^{n+1} \to \{0,1\}^{\mathsf{poly}(n+\lambda)}$ be defined as*

$$\mathsf{PRF}_k(\cdot) \stackrel{def}{=} \mathrm{Eval}_{\mathsf{PRF}}(1^{\mathsf{poly}(n+\lambda)}, 1^\lambda, k, \cdot).$$

Output the circuit for $\mathrm{RS}_{n,\lambda}^{U_{\mathsf{PRF}_k}} |0\rangle^{\otimes n}$ where $\mathrm{RS}_{n,\lambda}^{U_{\mathsf{PRF}_k}}$ is defined in eq. (1).

This result follows by replacing the random function in Theorem 4 with a quantum-secure pseudorandom function, which is a standard technique. A full proof is provided in Section B.

6 (Pseudo)random function-like quantum state generators

In this section, we modify our ARS algorithm $\mathrm{RS}_{n,\lambda}^{U_f}$ (eq. (1)) to accept an additional input state, so that we can get the construction for ARFS / PRFS.

For a classical function $f : \{0,1\}^{n+m+1} \to \{0,1\}^{\mathsf{poly}(n+m+\lambda)}$, we define the quantum oracle $U_f^{\leftarrow m}$ as follows: Prepare U_f and keep an m-qubit register. Each time the algorithm $\mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow m}}$ uses this oracle, this register is used as the first m qubits, and $\mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow m}}$ only provides the rest $n+1$ qubits (see Figure 6 with $a = n+m+1$ and $b = \mathsf{poly}(n+m+\lambda)$).

Similarly to $\mathrm{RS}_{n,\lambda}^{U_f}$, we define

$$\mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow m}} \stackrel{\mathrm{def}}{=} \mathrm{RP}_{n,\lambda}^{U_f^{\leftarrow m}} \mathrm{RA}_{n,\lambda}^{U_f^{\leftarrow m}}. \tag{2}$$

For the random amplitudes procedure, define $B_f'^{(t)}$ to compute

$$\theta_{t,z}^{(x)} = \frac{\arccos\sqrt{\mathcal{A}_{RB}(1^{n+\lambda}, 1^{n+m+\lambda}, 2^{n-t-1}, f(x\cdot 2^{n+1}+2^t+z))}}{2\pi}$$

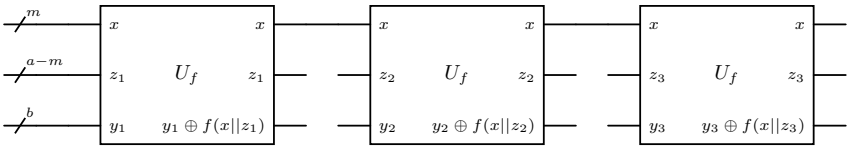


Fig. 6. How quantum oracle $U_f^{\leftarrow m}$ works, where $f : \{0,1\}^a \to \{0,1\}^b$.

with $(n+\lambda)$-bit precision. The first parameter of $\mathcal{A}_{RB}$ determines the precision of the rounded Beta sample, while the second makes its statistical distance at most $2^{-n-m-\lambda}$; thus, the latter does not enlarge the angle register. The algorithm $\mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow m}}$ is still a unitary (on $n+m$ qubits). Its random amplitudes procedure $\mathrm{RA}_{n,\lambda}^{U_f^{\leftarrow m}}$ is shown in Figure 7. After this, we apply random phases as before (i.e. $\mathrm{RP}_{n,\lambda}^{U_f^{\leftarrow m}}$).

When $m = 0$, the input register is trivial and $U_f^{\leftarrow 0} = U_f$. Moreover, $x = 0$, and the equality of the two sampler interfaces in Lemma 6 gives $B_f'^{(t)} = B_f^{(t)}$. Consequently,

$$\mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow 0}} = \mathrm{RS}_{n,\lambda}^{U_f}\,. \tag{3}$$

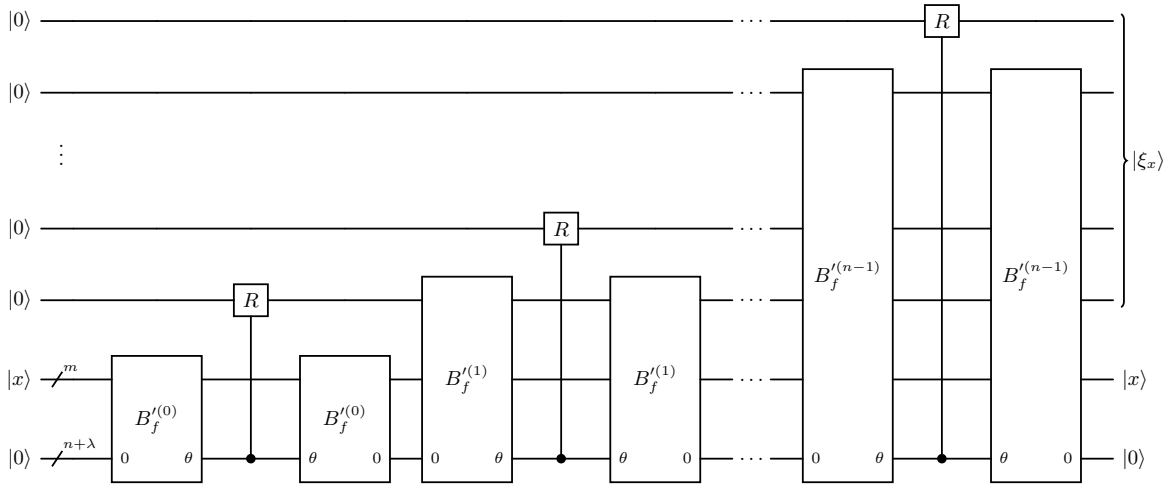


Fig. 7. The random amplitudes procedure $\mathrm{RA}_{n,\lambda}^{U_f^{\leftarrow m}}$ for ARFS/PRFS, where $B_f'^{(t)}$ takes the additional m-qubit input register into account.

Definition 21. *Let $n \in \mathbb{N}^+$ and $m \in \mathbb{N}$. For parameters $\mathbf{B} = (b_{t,z}^{(x)})_{t\in[n],z\in\{0,1\}^t,x\in\{0,1\}^m}$ and $\mathbf{U} = (u_z^{(x)})_{z\in\{0,1\}^n,x\in\{0,1\}^m}$, where $b_{t,z}^{(x)} \in [0,1]$ and $u_z^{(x)} \in [0,1)$, we define isometry $V_{\mathbf{B},\mathbf{U}} : \mathbb{C}^{2^m} \to \mathbb{C}^{2^{n+m}}$ as*

$$V_{\mathbf{B},\mathbf{U}} \stackrel{def}{=} \sum_{x=0}^{2^m-1} \left(|x\rangle\,|\psi_{\mathbf{B}_x,\mathbf{U}_x}\rangle\right)\langle x|\,,$$

where $\mathbf{B}_x = (b_{t,z}^{(x)})_{t\in[n],z\in\{0,1\}^t}$, $\mathbf{U}_x = (u_z^{(x)})_{z\in\{0,1\}^n}$, and $|\psi_{\mathbf{B}_x,\mathbf{U}_x}\rangle$ is defined in Definition 20. We use $\mathcal{E}_{\mathbf{B},\mathbf{U}}$ to denote $V_{\mathbf{B},\mathbf{U}}$ as a channel, i.e., $\mathcal{E}_{\mathbf{B},\mathbf{U}}(\rho) = V_{\mathbf{B},\mathbf{U}}\rho V_{\mathbf{B},\mathbf{U}}^\dagger$.

Lemma 4. *Let $n \in \mathbb{N}^+$ and $m \in \mathbb{N}$. If*

$$\mathbf{B} = (b_{t,z}^{(x)})_{t\in[n],z\in\{0,1\}^t,x\in\{0,1\}^m}, \mathbf{U} = (u_z^{(x)})_{z\in\{0,1\}^n,x\in\{0,1\}^m},$$
$$\mathbf{B}' = (b'^{(x)}_{t,z})_{t\in[n],z\in\{0,1\}^t,x\in\{0,1\}^m}, \mathbf{U}' = (u'^{(x)}_z)_{z\in\{0,1\}^n,x\in\{0,1\}^m}$$

satisfy $b_{t,z}^{(x)}, b'^{(x)}_{t,z} \in [0,1], u_z^{(x)}, u'^{(x)}_z \in [0,1)$, $\left|b_{t,z}^{(x)} - b'^{(x)}_{t,z}\right| < \delta_1$ and $\left|u_z^{(x)} - u'^{(x)}_z\right| < \delta_2$ for all t, z, x, then the following bound holds: [9]

$$\frac{1}{2}\left\|\mathcal{E}_{\mathbf{B},\mathbf{U}} - \mathcal{E}_{\mathbf{B}',\mathbf{U}'}\right\|_\diamond < \sqrt{2^n n\delta_1} + 2\pi\delta_2.$$

The proof is provided in Section E.

Theorem 6. *For every $n, \lambda \in \mathbb{N}^+$ and $m \in \mathbb{N}$, let $\mathcal{X} = \{0,1\}^{n+m+1}$ and $\mathcal{Y} = \{0,1\}^{\mathsf{poly}(n+m+\lambda)}$. With $\mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow m}}$ defined as in eq. (2), for any (non-uniform, quantum-output) quantum algorithm $\mathcal{A}$ that can access an oracle at most l times,*

$$\mathrm{TD}\left(\mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[\mathcal{A}^{V_f}(1^\lambda)\right], \mathop{\mathbb{E}}_{\{|\psi_x\rangle\leftarrow\mu_n\}_{x\in\{0,1\}^m}}\left[\mathcal{A}^{W_{\{|\psi_x\rangle\}_x}}(1^\lambda)\right]\right) < (1+\sqrt{2\pi})l\sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi l 2^{-\lambda} + 2^{-\lambda},$$

where

- *$V_f : \mathbb{C}^{2^m} \to \mathbb{C}^{2^{n+m}}$ is the isometry*

$$V_f \stackrel{def}{=} \mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow m}}(I_m \otimes |0\rangle^{\otimes n}).$$

- *$W_{\{|\psi_x\rangle\}_x} : \mathbb{C}^{2^m} \to \mathbb{C}^{2^{n+m}}$ is the isometry*

$$W_{\{|\psi_x\rangle\}_x} \stackrel{def}{=} \sum_{x=0}^{2^m-1}\left(|x\rangle|\psi_x\rangle\right)\langle x|.$$

Proof. Let $\varepsilon_1 = 2^{-n-\lambda}$ and $\varepsilon_2 = 2^{-\lambda}$. We use $\mathfrak{B}$ to denote the joint distribution of random variables $(B_{t,z}^{(x)})_{t\in[n],z\in\{0,1\}^t,x\in\{0,1\}^m}$ where $B_{t,z}^{(x)} \sim \mathrm{Beta}(2^{n-t-1}, 2^{n-t-1})$ independently. That is, $\mathbf{B} \leftarrow \mathfrak{B}$ is short for

$$\mathbf{B} = \left(b_{t,z}^{(x)} \leftarrow \mathrm{Beta}(2^{n-t-1}, 2^{n-t-1})\right)_{t\in[n],z\in\{0,1\}^t,x\in\{0,1\}^m}.$$

Similarly, we use $\mathfrak{U}$ to denote the joint distribution of random variables $(U_z^{(x)})_{z\in\{0,1\}^n,x\in\{0,1\}^m}$ where $U_z^{(x)} \sim \mathcal{U}(0,1)$ independently. We use notations $\mathbf{B}_x$, $\mathbf{U}_x$, $V_{\mathbf{B},\mathbf{U}}$ and $\mathcal{E}_{\mathbf{B},\mathbf{U}}$ for given $\mathbf{B}, \mathbf{U}$ as in Definition 21.

Consider the following hybrids:

P_0: $\mathbb{E}_{\{|\psi_x\rangle\leftarrow\mu_n\}_{x\in\{0,1\}^m}}\left[\mathcal{A}^{W_{\{|\psi_x\rangle\}_x}}(1^\lambda)\right]$.
P_1: $\mathbb{E}_{\mathbf{B}\leftarrow\mathfrak{B}\mathbf{U}\leftarrow\mathfrak{U}}\left[\mathcal{A}^{V_{\mathbf{B},\mathbf{U}}}(1^\lambda)\right]$.
P_2: $\mathbb{E}_{\mathbf{B}\leftarrow\mathfrak{B},\mathbf{U}\leftarrow\mathfrak{U}}\left[\mathcal{A}^{V_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}}}(1^\lambda)\right]$, where $\tilde{\mathbf{B}} = (\lfloor b_{t,z}^{(x)}/\varepsilon_1\rceil\varepsilon_1)_{t\in[n],z\in\{0,1\}^t,x\in\{0,1\}^m}$, $\tilde{\mathbf{U}} = (\lfloor u_z^{(x)}/\varepsilon_2\rfloor\varepsilon_2)_{z\in\{0,1\}^n,x\in\{0,1\}^m}$.
P_3: $\mathbb{E}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[\mathcal{A}^{V_{\mathbf{B}_f,\mathbf{U}_f}}(1^\lambda)\right]$, where

$$\mathbf{B}_f = \left(b_{t,z}^{(x)} := \mathcal{A}_{RB}(1^{n+\lambda}, 1^{n+m+\lambda}, 2^{n-t-1}, f(x\cdot 2^{n+1} + 2^t + z))\right)_{t\in[n],z\in\{0,1\}^t,x\in\{0,1\}^m},$$
$$\mathbf{U}_f = \left(u_z^{(x)} := f(x\cdot 2^{n+1} + 2^n + z)[1,\ldots,\lambda]/2^\lambda\right)_{z\in\{0,1\}^n,x\in\{0,1\}^m}.$$

[9] In the earlier conference version, the bound in this lemma was obtained through the trace norm of the Choi representation and incurred a factor of 2^m. Although this loss could be absorbed using the scalability of our PRS construction, the present proof avoids it by bounding the diamond-norm distance via the operator-norm distance.

P_4: $\mathbb{E}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[\mathcal{A}^{V_f}(1^\lambda)\right]$.

We now prove that each pair of consecutive distributions are close in trace distance.

