Sum-check protocol for approximate computations

Dor Bitan* Zachary DeStefano† Shafi Goldwasser‡
Yuval Ishai§ Yael Tauman Kalai¶ Justin Thaler‖

Abstract

Motivated by the mismatch between floating-point arithmetic, which is intrinsically approximate, and verifiable computing protocols for exact computations, we develop a generalization of the classical sum-check protocol. Our generalization proves claims of the form $\sum_{x\in\{0,1\}^v} g(x) \approx H$, where g is a low-degree v-variate polynomial over an integral domain $\mathbb{U}$. The verifier performs its check in each round of the protocol using a tunable error parameter δ. If Δ is the error in the prover's initial claim, then the soundness error of our protocols degrades gracefully with δ/Δ. In other words, if the initial error Δ is large relative to δ, then the soundness error is small, meaning the verifier is very likely to reject.

Unlike the classical sum-check protocol, which is fundamentally algebraic, our generalization exploits the *metric* structure of low-degree polynomials. The protocol can be instantiated over various domains, but is most natural over the complex numbers, where the analysis draws on the behavior of polynomials over the unit circle. We also analyze the protocol under the Fiat-Shamir transform, revealing a new "intermediate security" phenomenon that appears intrinsic to approximation.

Prior work on verifiable computing for numerical tasks typically verifies that a prover *exactly* executed a computation that only *approximates* the desired function. In contrast, our protocols treat approximation as a first-class citizen: the verifier's checks are relaxed to accept prover messages that are only approximately consistent with the claimed result. This establishes the first black-box feasibility result for approximate arithmetic proof systems: the protocol compiler is independent of how arithmetic operations are implemented, requiring only that they satisfy error bounds. This opens a path to verifying approximate computations while sidestepping much of the prover overhead imposed by existing techniques that require encoding real-valued data into finite field arithmetic.

*UC Berkeley — dorbi@post.bgu.ac.il
†NYU Department of Computer Science, Courant Institute — zd@nyu.edu
‡UC Berkeley — shafi.goldwasser@gmail.com
§Technion and AWS — yuval.ishai@gmail.com. This work is not associated with Amazon.
¶MIT — yaelism@gmail.com
‖a16z crypto research and Georgetown University — justin.r.thaler@gmail.com

Contents

1 Introduction **1**
- 1.1 Our contributions: an approximate sum-check protocol . . . 2
- 1.2 Related work . . . 4

2 Preliminaries **5**
- 2.1 Notation . . . 5
- 2.2 Integral domains and metrics . . . 5
- 2.3 Interactive proofs and the classical sum-check protocol . . . 6
- 2.4 Round-by-round soundness and the Fiat-Shamir transformation . . . 6
- 2.5 Remez theory and complex analysis . . . 8

3 Approximate arithmetic computations and the approximate sum-check protocol over integral domains **9**
- 3.1 The approximate sum-check protocol . . . 9
- 3.2 Proof of Theorem 3.1 . . . 12
  - 3.2.1 Completeness . . . 12
  - 3.2.2 Soundness . . . 13
  - 3.2.3 Round-by-round soundness . . . 15
- 3.3 Relation to the classical sum-check protocol . . . 16

4 The approximate sum-check protocol implemented over $\mathbb{R}$ and $\mathbb{C}$ **16**
- 4.1 Approximate Schwartz–Zippel lemmas over $\mathbb{R}$ and $\mathbb{C}$ . . . 16
- 4.2 Application to $\mathbb{R}$ . . . 18
  - 4.2.1 Bounding c_d . . . 20
  - 4.2.2 Round-by-round soundness . . . 20
  - 4.2.3 A simple soundness corollary . . . 20
- 4.3 Application to $\mathbb{C}$ . . . 21
  - 4.3.1 Bounding c_d . . . 21
  - 4.3.2 Round-by-round soundness . . . 22

5 Application: Approximate fully linear IOPs **22**

6 Detailed comparison with alternative approaches **23**

7 Discussion **25**

8 Acknowledgments **25**

A Numerical soundness analysis **29**

B Explicit malicious prover strategy **29**

C Simple soundness analysis **32**

1 Introduction

The problem of verifying the correctness of computation (VC) is one of the most fundamental problems in theoretical computer science and lies at the heart of celebrated results and open problems, including IP=PSPACE [LFKN92, Sha92], the PCP Theorem [AS92, ALM+92], and the P versus NP problem. There is a vast literature on the topic,[1] most of which focuses on computations over finite fields or other finite domains. These protocols ensure that an untrusted prover correctly executed a computation step-by-step, where each step consists of an exact operation over the finite domain used internally by the protocol.