- $\mathrm{TD}(P_0, P_1) = 0$.
  From Lemma 1 and Lemma 2, we know that for $\mathbf{B}, \mathbf{U}$ in S_1, for each x the coefficients of $|\psi_{\mathbf{B}_x,\mathbf{U}_x}\rangle$ follow the distribution of the normalized standard Gaussian random vector. Thus, from Fact 12, we know it's a Haar-random state. And for different x, they are all independent. Since
  $$W_{\{|\psi_x\rangle\}_x} \stackrel{\text{def}}{=} \sum_{x=0}^{2^m-1} (|x\rangle |\psi_x\rangle) \langle x|$$
  in P_0 and
  $$V_{\mathbf{B},\mathbf{U}} \stackrel{\text{def}}{=} \sum_{x=0}^{2^m-1} (|x\rangle |\psi_{\mathbf{B}_x,\mathbf{U}_x}\rangle) \langle x|$$
  in P_1 have exactly the same distribution, we have $\mathrm{TD}(P_0, P_1) = 0$.
- $\mathrm{TD}(P_1, P_2) < l(\sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi 2^{-\lambda})$.
  Given $\mathbf{B} = (b_{t,z}^{(x)})_{t\in[n],z\in\{0,1\}^t,x\in\{0,1\}^m}$ and $\mathbf{U} = (u_z^{(x)})_{z\in\{0,1\}^n,x\in\{0,1\}^m}$, we define hybrid $P_{1.j}|_{\mathbf{B},\mathbf{U}}$ as $\mathcal{A}^{V_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}},V_{\mathbf{B},\mathbf{U}},j}(1^\lambda)$. That is, the adversary $\mathcal{A}$ accesses the oracle $V_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}}$ for the first j queries, and then accesses the oracle $V_{\mathbf{B},\mathbf{U}}$ for the remaining queries.
  Since $\left|b_{t,z}^{(x)} - \lfloor b_{t,z}^{(x)}/\varepsilon_1\rceil\varepsilon_1\right| < \varepsilon_1$ and $\left|u_z^{(x)} - \lfloor u_z^{(x)}/\varepsilon_2\rfloor\varepsilon_2\right| < \varepsilon_2$, from Lemma 4 we have
  $$\begin{aligned}\frac{1}{2}\left\|\mathcal{E}_{\mathbf{B},\mathbf{U}} - \mathcal{E}_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}}\right\|_\diamond &< \sqrt{2^n n\varepsilon_1} + 2\pi\varepsilon_2 \\ &= \sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi 2^{-\lambda}.\end{aligned}$$
  By Fact 8, we have
  $$\mathrm{TD}(P_{1.j}|_{\mathbf{B},\mathbf{U}}, P_{1.j+1}|_{\mathbf{B},\mathbf{U}}) < \sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi 2^{-\lambda}, \tag{4}$$
  because the only difference between them is in the $(j+1)$-th query: one of them applies $V_{\mathbf{B},\mathbf{U}}$ and the other applies $V_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}}$, and the trace distance between them after this point is at most $\frac{1}{2}\left\|\mathcal{E}_{\mathbf{B},\mathbf{U}} - \mathcal{E}_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}}\right\|_\diamond$.
  Then we have
  $$\begin{aligned}\mathrm{TD}(P_1, P_2) &= \frac{1}{2}\|P_1 - P_2\|_1 \\ &= \frac{1}{2}\left\|\mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\mathfrak{B}\\ \mathbf{U}\leftarrow\mathfrak{U}}}[P_{1.0}|_{\mathbf{B},\mathbf{U}}] - \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\mathfrak{B}\\ \mathbf{U}\leftarrow\mathfrak{U}}}[P_{1.l}|_{\mathbf{B},\mathbf{U}}]\right\|_1 \\ &\le \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\mathfrak{B}\\ \mathbf{U}\leftarrow\mathfrak{U}}}\left[\frac{1}{2}\|P_{1.0}|_{\mathbf{B},\mathbf{U}} - P_{1.l}|_{\mathbf{B},\mathbf{U}}\|_1\right] \qquad \text{(Triangle inequality)} \\ &< \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\mathfrak{B}\\ \mathbf{U}\leftarrow\mathfrak{U}}}\left[l(\sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi 2^{-\lambda})\right] \qquad \text{(From eq. (4) and triangle inequality)} \\ &= l(\sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi 2^{-\lambda}).\end{aligned}$$
- $\mathrm{TD}(P_2, P_3) < 2^{-\lambda}$.
  In P_2, since $b_{t,z}^{(x)} \leftarrow \mathrm{Beta}(2^{n-t-1}, 2^{n-t-1})$ and $u_z^{(x)} \leftarrow \mathcal{U}(0,1)$, we know that $\lfloor b_{t,z}^{(x)}/\varepsilon_1\rceil\varepsilon_1$ follows the rounded Beta distribution $\mathrm{Beta}_{R(\varepsilon_1)}(2^{n-t-1}, 2^{n-t-1})$ (see Definition 23) and $\lfloor u_z^{(x)}/\varepsilon_2\rfloor$ is uniformly distributed in $\{0,1\}^\lambda$ (recall that $\varepsilon_2 = 2^{-\lambda}$).

We use $\tilde{\mathfrak{B}}$ to denote the joint distribution of discrete random variables $(\tilde{B}_{t,z}^{(x)})_{t\in[n],z\in\{0,1\}^t,x\in\{0,1\}^m}$ where $\tilde{B}_{t,z}^{(x)} \sim \mathrm{Beta}_{R(\varepsilon_1)}(2^{n-t-1}, 2^{n-t-1})$ independently, and $\tilde{\mathfrak{U}}$ to denote the joint distribution of discrete random variables $(\tilde{U}_z^{(x)})_{z\in\{0,1\}^n,x\in\{0,1\}^m}$ where $\tilde{U}_z^{(x)} \sim \mathcal{U}_\lambda/2^\lambda$ independently. Then P_2 can be rewritten as

$$P_2 = \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\tilde{\mathfrak{B}}\\ \mathbf{U}\leftarrow\tilde{\mathfrak{U}}}} \left[\mathcal{A}^{V_{\mathbf{B},\mathbf{U}}}(1^\lambda)\right].$$

In P_3, since $f \leftarrow \mathcal{Y}^{\mathcal{X}}$ is a random function, each $f(x)$ is an independent uniformly random string for $x \in \mathcal{X}$. We use $\dot{\mathfrak{B}}$ to denote the joint distribution of discrete random variables

$$(\mathcal{A}_{RB}(1^{n+\lambda}, 1^{n+m+\lambda}, 2^{n-t-1}, \tilde{U}_{t,z}^{(x)}))_{t\in[n],z\in\{0,1\}^t,x\in\{0,1\}^m}$$

where $\tilde{U}_{t,z}^{(x)} \sim \mathcal{U}_{\mathsf{poly}(n+m+\lambda)}$ independently. Then P_3 can be rewritten as

$$P_3 = \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\dot{\mathfrak{B}}\\ \mathbf{U}\leftarrow\tilde{\mathfrak{U}}}} \left[\mathcal{A}^{V_{\mathbf{B},\mathbf{U}}}(1^\lambda)\right].$$

Lemma 6 shows that

$$\Delta(\mathcal{A}_{RB}(1^{n+\lambda}, 1^{n+m+\lambda}, 2^{n-t-1}, \tilde{U}), \tilde{B}) < 2^{-n-m-\lambda},$$

where $\tilde{U} \sim \mathcal{U}_{\mathsf{poly}(n+m+\lambda)}$ and $\tilde{B} \sim \mathrm{Beta}_{R(2^{-n-\lambda})}(2^{n-t-1}, 2^{n-t-1})$. Since $\tilde{\mathfrak{B}}$ (and $\dot{\mathfrak{B}}$) has $2^m(2^n-1)$ random variables, by triangle inequality we have

$$\begin{aligned}\Delta(\tilde{\mathfrak{B}}, \dot{\mathfrak{B}}) &\le 2^m(2^n-1)2^{-n-m-\lambda}\\ &< 2^m \cdot 2^n \cdot 2^{-n-m-\lambda}\\ &= 2^{-\lambda}.\end{aligned}$$

Then we have

$$\begin{aligned}\mathrm{TD}(P_2, P_3) &= \mathrm{TD}\left(\mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\tilde{\mathfrak{B}}\\ \mathbf{U}\leftarrow\tilde{\mathfrak{U}}}} \left[\mathcal{A}^{V_{\mathbf{B},\mathbf{U}}}\right], \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\dot{\mathfrak{B}}\\ \mathbf{U}\leftarrow\tilde{\mathfrak{U}}}} \left[\mathcal{A}^{V_{\mathbf{B},\mathbf{U}}}\right]\right)\\ &\le \Delta\left(\tilde{\mathfrak{B}}, \dot{\mathfrak{B}}\right) \qquad \text{(Data processing inequality)}\\ &< 2^{-\lambda}.\end{aligned}$$

– $\mathrm{TD}(P_3, P_4) < l\sqrt{2\pi n}2^{-\frac{\lambda}{2}}$.
Given $f \in \mathcal{Y}^{\mathcal{X}}$, we define hybrid $P_{3.j}|_f$ as

$$\mathcal{A}^{V_f, V_{\mathbf{B}_f,\mathbf{U}_f},j}(1^\lambda).$$

That is, the adversary $\mathcal{A}$ accesses the oracle V_f for the first j queries, and then accesses the oracle $V_{\mathbf{B}_f,\mathbf{U}_f}$ for the remaining queries.
For each classical input $|x\rangle$, we now check the output of $\mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow m}}(|x\rangle \otimes |0\rangle^{\otimes n})$. In Figure 7, we compute $\theta_{t,z}^{(x)}$ with $(n+\lambda)$-bit precision, which is $\lfloor\theta_{t,z}^{(x)}/\varepsilon_1\rfloor\varepsilon_1$, and then use it to do rotation. In Algorithm 2, we apply the phase exactly as $e^{2\pi i f(x\cdot 2^{n+1}+2^n+z)[1,\cdots,\lambda]/2^\lambda}$ for each $|z\rangle$. Thus, $\mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow m}}(|x\rangle \otimes |0\rangle^{\otimes n})$ can be written as

$$\mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow m}}(|x\rangle \otimes |0\rangle^{\otimes n}) = |\psi_{\hat{\mathbf{B}}_{f,x},\mathbf{U}_{f,x}}\rangle,$$

where

$$\hat{\mathbf{B}}_{f,x} = \left(\hat{b}_{t,z}^{(x)} := \cos^2\left(2\pi\lfloor\theta_{t,z}^{(x)}/\varepsilon_1\rfloor\varepsilon_1\right)\right)_{t\in[n],z\in\{0,1\}^t},$$

$$\theta_{t,z}^{(x)} = \frac{\arccos\sqrt{\mathcal{A}_{RB}(1^{n+\lambda}, 1^{n+m+\lambda}, 2^{n-t-1}, f(x\cdot 2^{n+1}+2^t+z))}}{2\pi}.$$

The isometry V_f can thus be written as

$$V_f \stackrel{\text{def}}{=} \mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow m}}(I_m\otimes|0\rangle^{\otimes n}) = \sum_{x=0}^{2^m-1}\left(|x\rangle\,|\psi_{\hat{\mathbf{B}}_{f,x},\mathbf{U}_{f,x}}\rangle\right)\langle x| = V_{\hat{\mathbf{B}}_f,\mathbf{U}_f}. \quad \text{(Definition 21)}$$

Since $\left|\frac{\mathrm{d}\cos^2\theta}{\mathrm{d}\theta}\right| \le 1$, we have $\left|\hat{b}_{t,z}^{(x)} - b_{t,z}^{(x)}\right| \le 2\pi\varepsilon_1$, where

$$b_{t,z}^{(x)} = \mathcal{A}_{RB}(1^{n+\lambda}, 1^{n+m+\lambda}, 2^{n-t-1}, f(x\cdot 2^{n+1}+2^t+z))$$

as defined in P_3. Then from Lemma 4 we have

$$\frac{1}{2}\left\|\mathcal{E}_{\mathbf{B}_f,\mathbf{U}_f} - \mathcal{E}_{\hat{\mathbf{B}}_f,\mathbf{U}_f}\right\|_\diamond < \sqrt{2^n n\cdot 2\pi\varepsilon_1} = \sqrt{2\pi n}2^{-\frac{\lambda}{2}}.$$

By Fact 8, we have

$$\mathrm{TD}(P_{3.j}|_f, P_{3.j+1}|_f) < \sqrt{2\pi n}2^{-\frac{\lambda}{2}} \tag{5}$$

because the only difference between them is in the $(j+1)$-th query: one of them applies $V_{\mathbf{B}_f,\mathbf{U}_f}$ and the other applies $V_{\hat{\mathbf{B}}_f,\mathbf{U}_f}$, and the trace distance between them after this point is at most $\frac{1}{2}\left\|\mathcal{E}_{\mathbf{B}_f,\mathbf{U}_f} - \mathcal{E}_{\hat{\mathbf{B}}_f,\mathbf{U}_f}\right\|_\diamond$. Then we have

$$\begin{aligned}
\mathrm{TD}(P_3,P_4) &= \frac{1}{2}\|P_3-P_4\|_1 \\
&= \frac{1}{2}\left\|\mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}[P_{3.0}|_f] - \mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}[P_{3.l}|_f]\right\|_1 \\
&\le \mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[\frac{1}{2}\|P_{3.0}|_f - P_{3.l}|_f\|_1\right] \qquad \text{(Triangle inequality)} \\
&< \mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[l\sqrt{2\pi n}2^{-\frac{\lambda}{2}}\right] \qquad \text{(From eq. (5) and triangle inequality)} \\
&= l\sqrt{2\pi n}2^{-\frac{\lambda}{2}}.
\end{aligned}$$

Theorem 7. *Let* $(\mathrm{KeyGen}_{\mathsf{PRF}}, \mathrm{Eval}_{\mathsf{PRF}})$ *be a scalable quantum-secure pseudorandom function generator as in Definition 15. Then the following is a scalable and adaptive pseudorandom function-like quantum state generator:*

- $\mathrm{KeyGen}(1^n, 1^m, 1^\lambda)$*: Let* $k \leftarrow \mathrm{KeyGen}_{\mathsf{PRF}}(1^{\mathsf{poly}(n+m+\lambda)}, 1^{n+m+1}, 1^\lambda)$*, output* k*.*
- $\mathrm{CircuitGen}(1^n, 1^m, 1^\lambda, k)$*: Let function* $\mathsf{PRF}_k : \{0,1\}^{n+m+1} \to \{0,1\}^{\mathsf{poly}(n+m+\lambda)}$ *be defined as*

$$\mathsf{PRF}_k(\cdot) \stackrel{def}{=} \mathrm{Eval}_{\mathsf{PRF}}(1^{\mathsf{poly}(n+m+\lambda)}, 1^\lambda, k, \cdot).$$

Output the circuit for $\mathrm{RS}_{n,\lambda}^{U_{\mathsf{PRF}_k}^{\leftarrow m}}(I_m\otimes|0\rangle^{\otimes n})$*, where* $\mathrm{RS}_{n,\lambda}^{U_{\mathsf{PRF}_k}^{\leftarrow m}}$ *is defined in eq.* (2)*.*

This result follows by replacing the random function in Theorem 6 with a quantum-secure pseudorandom function, which is a standard technique. A full proof is provided in Section B.

Acknowledgements

We thank Kishor Bharti, Srijita Kundu, and Minglong Qin for helpful discussions.

This project is supported by the National Research Foundation, Singapore through the National Quantum Office, hosted in A*STAR, under its Centre for Quantum Technologies Funding Initiative (S24Q2D0009) and its Advanced Quantum Algorithms and Solutions Funding Initiative (S25Q9DA001 and S25Q9DA002).

References

Aar09. Scott Aaronson. Quantum copy-protection and quantum money. In *24th Annual IEEE Conference on Computational Complexity*, pages 229–242. IEEE Computer Soc., Los Alamitos, CA, 2009.

AC12. Scott Aaronson and Paul Christiano. Quantum money from hidden subspaces. In *STOC'12—Proceedings of the 2012 ACM Symposium on Theory of Computing*, pages 41–60. ACM, New York, 2012.

AGQY22. Prabhanjan Ananth, Aditya Gulati, Luowen Qian, and Henry Yuen. Pseudorandom (function-like) quantum state generators: New definitions and applications. In Eike Kiltz and Vinod Vaikuntanathan, editors, *Theory of Cryptography*, pages 237–265, Cham, 2022. Springer Nature Switzerland.

ALY23. Prabhanjan Ananth, Yao-Ting Lin, and Henry Yuen. Pseudorandom strings from pseudorandom quantum states. Cryptology ePrint Archive, Paper 2023/904, 2023.

AQY22. Prabhanjan Ananth, Luowen Qian, and Henry Yuen. Cryptography from Pseudorandom Quantum States. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022*, pages 208–236, Cham, 2022. Springer Nature Switzerland.

BCKM21. James Bartusek, Andrea Coladangelo, Dakshita Khurana, and Fermi Ma. One-Way Functions Imply Secure Computation in a Quantum World. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021*, pages 467–496, Cham, 2021. Springer International Publishing.

BCQ23. Zvika Brakerski, Ran Canetti, and Luowen Qian. On the Computational Hardness Needed for Quantum Cryptography. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:21, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

BEM24. Samuel Bouaziz-Ermann and Garazi Muguruza. Quantum pseudorandomness cannot be shrunk in a black-box way. Cryptology ePrint Archive, Paper 2024/291, 2024.

BGHD+23. Khashayar Barooti, Alex B. Grilo, Loïs Huguenin-Dumittan, Giulio Malavolta, Or Sattath, Quoc-Huy Vu, and Michael Walter. Public-key encryption with quantum keys. Cryptology ePrint Archive, Paper 2023/877, 2023.

BM58. G. E. P. Box and Mervin E. Muller. A Note on the Generation of Random Normal Deviates. *The Annals of Mathematical Statistics*, 29(2):610 – 611, 1958.

BS19. Zvika Brakerski and Omri Shmueli. (pseudo) random quantum states with binary phase. In *Theory of Cryptography: 17th International Conference, TCC 2019, Nuremberg, Germany, December 1–5, 2019, Proceedings, Part I*, page 229–250, Berlin, Heidelberg, 2019. Springer-Verlag.

BS20. Zvika Brakerski and Omri Shmueli. Scalable pseudorandom quantum states. In *Advances in Cryptology – CRYPTO 2020: 40th Annual International Cryptology Conference, CRYPTO 2020, Santa Barbara, CA, USA, August 17–21, 2020, Proceedings, Part II*, page 417–440, Berlin, Heidelberg, 2020. Springer-Verlag.

CB02. G. Casella and R.L. Berger. *Statistical Inference*. Duxbury advanced series. Duxbury Thomson Learning, 2002.

DS12. M.H. DeGroot and M.J. Schervish. *Probability and Statistics*. Addison-Wesley, 2012.

FPVY26. Ben Foxman, Natalie Parham, Francisca Vasconcelos, and Henry Yuen. Random Unitaries in Constant (Quantum) Time. In Shubhangi Saraf, editor, *17th Innovations in Theoretical Computer Science Conference (ITCS 2026)*, volume 362 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 61:1–61:25, Dagstuhl, Germany, 2026. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

GGM86. Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. Assoc. Comput. Mach.*, 33(4):792–807, 1986.

GLSV21. Alex B. Grilo, Huijia Lin, Fang Song, and Vinod Vaikuntanathan. Oblivious Transfer Is in MiniQCrypt. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021*, pages 531–561, Cham, 2021. Springer International Publishing.

Gol01. Oded Goldreich. *Foundations of Cryptography*. Cambridge University Press, 2001.

GR02. Lov Grover and Terry Rudolph. Creating superpositions that correspond to efficiently integrable probability distributions. *arXiv preprint quant-ph/0208112*, 2002.

JLS18. Zhengfeng Ji, Yi-Kai Liu, and Fang Song. Pseudorandom quantum states. In *Advances in cryptology—CRYPTO 2018. Part III*, volume 10993 of *Lecture Notes in Comput. Sci.*, pages 126–152. Springer, Cham, 2018.

KQST23. William Kretschmer, Luowen Qian, Makrand Sinha, and Avishay Tal. Quantum Cryptography in Algorithmica. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC, 2023.

Kre21. William Kretschmer. Quantum Pseudorandomness and Classical Complexity. In Min-Hsiu Hsieh, editor, *16th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2021)*, volume 197 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 2:1–2:20, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

LQS+24. Chuhan Lu, Minglong Qin, Fang Song, Penghui Yao, and Mingnan Zhao. Quantum pseudorandom scramblers. In *Theory of Cryptography: 22nd International Conference, TCC 2024, Milan, Italy, December 2–6, 2024, Proceedings, Part II*, page 3–35, Berlin, Heidelberg, 2024. Springer-Verlag.

MH25. Fermi Ma and Hsin-Yuan Huang. How to construct random unitaries. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, STOC '25, page 806–809, New York, NY, USA, 2025. Association for Computing Machinery.

MPSY24. Tony Metger, Alexander Poremba, Makrand Sinha, and Henry Yuen. Simple Constructions of Linear-Depth t-Designs and Pseudorandom Unitaries . In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 485–492, Los Alamitos, CA, USA, October 2024. IEEE Computer Society.

MT00. George Marsaglia and Wai Wan Tsang. A simple method for generating gamma variables. *ACM Transactions on Mathematical Software (TOMS)*, 26(3):363–372, 2000.

MY22. Tomoyuki Morimae and Takashi Yamakawa. Quantum commitments and signatures without one-way functions. In *Advances in Cryptology – CRYPTO*, 2022.

SHH25. Thomas Schuster, Jonas Haferkamp, and Hsin-Yuan Huang. Random unitaries in extremely low depth. *Science*, 389(6755):92–96, 2025.

vN51. John von Neumann. Various techniques used in connection with random digits. *Applied Mathematics Series*, 12:36–38, 1951.

Wat18. John Watrous. *The Theory of Quantum Information*. Cambridge University Press, 2018.

Wil17. Mark M. Wilde. *Quantum information theory*. Cambridge University Press, Cambridge, second edition, 2017.

Zha21. Mark Zhandry. How to construct quantum random functions. *J. ACM*, 68(5):Art. 33, 43, 2021.

A Alternative proof of ARS

Here we show an independent proof of Theorem 4.

Proof. Let $\varepsilon_1 = 2^{-n-\lambda}$, and $\varepsilon_2 = 2^{-\lambda}$. We use $\mathfrak{B}$ to denote the joint distribution of random variables $(B_{t,z})_{t\in[n],z\in\{0,1\}^t}$ where $B_{t,z} \sim \mathrm{Beta}(2^{n-t-1}, 2^{n-t-1})$ independently. That is,

$$\mathbf{B} \leftarrow \mathfrak{B}$$

is short for

$$\mathbf{B} = \left(b_{t,z} \leftarrow \mathrm{Beta}(2^{n-t-1}, 2^{n-t-1})\right)_{t\in[n],z\in\{0,1\}^t}.$$

Similarly, we use $\mathfrak{U}$ to denote the joint distribution of random variables $(U_z)_{z\in\{0,1\}^n}$ where $U_z \sim \mathcal{U}(0,1)$ independently. We use notation $|\psi_{\mathbf{B},\mathbf{U}}\rangle$ as in Definition 20.

Consider the following hybrids:

S_0:

$$\mathop{\mathbb{E}}_{|\psi\rangle\leftarrow\mu_n}\left[\mathcal{A}^{W_{|\psi\rangle}}(1^\lambda)\right].$$

S_1:

$$\mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\mathfrak{B}\\ \mathbf{U}\leftarrow\mathfrak{U}}}\left[\mathcal{A}^{W_{|\psi_{\mathbf{B},\mathbf{U}}\rangle}}(1^\lambda)\right].$$

S_2:

$$\mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\mathfrak{B}\\ \mathbf{U}\leftarrow\mathfrak{U}}}\left[\mathcal{A}^{W_{|\psi_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}}\rangle}}(1^\lambda)\right],$$

where $\tilde{\mathbf{B}} = (\lfloor b_{t,z}/\varepsilon_1\rceil\varepsilon_1)_{t\in[n],z\in\{0,1\}^t}, \tilde{\mathbf{U}} = (\lfloor u_z/\varepsilon_2\rfloor\varepsilon_2)_{z\in\{0,1\}^n}$.

S_3:

$$\mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[\mathcal{A}^{W_{|\psi_{\mathbf{B}_f,\mathbf{U}_f}\rangle}}(1^\lambda)\right],$$

where

$$\mathbf{B}_f = \left(b_{t,z} := \mathcal{A}_{RB}(1^{n+\lambda}, 2^{n-t-1}, f(2^t+z))\right)_{t\in[n],z\in\{0,1\}^t},$$
$$\mathbf{U}_f = \left(u_z := f(2^n+z)[1,\ldots,\lambda]/2^\lambda\right)_{z\in\{0,1\}^n}.$$

S_4:

$$\mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[\mathcal{A}^{V_f}(1^\lambda)\right].$$

We now prove that each pair of consecutive distributions are close in trace distance.

- $\mathrm{TD}(S_0, S_1) = 0$.
  From Lemma 1 and Lemma 2, we know that the coefficients of $|\psi_{\mathbf{B},\mathbf{U}}\rangle$ in S_1 follow the distribution of the normalized standard Gaussian random vector. Thus, from Fact 12, we know it's a Haar-random state. Since $|\psi\rangle$ in S_0 and $|\psi_{\mathbf{B},\mathbf{U}}\rangle$ in S_1 have exactly the same distribution, we have $\mathrm{TD}(S_0, S_1) = 0$.

– $\mathrm{TD}(S_1, S_2) < l(\sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi 2^{-\lambda})$.
Given $\mathbf{B} = (b_{t,z})_{t\in[n],z\in\{0,1\}^t}$ and $\mathbf{U} = (u_z)_{z\in\{0,1\}^n}$, we define hybrid $S_{1.j}|_{\mathbf{B},\mathbf{U}}$ as

$$\mathcal{A}^{W_{|\psi_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}}\rangle}, W_{|\psi_{\mathbf{B},\mathbf{U}}\rangle}, j}(1^\lambda).$$

That is, the adversary $\mathcal{A}$ accesses the oracle $W_{|\psi_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}}\rangle}$ for the first j queries, and then accesses the oracle $W_{|\psi_{\mathbf{B},\mathbf{U}}\rangle}$ for the remaining queries.
Since $|b_{t,z} - \lfloor b_{t,z}/\varepsilon_1 \rceil \varepsilon_1| < \varepsilon_1$ and $|u_z - \lfloor u_z/\varepsilon_2 \rfloor \varepsilon_2| < \varepsilon_2$, from Lemma 3 we have

$$\begin{aligned}
&\mathrm{TD}(|\psi_{\mathbf{B},\mathbf{U}}\rangle\langle\psi_{\mathbf{B},\mathbf{U}}|, |\psi_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}}\rangle\langle\psi_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}}|)\\
&< \sqrt{2^n n\varepsilon_1} + 2\pi\varepsilon_2\\
&= \sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi 2^{-\lambda}.
\end{aligned}$$

By Fact 8, we have

$$\mathrm{TD}(S_{1.j}|_{\mathbf{B},\mathbf{U}}, S_{1.j+1}|_{\mathbf{B},\mathbf{U}}) < \sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi 2^{-\lambda}, \tag{6}$$

because the only difference between them is in the $(j+1)$-th query one of them gets $|\psi_{\mathbf{B},\mathbf{U}}\rangle$ and the other gets $|\psi_{\tilde{\mathbf{B}},\tilde{\mathbf{U}}}\rangle$.
Then we have

$$\begin{aligned}
&\mathrm{TD}(S_1, S_2)\\
&= \frac{1}{2}\left\|S_1 - S_2\right\|_1\\
&= \frac{1}{2}\left\|\mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\mathfrak{B}\\ \mathbf{U}\leftarrow\mathfrak{U}}}[S_{1.0}|_{\mathbf{B},\mathbf{U}}] - \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\mathfrak{B}\\ \mathbf{U}\leftarrow\mathfrak{U}}}[S_{1.l}|_{\mathbf{B},\mathbf{U}}]\right\|_1\\
&\le \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\mathfrak{B}\\ \mathbf{U}\leftarrow\mathfrak{U}}}\left[\frac{1}{2}\left\|S_{1.0}|_{\mathbf{B},\mathbf{U}} - S_{1.l}|_{\mathbf{B},\mathbf{U}}\right\|_1\right] && \text{(Triangle inequality)}\\
&< \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\mathfrak{B}\\ \mathbf{U}\leftarrow\mathfrak{U}}}\left[l(\sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi 2^{-\lambda})\right] && \text{(From eq. (6) and triangle inequality)}\\
&= l(\sqrt{n}2^{-\frac{\lambda}{2}} + 2\pi 2^{-\lambda}).
\end{aligned}$$

– $\mathrm{TD}(S_2, S_3) < 2^{-\lambda}$.
In S_2, since $b_{t,z} \leftarrow \mathrm{Beta}(2^{n-t-1}, 2^{n-t-1})$ and $u_z \leftarrow \mathcal{U}(0,1)$, we know that $\lfloor b_{t,z}/\varepsilon_1 \rceil \varepsilon_1$ follows the rounded Beta distribution $\mathrm{Beta}_{R(\varepsilon_1)}(2^{n-t-1}, 2^{n-t-1})$ (see Definition 23) and $\lfloor u_z/\varepsilon_2 \rfloor$ is uniformly distributed in $\{0,1\}^\lambda$ (recall that $\varepsilon_2 = 2^{-\lambda}$).
We use $\tilde{\mathfrak{B}}$ to denote the joint distribution of discrete random variables $(\tilde{B}_{t,z})_{t\in[n],z\in\{0,1\}^t}$ where $\tilde{B}_{t,z} \sim \mathrm{Beta}_{R(\varepsilon_1)}(2^{n-t-1}, 2^{n-t-1})$ independently, and $\tilde{\mathfrak{U}}$ to denote the joint distribution of discrete random variables $(\tilde{U}_z)_{z\in\{0,1\}^n}$ where $\tilde{U}_z \sim \mathcal{U}_\lambda/2^\lambda$ independently. Then S_2 can be rewritten as

$$S_2 = \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\tilde{\mathfrak{B}}\\ \mathbf{U}\leftarrow\tilde{\mathfrak{U}}}}\left[\mathcal{A}^{W_{|\psi_{\mathbf{B},\mathbf{U}}\rangle}}(1^\lambda)\right].$$

In S_3, since $f \leftarrow \mathcal{Y}^{\mathcal{X}}$ is a random function, each $f(x)$ is an independent uniformly random string for $x \in \mathcal{X}$. We use $\dot{\mathfrak{B}}$ to denote the joint distribution of discrete random variables

$$(\mathcal{A}_{RB}(1^{n+\lambda}, 2^{n-t-1}, \tilde{U}_{t,z}))_{t\in[n],z\in\{0,1\}^t}$$

where $\tilde{U}_{t,z} \sim \mathcal{U}_{\mathsf{poly}(n+\lambda)}$ independently. Then S_3 can be rewritten as

$$S_3 = \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\dot{\mathfrak{B}}\\ \mathbf{U}\leftarrow\tilde{\mathfrak{U}}}} \left[\mathcal{A}^{W_{|\psi_{\mathbf{B},\mathbf{U}}\rangle}}(1^\lambda)\right].$$

Lemma 6 shows that

$$\Delta(\mathcal{A}_{RB}(1^{n+\lambda}, 2^{n-t-1}, \tilde{U}), \tilde{B}) < 2^{-n-\lambda},$$

where $\tilde{U} \sim \mathcal{U}_{\mathsf{poly}(n+\lambda)}$ and $\tilde{B} \sim \mathrm{Beta}_{R(2^{-n-\lambda})}(2^{n-t-1}, 2^{n-t-1})$. Since $\tilde{\mathfrak{B}}$ (and $\dot{\mathfrak{B}}$) has $2^n - 1$ random variables, by triangle inequality we have

$$\Delta(\tilde{\mathfrak{B}}, \dot{\mathfrak{B}}) \le (2^n - 1)2^{-n-\lambda} < 2^{-\lambda}.$$

Then we have

$$\begin{aligned}
&\mathrm{TD}(S_2, S_3)\\
&= \mathrm{TD}\left(\mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\tilde{\mathfrak{B}}\\ \mathbf{U}\leftarrow\tilde{\mathfrak{U}}}} \left[\mathcal{A}^{W_{|\psi_{\mathbf{B},\mathbf{U}}\rangle}}\right], \mathop{\mathbb{E}}_{\substack{\mathbf{B}\leftarrow\dot{\mathfrak{B}}\\ \mathbf{U}\leftarrow\tilde{\mathfrak{U}}}} \left[\mathcal{A}^{W_{|\psi_{\mathbf{B},\mathbf{U}}\rangle}}\right]\right)\\
&\le \Delta\left(\tilde{\mathfrak{B}}, \dot{\mathfrak{B}}\right) && \text{(Data processing inequality)}\\
&< 2^{-\lambda}.
\end{aligned}$$

– $\mathrm{TD}(S_3, S_4) < l\sqrt{2\pi n}2^{-\frac{\lambda}{2}}$.
Given $f \in \mathcal{Y}^{\mathcal{X}}$, we define hybrid $S_{3.j}|_f$ as

$$\mathcal{A}^{V_f, W_{|\psi_{\mathbf{B}_f,\mathbf{U}_f}\rangle}, j}(1^\lambda).$$

That is, the adversary $\mathcal{A}$ accesses the oracle V_f for the first j queries, and then accesses the oracle $W_{|\psi_{\mathbf{B}_f,\mathbf{U}_f}\rangle}$ for the remaining queries.
We now check the output of $V_f = \mathrm{RS}_{n,\lambda}^{U_f}|0\rangle^{\otimes n}$. In Algorithm 1, we compute $\theta_{t,z}$ with $(n+\lambda)$-bit precision, which is $\lfloor\theta_{t,z}/\varepsilon_1\rfloor\varepsilon_1$, and then use it to do rotation. In Algorithm 2, we apply the phase exactly as $e^{2\pi i f(2^n+z)[1,\cdots,\lambda]/2^\lambda}$ for each $|z\rangle$. Thus, $\mathrm{RS}_{n,\lambda}^{U_f}|0\rangle^{\otimes n}$ can be written as

$$\mathrm{RS}_{n,\lambda}^{U_f}|0\rangle^{\otimes n} = |\psi_{\hat{\mathbf{B}}_f,\mathbf{U}_f}\rangle,$$

where

$$\hat{\mathbf{B}}_f = \left(\hat{b}_{t,z} := \cos^2\left(2\pi\lfloor\theta_{t,z}/\varepsilon_1\rfloor\varepsilon_1\right)\right)_{t\in[n],z\in\{0,1\}^t},$$

$$\theta_{t,z} = \frac{\arccos\sqrt{\mathcal{A}_{RB}(1^{n+\lambda}, 2^{n-t-1}, f(2^t+z))}}{2\pi}.$$

That is,

$$V_f = W_{|\psi_{\hat{\mathbf{B}}_f,\mathbf{U}_f}\rangle}.$$

Since $\left|\frac{\mathrm{d}\cos^2\theta}{\mathrm{d}\theta}\right| \le 1$, we have $\left|\hat{b}_{t,z} - b_{t,z}\right| \le 2\pi\varepsilon_1$, where $b_{t,z} = \mathcal{A}_{RB}(1^{n+\lambda}, 2^{n-t-1}, f(2^t+z)$ as defined in S_3. Then from Lemma 3 we have

$$\begin{aligned}
&\mathrm{TD}(|\psi_{\mathbf{B}_f,\mathbf{U}_f}\rangle\langle\psi_{\mathbf{B}_f,\mathbf{U}_f}|, |\psi_{\hat{\mathbf{B}}_f,\mathbf{U}_f}\rangle\langle\psi_{\hat{\mathbf{B}}_f,\mathbf{U}_f}|)\\
&< \sqrt{2^n n \cdot 2\pi\varepsilon_1}\\
&= \sqrt{2\pi n}2^{-\frac{\lambda}{2}}.
\end{aligned}$$

By Fact 8, we have

$$\mathrm{TD}(S_{3.j}|_f, S_{3.j+1}|_f) < \sqrt{2\pi n}2^{-\frac{\lambda}{2}} \tag{7}$$

because the only difference between them is in the $(j+1)$-th query one of them gets $|\psi_{\mathbf{B}_f,\mathbf{U}_f}\rangle$ and the other gets $|\psi_{\hat{\mathbf{B}}_f,\mathbf{U}_f}\rangle$.
Then we have

$$\begin{aligned}
&\mathrm{TD}(S_3, S_4) \\
&= \frac{1}{2}\left\|S_3 - S_4\right\|_1 \\
&= \frac{1}{2}\left\|\mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[S_{3.0}|_f\right] - \mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[S_{3.l}|_f\right]\right\|_1 \\
&\leq \mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[\frac{1}{2}\left\|S_{3.0}|_f - S_{3.l}|_f\right\|_1\right] && \text{(Triangle inequality)} \\
&< \mathop{\mathbb{E}}_{f\leftarrow\mathcal{Y}^{\mathcal{X}}}\left[l\sqrt{2\pi n}2^{-\frac{\lambda}{2}}\right] && \text{(From eq. (7) and triangle inequality)} \\
&= l\sqrt{2\pi n}2^{-\frac{\lambda}{2}}.
\end{aligned}$$

B Proof of asymptotically random to pseudorandom

Proof of Theorem 5

Proof. PRF_k can be implemented efficiently classically, so U_{PRF_k} can be implemented efficiently. Since we already show the efficient implementation of Algorithm 1 and Algorithm 2, implementation of $\mathrm{RS}_{n,\lambda}^{U_{\mathsf{PRF}_k}}$ can be output efficiently.

Let $\mathcal{X} = \{0,1\}^{n+1}$, $\mathcal{Y} = \{0,1\}^{\mathsf{poly}(n+\lambda)}$ and $f \in \mathcal{Y}^{\mathcal{X}}$. We define

- $V_f : \mathbb{C} \to \mathbb{C}^{2^n}$ is the isometry $V_f \stackrel{\text{def}}{=} \mathrm{RS}_{n,\lambda}^{U_f} |0\rangle^{\otimes n}$.
- $W_{|\psi\rangle} : \mathbb{C} \to \mathbb{C}^{2^n}$ is the isometry $W_{|\psi\rangle} \stackrel{\text{def}}{=} |\psi\rangle$.

Thus, $\mathrm{CircuitGen}(1^n, 1^\lambda, k)$ outputs V_{PRF_k}.

To show the computational indistinguishability, we start from truly random functions. For any (non-uniform) QPT distinguisher $\mathcal{A}$, it can only access the oracle polynomially many times (polynomial in λ). When n is also in $\mathsf{poly}(\lambda)$, from Theorem 4 we have

$$\left| \Pr_{f \leftarrow \mathcal{Y}^{\mathcal{X}}} \left[\mathcal{A}^{V_f}(1^\lambda) = 1 \right] - \Pr_{|\psi\rangle \leftarrow \mu_n} \left[\mathcal{A}^{W_{|\psi\rangle}}(1^\lambda) = 1 \right] \right| = \mathsf{negl}(\lambda). \tag{8}$$

Assuming $\mathcal{A}$ can distinguish $\{V_{\mathsf{PRF}_k}\}_k$ and $\{V_f\}_f$ with non-negligible advantage $\nu(\lambda)$, then we can construct a QPT distinguisher $\mathcal{B}$ to distinguish $\{U_{\mathsf{PRF}_k}\}_k$ and $\{U_f\}_f$:

1. Construct $\mathrm{RS}_{n,\lambda}^{\mathcal{O}}$ using the given oracle $\mathcal{O}$.
2. Run $\mathcal{A}$, using $\mathrm{RS}_{n,\lambda}^{\mathcal{O}} |0\rangle^{\otimes n}$ as the oracle, and output the result.

This gives

$$\left| \Pr_{k \leftarrow \mathrm{KeyGen}(1^n, 1^\lambda)} \left[\mathcal{B}^{U_{\mathsf{PRF}_k}}(1^\lambda) = 1 \right] - \Pr_{f \leftarrow \mathcal{Y}^{\mathcal{X}}} \left[\mathcal{B}^{U_f}(1^\lambda) = 1 \right] \right| = \nu(\lambda),$$

which contradicts that $\{\mathsf{PRF}_k\}_k$ is quantum-secure pseudorandom. Thus, we have

$$\left| \Pr_{k \leftarrow \mathrm{KeyGen}(1^n, 1^\lambda)} \left[\mathcal{A}^{V_{\mathsf{PRF}_k}}(1^\lambda) = 1 \right] - \Pr_{f \leftarrow \mathcal{Y}^{\mathcal{X}}} \left[\mathcal{A}^{V_f}(1^\lambda) = 1 \right] \right| = \mathsf{negl}(\lambda). \tag{9}$$

From eq. (8) and eq. (9), we get

$$\left| \Pr_{k \leftarrow \mathrm{KeyGen}(1^n, 1^\lambda)} \left[\mathcal{A}^{V_{\mathsf{PRF}_k}}(1^\lambda) = 1 \right] - \Pr_{|\psi\rangle \leftarrow \mu_n} \left[\mathcal{A}^{W_{|\psi\rangle}}(1^\lambda) = 1 \right] \right| = \mathsf{negl}(\lambda).$$

Proof of Theorem 7

Proof. PRF_k can be implemented efficiently classically, so U_{PRF_k} can be implemented efficiently. By Lemma 6, the two-parameter sampler used in $B_f'^{(t)}$ is efficient. Together with the efficient implementations of Algorithm 1 and Algorithm 2, the circuit of $\mathrm{RS}_{n,\lambda}^{U_{\mathsf{PRF}_k}^{\leftarrow m}}$ can be output efficiently.

Let $\mathcal{X} = \{0,1\}^{n+m+1}, \mathcal{Y} = \{0,1\}^{\mathsf{poly}(n+m+\lambda)}$ and $f \in \mathcal{Y}^{\mathcal{X}}$, we define

- $V_f : \mathbb{C}^{2^m} \to \mathbb{C}^{2^{n+m}}$ is the isometry

$$V_f \stackrel{\text{def}}{=} \mathrm{RS}_{n,\lambda}^{U_f^{\leftarrow m}} (I_m \otimes |0\rangle^{\otimes n}).$$

- $W_{\{|\psi_x\rangle\}_x} : \mathbb{C}^{2^m} \to \mathbb{C}^{2^{n+m}}$ is the isometry

$$W_{\{|\psi_x\rangle\}_x} \stackrel{\text{def}}{=} \sum_{x=0}^{2^m-1} \left(|x\rangle |\psi_x\rangle \right) \langle x| .$$

Thus, $\mathrm{CircuitGen}(1^n, 1^m, 1^\lambda, k, \cdot)$ output V_{PRF_k}.

To show computational indistinguishability, we start from truly random functions. For any (non-uniform) quantum polynomial-time distinguisher $\mathcal{A}$, it can only access the oracle polynomially many times (polynomial in λ). When n and m are also in $\mathsf{poly}(\lambda)$, from Theorem 6 we have

$$\left| \Pr_{f \leftarrow \mathcal{Y}^{\mathcal{X}}} [\mathcal{A}^{V_f}(1^\lambda) = 1] - \Pr_{\{|\psi_x\rangle \leftarrow \mu_n\}_{x \in \{0,1\}^m}} \left[\mathcal{A}^{W_{\{|\psi_x\rangle\}_x}}(1^\lambda) = 1\right] \right| = \mathsf{negl}(\lambda).$$

Since PRF is a family of quantum-secure pseudorandom functions, similar to Theorem 5, we have

$$\left| \Pr_{k \leftarrow \mathrm{KeyGen}(1^n, 1^m, 1^\lambda)} [\mathcal{A}^{V_{\mathsf{PRF}_k}}(1^\lambda) = 1] - \Pr_{\{|\psi_x\rangle \leftarrow \mu_n\}_{x \in \{0,1\}^m}} \left[\mathcal{A}^{W_{\{|\psi_x\rangle\}_x}}(1^\lambda) = 1\right] \right| = \mathsf{negl}(\lambda).$$

C Proof of the random amplitudes quantum state from Beta variables

Proof of Lemma 1

Proof. We proceed by induction on t to show that the state after the t-th iteration, denoted by $|\phi_t\rangle$, can be written as

$$|\phi_t\rangle = \frac{1}{\sqrt{\sum_{z=0}^{2^t-1} W_{t,z}}} \sum_{z=0}^{2^t-1} \sqrt{W_{t,z}} \, |z\rangle \, |0\rangle^{\otimes n-t},$$

where each $W_{t,z} \sim \chi^2_{2^{n-t+1}}$ i.i.d. and is also independent of all $B_{t',z'}$ for $t' \geq t$. We take $|\phi_0\rangle = |0\rangle^{\otimes n}$ as the initial state before the first iteration.

- Base case: This holds for $t = 0$ since $|\phi_0\rangle = |0\rangle^{\otimes n}$.
- Inductive step: Assume this holds for t, then the state after the $(t+1)$-th iteration is

$$\begin{aligned}
&|\phi_{t+1}\rangle \\
&= \frac{1}{\sqrt{\sum_{z=0}^{2^t-1} W_{t,z}}} \sum_{z=0}^{2^t-1} \sqrt{W_{t,z}} \, |z\rangle \left(\sqrt{B_{t,z}} \, |0\rangle + \sqrt{1 - B_{t,z}} \, |1\rangle \right) |0\rangle^{\otimes n-t-1} \\
&= \frac{1}{\sqrt{\sum_{z=0}^{2^t-1} (W_{t,z} B_{t,z} + W_{t,z}(1 - B_{t,z}))}} \sum_{z=0}^{2^t-1} \left(\sqrt{W_{t,z} B_{t,z}} \, |z\rangle \, |0\rangle + \sqrt{W_{t,z}(1 - B_{t,z})} \, |z\rangle \, |1\rangle \right) |0\rangle^{\otimes n-t-1} \\
&= \frac{1}{\sqrt{\sum_{z=0}^{2^{t+1}-1} W_{t+1,z}}} \sum_{z=0}^{2^{t+1}-1} \sqrt{W_{t+1,z}} \, |z\rangle \, |0\rangle^{\otimes n-(t+1)},
\end{aligned}$$

where $W_{t+1,2z} = W_{t,z} B_{t,z}, W_{t+1,2z+1} = W_{t,z}(1 - B_{t,z})$.
Notice that

$$W_{t,z} = W_{t+1,2z} + W_{t+1,2z+1},$$

and

$$B_{t,z} = W_{t+1,2z} / (W_{t+1,2z} + W_{t+1,2z+1}).$$

From Fact 4 we have

$$\begin{aligned}
&W_{t+1,2z}, W_{t+1,2z+1} \sim \chi^2_{2^{n-t}} \text{ independently} \\
&\Leftrightarrow W_{t,z} \sim \chi^2_{2^{n-t+1}}, B_{t,z} \sim \mathrm{Beta}(2^{n-t-1}, 2^{n-t-1}) \text{ independently.}
\end{aligned} \tag{10}$$

Since for all $z \in \{0,1\}^t$, $W_{t,z} \sim \chi^2_{2^{n-t}}$ (inductive hypothesis) and $B_{t,z} \sim \mathrm{Beta}(2^{n-t-1}, 2^{n-t-1})$, eq. (10) gives that for all $z \in \{0,1\}^{t+1}$, $W_{t+1,z} \sim \chi^2_{2^{n-t}}$ i.i.d. and is independent of all $B_{t',z'}$ for $t' \geq t+1$. That is, the statement also holds for $t+1$.

The output state of this procedure is

$$|\phi_n\rangle = \frac{1}{\sqrt{\sum_{z=0}^{2^n-1} W_{n,z}}} \sum_{z=0}^{2^n-1} \sqrt{W_{n,z}}\,|z\rangle$$

with $W_{n,z} \sim \chi^2_2$ i.i.d., which is a random amplitudes quantum state (Definition 19).

D Proof of that random phase gives standard Gaussian

Proof of Lemma 2

Proof. Let $U_1 = e^{-C/2} \in (0,1]$. PDF of χ^2_2 is $f_C(c) = \frac{1}{2}e^{-c/2}, c \in [0,+\infty)$ (Definition 5), then from Fact 6 we know that PDF of U_1 is $f_U(u) = f_C(c)/\left|\frac{\mathrm{d}}{\mathrm{d}c}e^{-c/2}\right| = 1$, which means $U_1 \sim \mathcal{U}(0,1)$. Rewrite $\sqrt{C}e^{2\pi i U}$ as

$$\begin{aligned}\sqrt{C}e^{2\pi i U} &= \sqrt{-2\ln U_1}e^{2\pi i U} \\ &= \sqrt{-2\ln U_1}\cos(2\pi U) + i\sqrt{-2\ln U_1}\sin(2\pi U).\end{aligned}$$

According to Box-Muller transform (Fact 1), $\sqrt{-2\ln U_1}\cos 2\pi U, \sqrt{-2\ln U_1}\sin 2\pi U \sim \mathcal{N}(0,1)$ i.i.d.

E Proof of the diamond-norm bound via operator norm

Proof of Lemma 4

Proof. We first relate the diamond-norm distance between two isometric channels to the operator-norm distance between their isometries. Let V and W be isometries and let $\mathcal{E}_V$ and $\mathcal{E}_W$ be their corresponding channels. For every reference register R and every $\rho \in D(\mathcal{H}_R \otimes \mathcal{H}_A)$,

$$\begin{aligned}&\|(\mathrm{id}_R \otimes \mathcal{E}_V)(\rho) - (\mathrm{id}_R \otimes \mathcal{E}_W)(\rho)\|_1 \\ &\quad= \left\|(I_R \otimes (V-W))\rho(I_R \otimes V^\dagger) + (I_R \otimes W)\rho(I_R \otimes (V^\dagger - W^\dagger))\right\|_1 \\ &\quad\le 2\,\|V-W\|_{\mathrm{op}},\end{aligned}$$

where we use $\|AXB\|_1 \le \|A\|_{\mathrm{op}}\|X\|_1\|B\|_{\mathrm{op}}$, $\|\rho\|_1 = 1$, and the fact that an isometry has operator norm one. Taking the supremum over R and ρ gives

$$\frac{1}{2}\|\mathcal{E}_V - \mathcal{E}_W\|_\diamond \le \|V-W\|_{\mathrm{op}}. \tag{11}$$

Let $N = 2^n$. From the preparation procedure in Definition 20, for every $x \in \{0,1\}^m$ we can write

$$|\psi_{\mathbf{B}_x,\mathbf{U}_x}\rangle = \sum_{z\in[N]} e^{2\pi i u_z^{(x)}}\sqrt{c_z^{(x)}}\,|z\rangle, \qquad |\psi_{\mathbf{B}'_x,\mathbf{U}'_x}\rangle = \sum_{z\in[N]} e^{2\pi i u'^{(x)}_z}\sqrt{c'^{(x)}_z}\,|z\rangle,$$

where

$$c_z^{(x)} = \prod_{t\in[n]} \mathrm{Flip}(b^{(x)}_{t,z[1,\ldots,t]}, z[t+1]), \qquad c'^{(x)}_z = \prod_{t\in[n]} \mathrm{Flip}(b'^{(x)}_{t,z[1,\ldots,t]}, z[t+1]),$$

and $\mathrm{Flip}(a,d) = a$ if $d = 0$ and $\mathrm{Flip}(a,d) = 1-a$ if $d = 1$. Since all the factors lie in $[0,1]$, a hybrid over the n factors gives

$$\left|c_z^{(x)} - c'^{(x)}_z\right| < n\delta_1. \tag{12}$$

It follows that changing the amplitudes while keeping the phases fixed changes each output vector by at most

$$\begin{aligned}\left\||\psi_{\mathbf{B}_x,\mathbf{U}_x}\rangle - |\psi_{\mathbf{B}'_x,\mathbf{U}_x}\rangle\right\|_2^2 &= \sum_{z\in[N]} \left(\sqrt{c_z^{(x)}} - \sqrt{c'^{(x)}_z}\right)^2 \\ &\leq \sum_{z\in[N]} \left|c_z^{(x)} - c'^{(x)}_z\right| \\ &< 2^n n\delta_1,\end{aligned}$$

where we use $(\sqrt{a}-\sqrt{b})^2 \leq |a-b|$ for $a,b\geq 0$. Similarly, changing only the phases gives

$$\begin{aligned}\left\||\psi_{\mathbf{B}'_x,\mathbf{U}_x}\rangle - |\psi_{\mathbf{B}'_x,\mathbf{U}'_x}\rangle\right\|_2^2 &= \sum_{z\in[N]} c'^{(x)}_z \left|e^{2\pi i u_z^{(x)}} - e^{2\pi i u'^{(x)}_z}\right|^2 \\ &< (2\pi\delta_2)^2,\end{aligned}$$

since $\left|e^{ia}-e^{ib}\right| \leq |a-b|$ and $\sum_z c'^{(x)}_z = 1$. Hence, by the triangle inequality, for every x,

$$\left\||\psi_{\mathbf{B}_x,\mathbf{U}_x}\rangle - |\psi_{\mathbf{B}'_x,\mathbf{U}'_x}\rangle\right\|_2 < \sqrt{2^n n\delta_1} + 2\pi\delta_2. \tag{13}$$

Finally, the classical input register makes the isometries block diagonal. Therefore,

$$\begin{aligned}\left\|V_{\mathbf{B},\mathbf{U}} - V_{\mathbf{B}',\mathbf{U}'}\right\|_{\text{op}} &= \max_{x\in\{0,1\}^m} \left\||\psi_{\mathbf{B}_x,\mathbf{U}_x}\rangle - |\psi_{\mathbf{B}'_x,\mathbf{U}'_x}\rangle\right\|_2 \\ &< \sqrt{2^n n\delta_1} + 2\pi\delta_2.\end{aligned}$$

Combining this with eq. (11) proves the claim without a dimension-dependent loss in m.

F Proof of claim about the scalable pseudorandom state generator

Proof of: Definition 17 implies Definition 16

Proof. We write $\text{KeyGen}(1^n,1^\lambda)$ as $\text{KeyGen}(1^n,1^\lambda;r)$, where r is the randomness of this PPT algorithm. For given λ and n, let $\mathcal{K}$ be the set of all possible r. Then the family of quantum states $\{|\phi_r\rangle \in S(\mathbb{C}^{2^n})\}_{r\in\mathcal{K}}$ is given by:

$G(r)$:
1. $k_r = \text{KeyGen}(1^n,1^\lambda;r)$.
2. $C_r = \text{CircuitGen}(1^n,1^\lambda,k_r)$.
3. Output $|\phi_r\rangle = C_r()$.

Since CircuitGen is a PT algorithm, the size of the quantum circuit C_r is polynomially bounded in λ. The above algorithm G is indeed a polynomial-time quantum algorithm.

To show the pseudorandomness, assume that there exists $t\in\mathsf{poly}(\lambda)$ and a QPT distinguisher $\mathcal{A}$ such that

$$\left|\Pr_{r\leftarrow\mathcal{K}}\left[\mathcal{A}(|\phi_r\rangle^{\otimes t})=1\right] - \Pr_{|\psi\rangle\leftarrow\mu_n}\left[\mathcal{A}(|\psi\rangle^{\otimes t})=1\right]\right| = \varepsilon(\lambda) \notin \mathsf{negl}(\lambda).$$

Then we can construct a QPT distinguisher $\mathcal{B}$ by using V_{k_r} t times to get t copies of $|\phi_r\rangle$, and then running $\mathcal{A}$. This gives

$$\left|\Pr_{\substack{r\leftarrow\mathcal{K}\\ k_r=\text{KeyGen}(1^n,1^\lambda;r)}}\left[\mathcal{B}^{V_{k_r}}(1^\lambda)=1\right] - \Pr_{|\psi\rangle\leftarrow\mu_n}\left[\mathcal{B}^{W_{|\psi\rangle}}(1^\lambda)=1\right]\right| = \varepsilon(\lambda),$$

and contradicts the pseudorandomness of $(\text{KeyGen},\text{CircuitGen})$ according to Definition 17. Thus, $\{|\phi_r\rangle\}_{r\in\mathcal{K}}$ is pseudorandom according to Definition 16.

G Classical sampling algorithms

The main result of this section is an efficient deterministic classical algorithm to sample from the (finite precision) Beta distribution with given randomness as input (Lemma 6).

First, without considering precision, we show that, given a random variable from the normal distribution and a random variable from the uniform distribution, we can efficiently sample from Gamma$(\alpha, 1)$ for $\alpha \geq 1$, with failure probability bounded by some constant.

Lemma 5. *There exists a deterministic algorithm $\mathcal{A}_G$, such that for any $\alpha \geq 1$, with $X \sim \mathcal{N}(0,1)$ and $U \sim \mathcal{U}(0,1)$ independently, the following holds:*

- $\mathcal{A}_G(\alpha, X, U)|_{\mathcal{A}_G(\alpha,X,U)\neq\perp} \sim \text{Gamma}(\alpha, 1)$.
- $\Pr[\mathcal{A}_G(\alpha, X, U) = \perp] < 0.05$.

Proof. We use Marsaglia and Tsang's algorithm [MT00], and provide a rigorous proof of its correctness.

Algorithm 3 $\mathcal{A}_G$: Sample from standard Gamma distribution

Input: $\alpha \geq 1$, $x \leftarrow \mathcal{N}(0,1)$, $u \leftarrow \mathcal{U}(0,1)$
1: $d := \alpha - \frac{1}{3}, c := \frac{1}{\sqrt{9d}}$.
2: $v := (1 + cx)^3$.
3: **if** $v > 0$ and $\ln(u) < \frac{1}{2}x^2 + d - dv + d\ln(v)$ **then**
4: return dv
5: **else**
6: return $\perp$
7: **end if**

First, we show that the output follows Gamma$(\alpha, 1)$ when this algorithm succeeds. Let $h(x) = d(1+cx)^3$ for $x \in (-\frac{1}{c}, +\infty)$. If a random variable X follows the distribution with PDF $h(x)^{\alpha-1}e^{-h(x)}h'(x)/\Gamma(\alpha)$, then PDF of $Y = h(X)$ will be $y^{\alpha-1}e^{-y}/\Gamma(\alpha)$ (Fact 6, since $h(x)$ is monotonic and differentiable, $h'(x) > 0$), which means Y follows Gamma$(\alpha, 1)$ (recall that Gamma(α, θ) has PDF $f(y) = \frac{1}{\Gamma(\alpha)\theta^\alpha}y^{\alpha-1}e^{-y/\theta}$).

We have

$$\begin{aligned}
&h(x)^{\alpha-1}e^{-h(x)}h'(x)/\Gamma(\alpha) \\
&= d^{\alpha-1}(1+cx)^{3(\alpha-1)}e^{-d(1+cx)^3}3cd(1+cx)^2/\Gamma(\alpha) \\
&= 3cd^\alpha/\Gamma(\alpha)\cdot e^{-d}\cdot e^{d\ln(1+cx)^3 - d(1+cx)^3 + d}. \qquad (\text{Since } d = \alpha - \frac{1}{3}) \qquad (14)
\end{aligned}$$

Let $g(x) = d\ln(1+cx)^3 - d(1+cx)^3 + d, x \in (-\frac{1}{c}, +\infty)$. Rewrite eq. (14) as $Ce^{g(x)}$, where the constant $C = 3cd^\alpha/\Gamma(\alpha)e^{-d}$. Our target PDF for x is

$$f_0(x) = \begin{cases} 0, & x \in (-\infty, -\frac{1}{c}] \\ Ce^{g(x)}, & x \in (-\frac{1}{c}, +\infty) \end{cases}.$$

From Lemma 7 we know $g(x) \leq -\frac{1}{2}x^2$, so we can sample x from $\mathcal{N}(0,1)$, whose PDF is $f_1(x) = \frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}x^2}$, to do the rejection sampling (Fact 7, with $M = \sqrt{2\pi}C$) , accepting x with probability

$$p_{\text{accept}}(x) = \begin{cases} 0, & x \in (-\infty, -\frac{1}{c}] \\ e^{g(x)}/e^{-\frac{1}{2}x^2}, & x \in (-\frac{1}{c}, +\infty) \end{cases}.$$

In $\mathcal{A}_G$ this acceptance probability is simulated by checking whether $x > -\frac{1}{c}$ and $u < e^{g(x)+\frac{1}{2}x^2}$ for $u \leftarrow \mathcal{U}(0,1)$.

Above rejection sampling ensures that x follows the target PDF $f_0(x)$ when it is accepted, so that the output $dv = d(1+cx)^3 = h(x)$ follows $\mathrm{Gamma}(\alpha, 1)$, i.e.

$$\mathcal{A}_G(\alpha, X, U)|_{\mathcal{A}_G(\alpha,X,U)\neq\perp} \sim \mathrm{Gamma}(\alpha, 1).$$

Denote the success probability of this rejection sampling by p_α, which is also the success probability of this algorithm. Then

$$\begin{aligned} p_\alpha &= \frac{1}{M} && \text{(Fact 7)} \\ &= \frac{1}{\sqrt{2\pi}\cdot 3cd^\alpha e^{-d}/\Gamma(\alpha)} \\ &= \frac{e^d\Gamma(\alpha)}{3cd^\alpha\sqrt{2\pi}} \\ &= \frac{e^{\alpha-\frac{1}{3}}\sqrt{\alpha-\frac{1}{3}}\Gamma(\alpha)}{\sqrt{2\pi}(\alpha-\frac{1}{3})^\alpha}. \end{aligned}$$

From Lemma 8, we know $p_\alpha > 0.95$, so $\Pr[\mathcal{A}_G(\alpha, X, U) = \perp] < 0.05$.

Definition 22 (Rounded Gaussian distribution [BS20]). *We define the rounded standard Gaussian distribution $\mathcal{N}_{R(\varepsilon,B)}(0,1)$ to be the output distribution of the following process: Sample z from $\mathcal{N}(0,1)$, if $|z| > B$ then output 0, otherwise output $\lfloor z/\varepsilon \rfloor \varepsilon$.*

Fact 15 (Efficient Rounded Gaussian Sampling [BS20, Fact 3.14]). *There is a sampling algorithm $\mathcal{A}_{RN}$ that takes $1^m, B$ and a random tape as input, runs in polynomial time, i.e. $\mathsf{poly}(m, \log B)$, and samples from a distribution that has statistical distance at most 2^{-m} from the rounded standard Gaussian distribution $\mathcal{N}_{R(2^{-m},B)}(0,1)$.*

Definition 23 (Rounded Beta distribution). *We define the rounded Beta distribution $\mathrm{Beta}_{R(\varepsilon)}(\alpha,\beta)$ to be the output distribution of the following process: Sample z from $\mathrm{Beta}(\alpha,\beta)$, output $\lfloor z/\varepsilon \rceil \varepsilon$.*

Lemma 6. *There exists a deterministic polynomial sampling algorithm $\mathcal{A}_{RB}$ such that for any $m, \alpha \in \mathbb{N}^+$, with $U \sim \mathcal{U}_{\mathsf{poly}(m)}$ uniformly random, $\mathcal{A}_{RB}(1^m, \alpha, U)$ has statistical distance at most 2^{-m} from the rounded Beta distribution $\mathrm{Beta}_{R(2^{-m})}(\alpha,\alpha)$. More generally, the output precision and the statistical accuracy can be chosen independently: for any $p, s, \alpha \in \mathbb{N}^+$, with $U \sim \mathcal{U}_{\mathsf{poly}(p+s)}$ uniformly random,*

$$\mathcal{A}_{RB}(1^p, 1^s, \alpha, U)$$

has statistical distance at most 2^{-s} from $\mathrm{Beta}_{R(2^{-p})}(\alpha,\alpha)$. The two interfaces agree when the parameters are equal:

$$\mathcal{A}_{RB}(1^p, 1^p, \alpha, U) = \mathcal{A}_{RB}(1^p, \alpha, U).$$

Proof. Without considering precision, we first show a deterministic algorithm $\mathcal{A}_{B1}$ to sample from $\mathrm{Beta}(\alpha,\alpha)$ with a polynomial number of standard normal random variables and uniform random variables as inputs.

Algorithm 4 $\mathcal{A}_{B1}$

Input: $1^m, \alpha \in \mathbb{N}^+$, $x_1, \cdots, x_{2m} \leftarrow \mathcal{N}(0,1)$, $u_1, \cdots, u_{2m} \leftarrow \mathcal{U}(0,1)$

1: **for** $i := 1$ to m **do**
2: $\quad a := \mathcal{A}_G(\alpha, x_{2i-1}, u_{2i-1})$
3: $\quad$ **if** $a \neq \perp$ **then**
4: $\quad\quad$ **break**
5: $\quad$ **end if**
6: **end for**
7: **for** $i := 1$ to m **do**
8: $\quad b := \mathcal{A}_G(\alpha, x_{2i}, u_{2i})$
9: $\quad$ **if** $b \neq \perp$ **then**
10: $\quad\quad$ **break**
11: $\quad$ **end if**
12: **end for**
13: **if** $a = \perp$ or $b = \perp$ **then**
14: $\quad$ return $\perp$
15: **else**
16: $\quad$ return $a/(a+b)$
17: **end if**

Let the values of a and b at line 13 of Algorithm 4 be a^* and b^*. We have

$$\begin{aligned} &\Pr[a^* = \perp] \leq 0.05^m < 2^{-4m}, \\ &\Pr[b^* = \perp] \leq 0.05^m < 2^{-4m}. \qquad \text{(From Lemma 5)} \end{aligned}$$

Using the union bound, we have

$$\Pr[a^* = \perp \text{ or } b^* = \perp] < 2^{-4m+1}. \tag{15}$$

This gives

$$\Pr[\mathcal{A}_{B1}(1^m, \alpha, X_1, \cdots, X_{2m}, U_1, \ldots, U_{2m}) = \perp] < 2^{-4m+1},$$

where random variables $X_i \sim \mathcal{N}(0,1), U_i \sim \mathcal{U}(0,1)$ independently.

From Fact 4, we know that if both a^* and b^* follow Gamma$(\alpha, 1)$ independently, then $a^*/(a^*+b^*)$ follows Beta(α, α). Thus, we have

$$\mathcal{A}_{B1}(1^m, \alpha, X_1, \cdots, X_{2m}, U_1, \ldots, U_{2m})|_{\mathcal{A}_{B1}(1^m, \alpha, X_1, \cdots, X_{2m}, U_1, \ldots, U_{2m}) \neq \perp} \sim \text{Beta}(\alpha, \alpha).$$

We now modify $\mathcal{A}_{B1}$ to produce an output with m-bit precision. We define $\tilde{\mathcal{A}}_G^{\langle m_1 \rangle}$ to be the rounded version of $\mathcal{A}_G$, which only reads the first m_1 bits after the binary point of inputs (always round down even if the input is negative); i.e.

$$\tilde{\mathcal{A}}_G^{\langle m_1 \rangle}(\alpha, x, u) \stackrel{\text{def}}{=} \mathcal{A}_G(\alpha, \lfloor 2^{m_1} x \rfloor 2^{-m_1}, \lfloor 2^{m_1} u \rfloor 2^{-m_1}).$$

We need to consider when the output (after rounding to m-bit precision) of the original $\mathcal{A}_G$ and the new one using $\tilde{\mathcal{A}}_G^{\langle m_1 \rangle}$ differ. There are only two possible scenarios:

1. $\mathcal{A}_G(\alpha, x_i, u_i)$ and $\tilde{\mathcal{A}}_G^{\langle m_1 \rangle}(\alpha, x_i, u_i)$ behave differently in some iteration: One accepts and the other rejects.
2. Each pair of them behave the same, but the final output is different under m-bit precision, i.e. $\lfloor 2^m a/(a+b) \rceil 2^{-m} \neq \lfloor 2^m \tilde{a}/(\tilde{a}+\tilde{b}) \rceil 2^{-m}$, where $\tilde{a}, \tilde{b}$ are the outputs of $\tilde{\mathcal{A}}_G^{\langle m_1 \rangle}$ in the same iteration as a, b.

To detect the first one, we define a checker $\mathcal{C}$ as follows (and choose $\varepsilon = 2^{-m_1}$):

– $\mathcal{C}(\varepsilon, \alpha, x, u)$:

1. Let $d = \alpha - \frac{1}{3}, c = \frac{1}{\sqrt{9d}}$, and $\tilde{x} = \lfloor x/\varepsilon \rfloor \varepsilon, \tilde{u} = \lfloor u/\varepsilon \rfloor \varepsilon$.
2. If $\mathbf{1}\left\{(1+cx)^3 > 0\right\} \neq \mathbf{1}\left\{(1+c\tilde{x})^3 > 0\right\}$, output false.
3. Let $s(x) = \frac{1}{2}x^2 + d - d(1+cx)^3 + d\ln((1+cx)^3)$. If $\mathbf{1}\left\{u < e^{s(x)}\right\} \neq \mathbf{1}\left\{\tilde{u} < e^{s(\tilde{x})}\right\}$, output false.
4. Otherwise, output true.

This checker checks whether $\mathcal{A}_G$ behaves the same, both accepting or both rejecting, when run on (α, x, u) versus $(\alpha, \tilde{x}, \tilde{u})$ (see $\mathcal{A}_G$ Algorithm 3). Thus, as long as $\mathcal{C}(2^{-m_1}, \alpha, x_{2i-1}, u_{2i-1})$ outputs true, the branch taken by the if statement at line 3 is unaffected by replacing $\mathcal{A}_G$ with $\tilde{\mathcal{A}}_G^{\langle m_1 \rangle}$, and similarly for the branch at line 9.

We also exclude the case that $|x|$ is too large, because later we will use x sampled from the rounded Gaussian distribution, whose range is bounded. Now we have our new algorithm $\mathcal{A}_{B2}$.

Let $m_1 = 3m + 3 + \lceil \log_2(\eta + 3) \rceil$, where η is the constant from Lemma 10.

Algorithm 5 $\mathcal{A}_{B2}$

Input: $1^m, \alpha \in \mathbb{N}^+$, $x_1, \cdots, x_{2m} \leftarrow \mathcal{N}(0,1)$, $u_1, \cdots, u_{2m} \leftarrow \mathcal{U}(0,1)$

1: **for** $i := 1$ to m **do**
2: **if** $|x_{2i-1}| > 2m$ or not $\mathcal{C}(2^{-m_1}, \alpha, x_{2i-1}, u_{2i-1})$ **then**
3: return $\perp_R$
4: **end if**
5: $\tilde{a} := \tilde{\mathcal{A}}_G^{\langle m_1 \rangle}(1^m, \alpha, x_{2i-1}, u_{2i-1})$
6: $a := \mathcal{A}_G(1^m, \alpha, x_{2i-1}, u_{2i-1})$
7: **if** $\tilde{a} \neq \perp$ **then**
8: **break**
9: **end if**
10: **end for**
11: **for** $i := 1$ to m **do**
12: **if** $|x_{2i}| > 2m$ or not $\mathcal{C}(2^{-m_1}, \alpha, x_{2i}, u_{2i})$ **then**
13: return $\perp_R$
14: **end if**
15: $\tilde{b} := \tilde{\mathcal{A}}_G^{\langle m_1 \rangle}(1^m, \alpha, x_{2i}, u_{2i})$
16: $b := \mathcal{A}_G(1^m, \alpha, x_{2i}, u_{2i})$
17: **if** $\tilde{b} \neq \perp$ **then**
18: **break**
19: **end if**
20: **end for**
21: **if** $\tilde{a} = \perp$ or $\tilde{b} = \perp$ **then**
22: return $\perp$
23: **else if** $\lfloor 2^m a/(a+b) \rceil 2^{-m} \neq \lfloor 2^m \tilde{a}/(\tilde{a}+\tilde{b}) \rceil 2^{-m}$ **then**
24: return $\perp_R$
25: **else**
26: return $\lfloor 2^m \tilde{a}/(\tilde{a}+\tilde{b}) \rceil 2^{-m}$
27: **end if**

We need to check the probability that $\mathcal{A}_{B2}$ outputs $\perp_R$. From Fact 2, we have

$$\Pr_{x \leftarrow \mathcal{N}(0,1)}[|x| > 2m] < e^{-\frac{(2m)^2}{2}}/(2m). \tag{16}$$

We also have

$$\begin{aligned}
&\Pr_{x\leftarrow\mathcal{N}(0,1)}\left[\mathbf{1}\left\{(1+cx)^3>0\right\}\neq\mathbf{1}\left\{(1+c\lfloor x/\varepsilon\rfloor\varepsilon)^3>0\right\}\right]\\
&=\Pr_{x\leftarrow\mathcal{N}(0,1)}\left[\mathbf{1}\left\{x>-\frac{1}{c}\right\}\neq\mathbf{1}\left\{\lfloor x/\varepsilon\rfloor\varepsilon>-\frac{1}{c}\right\}\right]\\
&\leq\Pr_{x\leftarrow\mathcal{N}(0,1)}\left[x\in(-\frac{1}{c},-\frac{1}{c}+\varepsilon)\right]\\
&<\varepsilon.\qquad\text{(Since PDF of }\mathcal{N}(0,1)\text{ is }\frac{1}{\sqrt{2\pi}}e^{-\frac{1}{2}x^2}<1)
\end{aligned}\tag{17}$$

Since $\left|(e^{s(x)})'\right|\leq\eta$ (Lemma 10) and $|x-\lfloor x/\varepsilon\rfloor\varepsilon|<\varepsilon$, we know

$$\left|e^{s(\lfloor x/\varepsilon\rfloor\varepsilon)}-e^{s(x)}\right|<\eta\varepsilon.\qquad\text{(Mean value inequality)}\tag{18}$$

We also know that $|u-\lfloor u/\varepsilon\rfloor\varepsilon|<\varepsilon$. Then, when $u\leq\min\left\{e^{s(x_0)},e^{s(\lfloor x_0/\varepsilon\rfloor\varepsilon)}\right\}$,

$$\mathbf{1}\left\{u<e^{s(x_0)}\right\}=\mathbf{1}\left\{\lfloor u/\varepsilon\rfloor\varepsilon<e^{s(\lfloor x_0/\varepsilon\rfloor\varepsilon)}\right\}=1,$$

and when $u\geq\max\left\{e^{s(x_0)},e^{s(\lfloor x_0/\varepsilon\rfloor\varepsilon)}\right\}+\varepsilon$,

$$\mathbf{1}\left\{u<e^{s(x_0)}\right\}=\mathbf{1}\left\{\lfloor u/\varepsilon\rfloor\varepsilon<e^{s(\lfloor x_0/\varepsilon\rfloor\varepsilon)}\right\}=0.$$

Thus, for a fixed x_0 and uniformly random u,

$$\begin{aligned}
&\Pr_{u\leftarrow\mathcal{U}(0,1)}\left[\mathbf{1}\left\{u<e^{s(x_0)}\right\}\neq\mathbf{1}\left\{\lfloor u/\varepsilon\rfloor\varepsilon<e^{s(\lfloor x_0/\varepsilon\rfloor\varepsilon)}\right\}\right]\\
&\leq\Pr_{u\leftarrow\mathcal{U}(0,1)}\left[u\in\left(\min\left\{e^{s(x_0)},e^{s(\lfloor x_0/\varepsilon\rfloor\varepsilon)}\right\},\max\left\{e^{s(x_0)},e^{s(\lfloor x_0/\varepsilon\rfloor\varepsilon)}\right\}+\varepsilon\right)\right]\\
&=\max\left\{e^{s(x_0)},e^{s(\lfloor x_0/\varepsilon\rfloor\varepsilon)}\right\}+\varepsilon-\min\left\{e^{s(x_0)},e^{s(\lfloor x_0/\varepsilon\rfloor\varepsilon)}\right\}\\
&<(\eta+1)\varepsilon,\qquad\text{(from eq. (18))}
\end{aligned}$$

so

$$\Pr_{u\leftarrow\mathcal{U}(0,1),x\leftarrow\mathcal{N}(0,1)}\left[\mathbf{1}\left\{u<e^{s(x)}\right\}\neq\mathbf{1}\left\{\lfloor u/\varepsilon\rfloor\varepsilon<e^{s(\lfloor x/\varepsilon\rfloor\varepsilon)}\right\}\right]<(\eta+1)\varepsilon.\tag{19}$$

The final point at which $\mathcal{A}_{B_2}$ may output $\perp_R$ is in line 24. We show that the probability of this is small enough by discussing the following two cases:

1. At least one of a and b is not so close to 0. In this case, $\left|a/(a+b)-\tilde{a}/(\tilde{a}+\tilde{b})\right|$ can be bounded, then the probability that they differ after rounding can be bounded.
2. Both a and b are very close to 0. The probability that this happens is small enough.

$\mathcal{A}_{B_2}$ reaches line 24 only when both a and b are not $\perp$. Let x_a,x_b be the value of input x of $\mathcal{A}_G$ for a and b respectively; i.e.

$$a=d(1+cx_a)^3,b=d(1+cx_b)^3,$$

where d,c are defined in $\mathcal{A}_G$. Let $\tilde{x}_a,\tilde{x}_b$ be the rounded version of x_a,x_b respectively; i.e.

$$\tilde{x}_a=\lfloor x_a/2^{-m_1}\rfloor 2^{-m_1},\tilde{x}_b=\lfloor x_b/2^{-m_1}\rfloor 2^{-m_1}.$$

We have $x_a-\tilde{x}_a\in[0,2^{-m_1})$ and $x_b-\tilde{x}_b\in[0,2^{-m_1})$, so $(1+cx_a)-(1+c\tilde{x}_a)\in[0,c2^{-m_1})$ and $(1+cx_b)-(1+c\tilde{x}_b)\in[0,c2^{-m_1})$.

First consider the case that $x_a > -\frac{1}{c} + 2^{-m} + 2^{-m_1}$ or $x_b > -\frac{1}{c} + 2^{-m} + 2^{-m_1}$, which gives

$$1 + cx_a, 1 + c\tilde{x}_a > c2^{-m} \text{ or } 1 + cx_b, 1 + c\tilde{x}_b > c2^{-m}.$$

Define function $f(u, v) = u^3/(u^3 + v^3)$, and let $u = 1 + cx_a, v = 1 + cx_b$, $\tilde{u} = 1 + c\tilde{x}_a, \tilde{v} = 1 + c\tilde{x}_b$. We have

$$\begin{aligned}
&\left|\frac{a}{a+b} - \frac{\tilde{a}}{\tilde{a}+\tilde{b}}\right| \\
&= \left|\frac{(1+cx_a)^3}{(1+cx_a)^3 + (1+cx_b)^3} - \frac{(1+c\tilde{x}_a)^3}{(1+c\tilde{x}_a)^3 + (1+c\tilde{x}_b)^3}\right| \\
&= |f(u, v) - f(\tilde{u}, \tilde{v})| \\
&\leq \max\left\{ \left| (u-\tilde{u}) \min_{\substack{u>c2^{-m}\vee \\ v>c2^{-m}}} \frac{\partial f}{\partial u}(u, v) + (v-\tilde{v}) \min_{\substack{u>c2^{-m}\vee \\ v>c2^{-m}}} \frac{\partial f}{\partial v}(u, v) \right|, \right. \\
&\qquad \left. \left| (u-\tilde{u}) \max_{\substack{u>c2^{-m}\vee \\ v>c2^{-m}}} \frac{\partial f}{\partial u}(u, v) + (v-\tilde{v}) \max_{\substack{u>c2^{-m}\vee \\ v>c2^{-m}}} \frac{\partial f}{\partial v}(u, v) \right| \right\} \qquad \text{(Mean value inequality)} \\
&< c2^{-m_1} \cdot \frac{1}{c2^{-m}} \qquad \text{(Using Lemma 9)} \\
&= 2^{-m_1+m}.
\end{aligned} \tag{20}$$

In the penultimate line, a factor of 2 is not needed because $\frac{\partial f}{\partial u}(u, v)$ and $\frac{\partial f}{\partial v}(u, v)$ always have opposite signs.

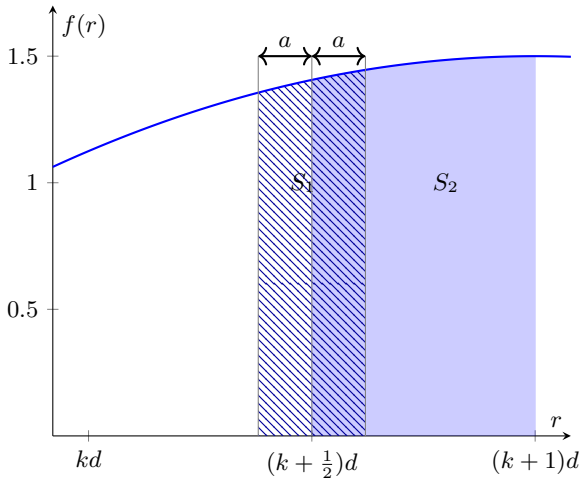
For the case that $x_a, x_b \in (-\frac{1}{c}, -\frac{1}{c} + 2^{-m} + 2^{-m_1}]$, we have

$$\begin{aligned}
a = d(1+cx_a)^3 &\in \left(0, dc^3(2^{-m} + 2^{-m_1})^3\right] \\
&= \left(0, \frac{1}{9}c(2^{-m} + 2^{-m_1})^3\right] \qquad \text{(Since } c^2 d = \frac{1}{9}\text{)} \\
&\subseteq \left(0, c\left(\frac{2^{-m} + 2^{-m_1}}{2}\right)^3\right) \\
&\subseteq \left(0, c2^{-3m}\right). \qquad \text{(Since } m_1 > m\text{)}
\end{aligned} \tag{21}$$

Similarly, we have $b \in (0, c2^{-3m})$.

We already show that a, b follow Gamma$(\alpha, 1)$ and $a/(a+b)$ follows Beta(α, α) when $a, b \neq \perp$, if we don't check and output $\perp_R$ earlier. Thus,

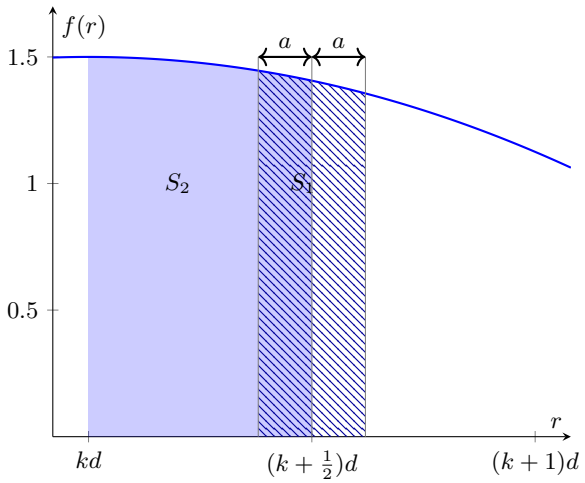
$$
\begin{aligned}
&\Pr[\ \mathcal{A}_{B2} \text{ outputs } \perp_R \text{ at line 24 }] \\
&\leq \Pr\left[\mathcal{A}_{B_2} \text{ reaches line 24 and } x_a, x_b \in (-\frac{1}{c}, -\frac{1}{c} + 2^{-m} + 2^{-m_1}]\right] \\
&\quad + \Pr\left[\mathcal{A}_{B2} \text{ outputs } \perp_R \text{ at line 24 } \mid \mathcal{A}_{B_2} \text{ reaches line 24 and } x_a \text{ or } x_b \text{ larger than } -\frac{1}{c} + 2^{-m} + 2^{-m_1}\right] \\
&< \Pr_{a,b \leftarrow \text{Gamma}(\alpha,1)}\left[a, b \in (0, c2^{-3m})\right] \qquad \text{(From eq. (21))} \\
&\quad + \Pr_{r \leftarrow \text{Beta}(\alpha,\alpha)}\left[\exists \tilde{r} : |r - \tilde{r}| < 2^{-m_1+m}, \lfloor 2^m r \rceil 2^{-m} \neq \lfloor 2^m \tilde{r} \rceil 2^{-m}\right] \qquad \text{(From eq. (20))} \\
&\leq (c2^{-3m})^2 \qquad \text{(Since PDF of Gamma}(\alpha, 1) \text{ is } f(x) = \frac{1}{\Gamma(\alpha)} x^{\alpha-1} e^{-x} \leq 1) \\
&\quad + \Pr_{r \leftarrow \text{Beta}(\alpha,\alpha)}\left[\exists \tilde{r} : |r - \tilde{r}| < 2^{-m_1+m}, \lfloor 2^m r \rceil 2^{-m} \neq \lfloor 2^m \tilde{r} \rceil 2^{-m}\right] \\
&= c^2 2^{-6m} + \Pr_{r \leftarrow \text{Beta}(\alpha,\alpha)}\left[r \in \left((k + \frac{1}{2})2^{-m} - 2^{-m_1+m}, (k + \frac{1}{2})2^{-m} + 2^{-m_1+m}\right) \text{ for some } k \in \mathbb{Z}\right] \\
&= c^2 2^{-6m} + \Pr_{r \leftarrow \text{Beta}(\alpha,\alpha)}\left[r \in \left((k + \frac{1}{2})2^{-m} - 2^{-m_1+m}, (k + \frac{1}{2})2^{-m} + 2^{-m_1+m}\right) \text{ for some } k \in \mathbb{Z}, k2^{-m} < \frac{1}{2}\right] \\
&\qquad + \Pr_{r \leftarrow \text{Beta}(\alpha,\alpha)}\left[r \in \left((k + \frac{1}{2})2^{-m} - 2^{-m_1+m}, (k + \frac{1}{2})2^{-m} + 2^{-m_1+m}\right) \text{ for some } k \in \mathbb{Z}, k2^{-m} \geq \frac{1}{2}\right] \\
&< c^2 2^{-6m} + \frac{2 \cdot 2^{-m_1+m}}{\frac{1}{2} \cdot 2^{-m}} \left(\Pr_{r \leftarrow \text{Beta}(\alpha,\alpha)}\left[r \in \left((k + \frac{1}{2})2^{-m}, (k+1)2^{-m}\right) \text{ for some } k \in \mathbb{Z}, k2^{-m} < \frac{1}{2}\right] \right. \\
&\qquad \left. + \Pr_{r \leftarrow \text{Beta}(\alpha,\alpha)}\left[r \in \left(k2^{-m}, (k + \frac{1}{2})2^{-m}\right) \text{ for some } k \in \mathbb{Z}, k2^{-m} \geq \frac{1}{2}\right] \right) \qquad (*) \\
&\qquad \text{(Since PDF of Beta}(\alpha, \alpha) \text{ is monotonic in } (0, \frac{1}{2}) \text{ and } (\frac{1}{2}, 1) \text{ respectively, see Figure 8 and Figure 9)} \\
&< c^2 2^{-6m} + 4 \cdot 2^{-m_1+2m} \cdot 1 \\
&< 2^{-6m} + 4 \cdot 2^{-m_1+2m}. \qquad \text{(Since } c = \frac{1}{\sqrt{9\alpha - 3}} \leq \frac{1}{\sqrt{6}} < 1) \qquad (22)
\end{aligned}
$$



$$S_1 < \frac{2a}{\frac{1}{2}d} S_2$$

$$\Pr\left[r \in \left((k+\frac{1}{2})d - a, (k+\frac{1}{2})d - a\right)\right] < \frac{2a}{\frac{1}{2}d} \Pr\left[r \in \left((k+\frac{1}{2})d, (k+1)d\right)\right]$$

Fig. 8. The case $kd < \frac{1}{2}$



$$S_1 < \frac{2a}{\frac{1}{2}d} S_2$$

$$\Pr\left[r \in \left((k+\frac{1}{2})d - a, (k+\frac{1}{2})d - a\right)\right] < \frac{2a}{\frac{1}{2}d} \Pr\left[r \in \left(kd, (k+\frac{1}{2})d\right)\right]$$

Fig. 9. The case $kd \geq \frac{1}{2}$

Notice that checker $\mathcal{C}$ runs at most $2m$ times with independent inputs x_i, u_i. Combining eq. (16), eq. (17), eq. (19) and eq. (22) we have

$$\begin{aligned}
&\Pr[\mathcal{A}_{B2}(1^m, \alpha, X_1, \cdots, X_{2m}, U_1, \ldots, U_{2m}) = \perp_R] \\
&< 2m\left(e^{-2m^2}/(2m) + (\eta+2)\cdot 2^{-m_1}\right) + 2^{-6m} + 4\cdot 2^{-m_1+2m} \\
&= e^{-2m^2} + 2(\eta+2)m\cdot 2^{-m_1} + 2^{-6m} + 4\cdot 2^{-m_1+2m},
\end{aligned} \tag{23}$$

where random variables $X_i \sim \mathcal{N}(0,1), U_i \sim \mathcal{U}(0,1)$ independently.

When the output is not $\perp_R$, $\mathcal{A}_{B2}$ outputs the rounded result of $\mathcal{A}_{B1}$ under the same inputs. Combining eq. (15) and eq. (23), we have

$$\begin{aligned}
&\Delta\left(\mathcal{A}_{B2}(1^m, \alpha, X_1, \cdots, X_{2m}, U_1, \ldots, U_{2m}), \mathrm{Beta}_{R(2^{-m})}(\alpha,\alpha)\right) \\
&< 2^{-4m+1} + e^{-2m^2} + 2(\eta+2)m\cdot 2^{-m_1} + 2^{-6m} + 4\cdot 2^{-m_1+2m}.
\end{aligned} \tag{24}$$

By modifying $\mathcal{A}_{B2}$ to use rounded inputs and avoid output $\perp$ or $\perp_R$, we obtain $\mathcal{A}_{B3}$.

Algorithm 6 $\mathcal{A}_{B3}$

Input: $1^m, \alpha \in \mathbb{N}^+$, $x_1, \cdots, x_{2m} \leftarrow \mathcal{N}_{R(2^{-m_1},2m)}(0,1)$, $u_1, \cdots, u_{2m} \leftarrow \{0,1\}^{m_1}$

for $i := 1$ to m **do**
  $\tilde{a} := \tilde{\mathcal{A}}_G^{\langle m_1\rangle}(1^m, \alpha, x_{2i-1}, u_{2i-1})$
  if $\tilde{a} \neq \perp$ **then**
    break
  end if
end for
for $i := 1$ to m **do**
  $\tilde{b} := \tilde{\mathcal{A}}_G^{\langle m_1\rangle}(1^m, \alpha, x_{2i}, u_{2i})$
  if $\tilde{b} \neq \perp$ **then**
    break
  end if
end for
if $\tilde{a} = \perp$ or $\tilde{b} = \perp$ **then**
  return $\frac{1}{2}$
else
  return $\lfloor 2^m \tilde{a}/(\tilde{a}+\tilde{b})\rceil 2^{-m}$
end if

Notice that in $\mathcal{A}_{B2}$ we have the following:

- Only the first m_1 bits (after the binary point) of x_i and u_i are used for computing $\tilde{a}$ and $\tilde{b}$.
- If any $|x_i| > 2m$, then it will output $\perp_R$.
- $\perp$ and $\perp_R$ are not in the support of $\mathrm{Beta}_{R(2^{-m})}(\alpha,\alpha)$.

Thus, the rounded version of the output of $\mathcal{A}_{B2}$ is the same as the output of $\mathcal{A}_{B3}$ with corresponding inputs, when it is not $\perp$ or $\perp_R$. From eq. (24) we have

$$\begin{aligned}
&\Delta\left(\mathcal{A}_{B3}(1^m, \alpha, \tilde{X}_1, \cdots, \tilde{X}_{2m}, \tilde{U}), \mathrm{Beta}_{R(2^{-m})}(\alpha,\alpha)\right) \\
&< 2^{-4m+1} + e^{-2m^2} + 2(\eta+2)m\cdot 2^{-m_1} + 2^{-6m} + 4\cdot 2^{-m_1+2m},
\end{aligned} \tag{25}$$

where random variables $\tilde{X}_i \sim \mathcal{N}_{R(2^{-m_1},2m)}(0,1)$, $\tilde{U} \sim \mathcal{U}_{m_1\cdot 2m}$ independently.

Finally, algorithm $\mathcal{A}_{RB}$ is defined as follows:

1. Run $\mathcal{A}_{RN}$ in Fact 15 multiple times to sample each x_i with fresh randomness.
2. Run $\mathcal{A}_{B3}$ with those x_i's and a fresh uniform random u_i.

More formally,

$$\begin{aligned}&\mathcal{A}_{RB}(1^m, \alpha, \tilde{U}_1' || \tilde{U}_2' || \cdots || \tilde{U}_{2m}' || \tilde{U}_1 || \tilde{U}_2 || \cdots || \tilde{U}_{2m}) \\ &\stackrel{\text{def}}{=} \mathcal{A}_{B3}\left(1^m, \alpha, \mathcal{A}_{RN}(1^{m_1}, 2m, \tilde{U}_1'), \cdots, \mathcal{A}_{RN}(1^{m_1}, 2m, \tilde{U}_{2m}'), \tilde{U}_1, \cdots, \tilde{U}_{2m}\right).\end{aligned}$$

From Fact 15, we have

$$\Delta\left(\mathcal{A}_{RN}(1^{m_1}, 2m, \tilde{U}), \mathcal{N}_{R(2^{-m_1}, 2m)}(0,1)\right) \leq 2^{-m_1},$$

where $\tilde{U} \sim \mathcal{U}_{\mathsf{poly}(m)}$. Then, from the triangle inequality and the data processing inequality, we have

$$\begin{aligned}&\Delta\Big(\\ &\quad \mathcal{A}_{B3}\left(1^m, \alpha, \mathcal{A}_{RN}(1^{m_1}, 2m, \tilde{U}_1'), \cdots, \mathcal{A}_{RN}(1^{m_1}, 2m, \tilde{U}_{2m}'), \tilde{U}_1, \cdots, \tilde{U}_{2m}\right), \\ &\quad \mathcal{A}_{B3}\left(1^m, \alpha, \tilde{X}_1, \cdots, \tilde{X}_{2m}, \tilde{U}_1, \cdots, \tilde{U}_{2m}\right) \\ &\Big) \leq 2m \cdot 2^{-m_1},\end{aligned}$$

which gives

$$\Delta\left(\mathcal{A}_{RB}(1^m, \alpha, \tilde{U}), \mathcal{A}_{B3}(1^m, \alpha, \tilde{X}_1, \cdots, \tilde{X}_{2m}, \tilde{U}_1, \cdots, \tilde{U}_{2m})\right) \leq 2m \cdot 2^{-m_1}, \tag{26}$$

where $\tilde{U} = \tilde{U}_1' \| \tilde{U}_2' \| \cdots \| \tilde{U}_{2m}' \| \tilde{U}_1 \| \tilde{U}_2 \| \cdots \| \tilde{U}_{2m} \sim \mathcal{U}_{\mathsf{poly}(m)}$.

Combining eq. (25) and eq. (26), we have

$$\begin{aligned}&\Delta\left(\mathcal{A}_{RB}(1^m, \alpha, \tilde{U}), \mathrm{Beta}_{R(2^{-m})}(\alpha, \alpha)\right) \\ &< 2^{-4m+1} + e^{-2m^2} + 2(\eta+2)m \cdot 2^{-m_1} + 2^{-6m} + 4 \cdot 2^{-m_1+2m} + 2m \cdot 2^{-m_1} \\ &< 2^{-m}.\end{aligned}$$

For the last step, notice that $m_1 = 3m + 3 + \lceil \log_2(\eta+3) \rceil > 3m + 4$, so

$$2(\eta+2)m \cdot 2^{-m_1} + 2m \cdot 2^{-m_1} = 2(\eta+3)m \cdot 2^{-m_1} \leq (\eta+3) \cdot 2^m 2^{-m_1} < \frac{1}{8} \cdot 2^{-m}$$

and

$$4 \cdot 2^{-m_1+2m} = 2^{-m_1+2m+2} < \frac{1}{4} \cdot 2^{-m}.$$

To obtain the more general guarantee, first suppose that $s \leq p$. In this case, define

$$\mathcal{A}_{RB}(1^p, 1^s, \alpha, U) \stackrel{\text{def}}{=} \mathcal{A}_{RB}(1^p, \alpha, U).$$

Its statistical distance from $\mathrm{Beta}_{R(2^{-p})}(\alpha, \alpha)$ is less than $2^{-p} \leq 2^{-s}$. In particular, this gives the claimed equality of the two interfaces when $s = p$.

It remains to consider $s > p$. Set $q = p + s + 3$ and write $Q_r(y) = \lfloor 2^r y \rceil 2^{-r}$. Define

$$\mathcal{A}_{RB}(1^p, 1^s, \alpha, U) \stackrel{\text{def}}{=} Q_p\left(\mathcal{A}_{RB}(1^q, \alpha, U)\right),$$

where the algorithm on the right is the original three-argument algorithm. If $X \sim \mathrm{Beta}(\alpha, \alpha)$, the first part of the proof and the data processing inequality give

$$\Delta\left(\mathcal{A}_{RB}(1^p, 1^s, \alpha, U), Q_p(Q_q(X))\right) < 2^{-q}.$$

Moreover, $Q_p(Q_q(X)) \neq Q_p(X)$ only if X is within 2^{-q-1} of a rounding boundary $(k+\frac{1}{2})2^{-p}$. Since the density of $\mathrm{Beta}(\alpha,\alpha)$ is nondecreasing on $(0,\frac{1}{2})$ and nonincreasing on $(\frac{1}{2},1)$, the same interval comparison used in $(*)$ above shows that

$$\Pr[Q_p(Q_q(X)) \neq Q_p(X)] \leq 4 \cdot 2^{-q-1+p} = 2^{p-q+1}.$$

Hence

$$\Delta\left(\mathcal{A}_{RB}(1^p, 1^s, \alpha, U), \mathrm{Beta}_{R(2^{-p})}(\alpha,\alpha)\right) < 2^{-q} + 2^{p-q+1} < 2^{-s}.$$

The running time and the number of random bits are polynomial in q, and hence in $p+s$.

H Technical lemmas on function properties

Lemma 7. *Let $d > 0$, $c = \frac{1}{\sqrt{9d}}$ and $g(x) = d\ln(1+cx)^3 - d(1+cx)^3 + d$ for $x \in (-\frac{1}{c}, +\infty)$. Then*

$$g(x) \le -\frac{1}{2}x^2.$$

Proof. Let

$$\begin{aligned} s(x) &= g(x) - (-\frac{1}{2}x^2) \\ &= \frac{1}{2}x^2 + d - d(1+cx)^3 + d\ln(1+cx)^3. \end{aligned}$$

Then

$$\begin{aligned} s'(x) &= x - 3cd(1+cx)^2 + \frac{3cd}{1+cx} \\ &= \frac{x(-3c^4dx^2 - 9c^3dx - 9c^2d + cx + 1)}{1+cx} \\ &= -\frac{c^2x^3}{3(1+cx)}. \qquad (\text{Since } c^2d = \frac{1}{9}) \end{aligned}$$

We have $s(0) = 0$ and $\begin{cases} s'(x) > 0, & x \in (-\frac{1}{c}, 0) \\ s'(x) \le 0, & x \in [0, +\infty) \end{cases}$, so $s(x) \le 0$ for $x \in (-\frac{1}{c}, +\infty)$, which means $g(x) \le -\frac{1}{2}x^2$ for $x \in (-\frac{1}{c}, +\infty)$.

Lemma 8. *Let $p(\alpha) = e^{\alpha - \frac{1}{3}}\sqrt{\alpha - \frac{1}{3}}\Gamma(\alpha) / \left(\sqrt{2\pi}(\alpha - \frac{1}{3})^\alpha\right)$, $\alpha \in \mathbb{N}^+$. Then $p(\alpha) > 0.95$.*

Proof. Let

$$\begin{aligned} r(\alpha) &= \frac{p(\alpha+1)}{p(\alpha)} \\ &= e\frac{\sqrt{\alpha + \frac{2}{3}}}{\sqrt{\alpha - \frac{1}{3}}}\alpha\frac{(\alpha - \frac{1}{3})^\alpha}{(\alpha + \frac{2}{3})^{\alpha+1}} \\ &= e\alpha\frac{(\alpha - \frac{1}{3})^{\alpha - \frac{1}{2}}}{(\alpha + \frac{2}{3})^{\alpha + \frac{1}{2}}}. \end{aligned}$$

Let

$$\begin{aligned} f(\alpha) &= \ln r(\alpha) \\ &= 1 + \ln\alpha + (\alpha - \frac{1}{2})\ln(\alpha - \frac{1}{3}) - (\alpha + \frac{1}{2})\ln(\alpha + \frac{2}{3}). \end{aligned}$$

Notice that $f(\alpha)$ is well-defined and differentiable on $[1,+\infty)$. Let $\beta=\alpha+\frac{2}{3}$, we have

$$\begin{aligned}
f'(\alpha) &= \frac{1}{\alpha}+\ln(\alpha-\frac{1}{3})+\frac{\alpha-\frac{1}{2}}{\alpha-\frac{1}{3}}-\ln(\alpha+\frac{2}{3})-\frac{\alpha+\frac{1}{2}}{\alpha+\frac{2}{3}}\\
&= \frac{1}{\alpha}+\ln(\beta-1)+\frac{\alpha-\frac{1}{2}}{\beta-1}-\ln\beta-\frac{\alpha+\frac{1}{2}}{\beta}\\
&= \ln(\frac{\beta-1}{\beta})+\frac{1}{\alpha}+\frac{\alpha-\frac{1}{2}}{\beta-1}-\frac{\alpha+\frac{1}{2}}{\beta}\\
&= \ln(1-\frac{1}{\beta})+\frac{1}{\alpha}-\frac{1}{6\beta(\beta-1)}\\
&= -\sum_{k=1}^{+\infty}\frac{1}{k\beta^k}+\frac{1}{\alpha}-\frac{1}{6\beta(\beta-1)} \qquad \text{(Taylor expansion)}\\
&< -\sum_{k=1}^{3}\frac{1}{k\beta^k}+\frac{1}{\alpha}-\frac{1}{6\beta(\beta-1)}\\
&= -\frac{2\alpha(\beta-1)+3\alpha\beta(\beta-1)+6\alpha\beta^2(\beta-1)-6\beta^3(\beta-1)+\alpha\beta^2}{6\alpha\beta^3(\beta-1)}\\
&= -\frac{\frac{1}{27}(3\alpha-4)^2}{6\alpha\beta^3(\beta-1)}\\
&\le 0.
\end{aligned}$$

This shows that $f(\alpha)$ decreases monotonically, so $r(\alpha)$ decreases monotonically.

We also have

$$\begin{aligned}
\lim_{\alpha\to+\infty} r(\alpha) &= \lim_{\alpha\to+\infty} e\frac{\alpha}{\alpha-\frac{1}{3}}\frac{(\alpha-\frac{1}{3})^{\alpha+\frac{1}{2}}}{(\alpha+\frac{2}{3})^{\alpha+\frac{1}{2}}}\\
&= e\lim_{\alpha\to+\infty}\left(\frac{\alpha-\frac{1}{3}}{\alpha+\frac{2}{3}}\right)^{\alpha+\frac{1}{2}}\\
&= e\lim_{\alpha\to+\infty}\left(1-\frac{1}{\alpha+\frac{2}{3}}\right)^{\alpha+\frac{1}{2}}\\
&= e\cdot e^{\lim_{\alpha\to+\infty}\ln\left(1-\frac{1}{\alpha+\frac{2}{3}}\right)(\alpha+\frac{1}{2})}\\
&= e\cdot e^{\lim_{\alpha\to+\infty}-\left(\frac{1}{\alpha+\frac{2}{3}}+O\left(\left(\frac{1}{\alpha+\frac{2}{3}}\right)^2\right)\right)(\alpha+\frac{1}{2})} \qquad \text{(Taylor expansion)}\\
&= e\cdot e^{-1}\\
&= 1.
\end{aligned}$$

Thus, $r(\alpha)>1$. This gives that $p(\alpha)$ increases monotonically. Then for all $\alpha\ge 1$,

$$p(\alpha)\ge p(1)=e^{\frac{2}{3}}\sqrt{\frac{2}{3}}/(\sqrt{2\pi}\cdot\frac{2}{3})\approx 0.95167>0.95.$$

Lemma 9. *Let $f(x,y)=\frac{x^3}{x^3+y^3}$, $x,y\in\mathbb{R}^+$. Given constants $\delta>0$, then*

$$\begin{aligned}
0<\frac{\partial f}{\partial x}&<\frac{1}{\delta},\\
-\frac{1}{\delta}<\frac{\partial f}{\partial y}&<0,
\end{aligned}$$

when $x > \delta$ or $y > \delta$.

Proof. Let $u = x^3, v = y^3$, then

$$\begin{aligned}\frac{\partial f}{\partial x} &= \frac{\partial f}{\partial u}\frac{\mathrm{d}u}{\mathrm{d}x} \\ &= \frac{v}{(u+v)^2} \cdot 3x^2 \\ &= \frac{3x^2y^3}{(x^3+y^3)^2} \\ &> 0. && \text{(Since } x, y \in \mathbb{R}^+\text{)}\end{aligned}$$

When $x > \delta$,

$$\begin{aligned}\frac{\partial f}{\partial x} =& \frac{3x^2y^3}{(x^3+y^3)^2} \\ &\leq \frac{3x^2y^3}{4x^3y^3} && \text{(Since } (A+B)^2 \geq 4AB\text{)} \\ &= \frac{3}{4x} \\ &< \frac{3}{4\delta} \\ &< \frac{1}{\delta}.\end{aligned}$$

When $y > \delta$, let $t = \frac{x}{y}$, then

$$\begin{aligned}\frac{\partial f}{\partial x} &= \frac{3x^2y^3}{(x^3+y^3)^2} \\ &= \frac{3(ty)^2y^3}{((ty)^3+y^3)^2} \\ &= \frac{3}{y} \cdot \left(\frac{t}{t^3+1}\right)^2 \\ &\leq \frac{3}{y} \cdot \frac{2^{\frac{4}{3}}}{9} && \text{(} \frac{t}{t^3+1} \text{ achieves maximum value } \frac{2^{\frac{4}{3}}}{9} \text{ at } t = 2^{-\frac{1}{3}} \text{ for } t > 0 \text{)} \\ &\leq \frac{2^{\frac{4}{3}}}{3\delta} \\ &< \frac{1}{\delta}.\end{aligned}$$

$f(x, y)$ can be represented as $f(x, y) = 1 - \frac{y^3}{x^3+y^3} = 1 - f(y, x)$, which shows the result for $\dfrac{\partial f}{\partial y}$.

Lemma 10. *Let $\alpha \geq 1$, $d = \alpha - \frac{1}{3}$, $c = \frac{1}{\sqrt{9d}}$, and $s(x) = \frac{1}{2}x^2 + d - d(1+cx)^3 + d\ln((1+cx)^3)$ for $x \in (-\frac{1}{c}, +\infty)$. Let $f(x) = e^{s(x)}$, then there exists a constant η (independent of α) such that*

$$|f'(x)| \leq \eta.$$

Proof. Same as in Lemma 7, We have $s'(x) = -\frac{c^2x^3}{3(1+cx)}$. Then

$$\begin{aligned} f'(x) &= e^{s(x)}s'(x) \\ &= e^{\frac{1}{2}x^2+d-d(1+cx)^3}(1+cx)^{3d} \cdot -\frac{c^2x^3}{3(1+cx)} \\ &= -\frac{c^2}{3}x^3(1+cx)^{3d-1}e^{\frac{1}{2}x^2+d-d(1+cx)^3}. \end{aligned} \tag{27}$$

By direct computation, we have

$$\begin{aligned} \lim_{x\to\left(-\frac{1}{c}\right)^+} f'(x) = 0, \\ \lim_{x\to+\infty} f'(x) = 0, \end{aligned}$$

and

$$\begin{cases} f'(x) > 0, & x \in (-\frac{1}{c}, 0) \\ f'(x) = 0, & x = 0 \\ f'(x) < 0, & x \in (0, +\infty). \end{cases}$$

Since $f'(x)$ is continuous and differentiable, we know that $f'(x)$ achieves its maximum value at some point in $(-\frac{1}{c}, 0)$ and its minimum value at some point in $(0, +\infty)$, both only when $f''(x) = 0$. Since

$$f''(x) = e^{s(x)}\left(s'(x)^2 + s''(x)\right),$$

$f''(x) = 0$ if and only if $s'(x)^2 + s''(x) = 0$. We have

$$\begin{aligned} & s'(x)^2 + s''(x) \\ &= \frac{c^4x^6}{9(1+cx)^2} - \frac{c^2x^2(2cx+3)}{3(1+cx)^2} \\ &= \frac{c^2x^2(c^2x^4 - 6cx - 9)}{9(cx+1)^2}. \end{aligned}$$

Thus, $f''(x) = 0$ if and only if $c^2x^4 - 6cx - 9 = 0$, for both $x \in (-\frac{1}{c}, 0)$ and $x \in (0, +\infty)$

Let $w(x) = c^2x^4 - 6cx - 9$, then

$$w'(x) = 4c^2x^3 - 6c$$

has only one real root $x_0 = \sqrt[3]{\frac{3}{2c}}$, and we have

$$\begin{aligned} w(-\frac{1}{c}) = \frac{1}{c^2} - 3 = 9d - 3 > 0, \\ w(0) = -9 < 0, \\ \lim_{x\to+\infty} w(x) = +\infty. \end{aligned}$$

These show that $w(x)$ has exactly one real root in $(-\frac{1}{c}, 0)$ and one real root in $(0, +\infty)$, denote them as x_1 and x_2 respectively. Then they are the points where $f'(x)$ achieves its maximum and minimum values, respectively, i.e. $f'(x_2) \le f'(x) \le f'(x_1)$.

This already shows that for each fixed c, there exists $\eta_1(c) = f'(x_1), \eta_2(c) = f'(x_2)$ as the upper and lower bounds of $f'(x)$, respectively. Now we consider the bound over all possible c. Since $\alpha \ge 1$, we have $c = \frac{1}{\sqrt{9\alpha-3}} \in (0, \sqrt{6}]$.

First, we consider the area $x \in (-\frac{1}{c}, 0)$. Let $u = cx^2$, then $x = -\sqrt{\frac{u}{c}}$. $c^2x^4 - 6cx - 9 = 0$ can be represented as

$$u^2 + 6\sqrt{cu} - 9 = 0, \tag{28}$$

which has a positive root u_1. When $c = 0$ the root $u_1 = 3$, and $p(u, c) = u^2 + 6\sqrt{cu} - 9$ is a continuous function, so we have

$$\lim_{c\to 0^+} u_1 = 3.$$

We also represent $f'(x)$ in terms of u by putting $x = -\sqrt{\frac{u}{c}}$ into eq. (27):

$$h_1(u) = \frac{1}{3}\sqrt{cu^3}(1 - \sqrt{cu})^{3d-1} e^{\frac{u}{2c} + d - d(1-\sqrt{cu})^3}.$$

Thus, we have

$$\begin{aligned}
\lim_{c\to 0^+} \eta_1(c) &= \lim_{c\to 0^+} h_1(u_1) \\
&= \lim_{c\to 0^+} h_1(3) \\
&= \lim_{c\to 0^+} \frac{1}{3}\sqrt{3^3 c}(1 - \sqrt{3c})^{3d-1} e^{\frac{3}{2c} + d - d(1-\sqrt{3c})^3} \\
&= \lim_{t\to 0^+} t(1-t)^{\frac{3}{t^4}-1} e^{\frac{9}{2t^2} + \frac{1}{t^4} - \frac{1}{t^4}(1-t)^3} \qquad (\text{Let } t = \sqrt{3c} = \frac{1}{\sqrt[4]{d}}) \\
&= \lim_{t\to 0^+} t(1-t)^{\frac{3}{t^4}-1} e^{\frac{3}{t^3} + \frac{3}{2t^2} + \frac{1}{t}} \\
&= \lim_{t\to 0^+} te^{\left(\frac{3}{t^4}-1\right)\ln(1-t) + \frac{3}{t^3} + \frac{3}{2t^2} + \frac{1}{t}} \\
&= \lim_{t\to 0^+} te^{-\left(\frac{3}{t^4}-1\right)\left(\sum_{i=1}^{4} \frac{t^i}{i} + O(t^5)\right) + \frac{3}{t^3} + \frac{3}{2t^2} + \frac{1}{t}} \qquad (\text{Taylor expansion}) \\
&= \lim_{t\to 0^+} te^{-\left(\frac{3}{t^3} + \frac{3}{2t^2} + \frac{1}{t} + \frac{3}{4} + O(t)\right) + \frac{3}{t^3} + \frac{3}{2t^2} + \frac{1}{t}} \\
&= \lim_{t\to 0^+} te^{-\frac{3}{4} + O(t)} \\
&= 0.
\end{aligned}$$

Then we consider the area $x \in (0, +\infty)$. Let $u = cx^2$, then $x = \sqrt{\frac{u}{c}}$. $c^2x^4 - 6cx - 9 = 0$ can be represented as

$$u^2 - 6\sqrt{cu} - 9 = 0, \tag{29}$$

which has root u_2. Similarly, we can get

$$\lim_{c\to 0^+} u_2 = 3,$$

represent $f'(x)$ as

$$h_2(u) = \frac{1}{3}\sqrt{cu^3}(1 + \sqrt{cu})^{3d-1} e^{\frac{u}{2c} + d - d(1+\sqrt{cu})^3},$$

and get

$$\lim_{c\to 0^+} \eta_2(c) = \lim_{c\to 0^+} h_2(3) = 0.$$

Since $\lim_{c\to 0^+} \eta_1(c) = \lim_{c\to 0^+} \eta_2(c) = 0$, there exists a constant c_0 such that for all $c \in (0, c_0)$, we have $|\eta_1(c)|, |\eta_2(c)| < 1$. Then let the constant

$$\eta = \max\{1, \max_{c\in[c_0,\sqrt{6}]} |\eta_1(c)|, \max_{c\in[c_0,\sqrt{6}]} |\eta_2(c)|\}.$$

We have $|\eta_1(c)|, |\eta_2(c)| \le \eta$ for all $c \in (0, \sqrt{6}]$, which means $|f'(x)| \le \eta$ always holds.