This focus on exact computation creates a fundamental mismatch with numerical workloads. Scientific computation, data analysis, and machine learning perform computations on real-valued data, where exact arithmetic is neither feasible nor desired. Because representing exact intermediate results over the reals requires precision that grows with computational depth, practical systems universally rely on approximate arithmetic—typically floating-point operations that trade exactness for speed. The resulting computations are inherently approximate: rounding errors accumulate across operations, leading to deviations from any hypothetical exact result.

Existing verification protocols prove exact execution—precisely what scientific computing neither achieves nor requires. A user running a fluid dynamics simulation does not care whether the prover executed each floating-point operation exactly as specified; they care whether the simulation output is accurate to within physically meaningful tolerances. Numerical analysis provides formal guarantees about accuracy and convergence of approximate algorithms, without any assumptions or guarantees on exact execution order or bit-perfect reproduction of intermediate values.

Beyond the fundamental approximation inherent in floating-point arithmetic, approximate computation introduces behavior absent from exact computation. Reordering a sequence of additions can yield different results due to the non-associativity of floating-point arithmetic. Modern hardware amplifies this issue: GPU programmers typically cannot control the exact order in which parallelized floating-point operations execute, making computations non-deterministic even when running the same program twice on identical hardware.[2] The performance of supercomputers is measured in FLOPS (floating-point operations per second), a reflection that approximation is foundational, not a defect.

Despite this mismatch, many works have begun to adapt verifiable computing to numerical tasks. The prevailing approach encodes approximate real-number computations as exact arithmetic over finite fields using fixed-point or quantized representations. While this enables the use of established SNARK frameworks, it introduces substantial overhead: the proved computation often differs significantly from the originally intended computation due to quantization, or alternatively, significant restrictions must be placed on the types of computations that can be verified. These systems verify that a prover exactly executed a computation that only approximates the desired function, an awkward indirection that inherits neither the flexibility of approximate computing nor the certainty of exact verification.

The sum-check protocol and its limitations. At the heart of many verification protocols lies the sum-check protocol [LFKN92], an interactive proof for verifying the correctness of a computation of the form $\sum_{x\in\{0,1\}^v} g(x) = H$ where g is a low-degree multivariate polynomial over any *integral domain*.[3] Although claims of this form may initially appear to have limited applicability, sum-check has proven remarkably powerful: it can verify any computation expressible as a low-depth arithmetic circuit [GKR08], and when combined with cryptographic commitments, it forms the foundation of protocols that verify arbitrary computations under standard cryptographic assumptions. This protocol and its descendants now power production systems in blockchain ecosystems including Ethereum.

However, sum-check—like the broader VC ecosystem it enables—assumes exact arithmetic. The verifier's checks demand precise polynomial equality. This assumption permeates the soundness analysis, which relies fundamentally on algebraic properties—specifically, the zero-sets of polynomials and error-correcting codes—rather than on any notion of approximation.

In this work, we take a fundamentally different approach. Rather than forcing approximate computations

[1] For recent surveys, see [Tha22, CY24, Tha25]; even these surveys cover only portions of the area.

[2] See [HL25] for a recent overview of errors and non-associativity in floating point operations, and other sources of non-determinism in machine learning workloads.

[3] An integral domain (Definition 1) is an algebraic structure between fields and rings. Modern VC results are often stated over fields, even when they apply more generally.

into exact algebraic structures, we generalize the sum-check protocol itself to operate natively with approximation. Our protocol proves claims of the form $\sum_{x \in \{0,1\}^v} g(x) \approx H$, where both the prover's messages and the verifier's checks tolerate bounded approximation error. The soundness guarantee degrades gracefully: the verifier is convinced with high probability that the claimed sum is accurate up to a tunable error bound Δ. Critically, both prover and verifier can execute their algorithms using purely approximate arithmetic (e.g., floating-point operations) at a precision determined by the desired accuracy Δ and security parameter.

This eliminates the need for arithmetization. We avoid encoding the computation into finite field operations and instead work directly with the original approximate computation. Our soundness analysis departs entirely from algebraic techniques: rather than relying on error-correcting codes and polynomial zero-sets, we instead control the *sublevel sets* of low-degree polynomials–a *metric* notion rooted in approximation theory and complex analysis. This perspective yields what we believe is the first "black-box feasibility" result for verifying approximate arithmetic: the protocol compiler is completely independent of *how* the prover and the verifier implement approximate arithmetic. The protocol itself simply has the prover and verifier compute certain expressions to accuracy within a specified error bound (see Section 1.1 for elaboration).

1.1 Our contributions: an approximate sum-check protocol

First, we construct and analyze a generalized *sum-check protocol for approximate computations* (Figure 2) which applies to approximate arithmetic over any integral domain equipped with a suitable metric. We prove the completeness, soundness, and round-by-round soundness of this protocol (Theorem 3.1). This protocol can be carried out exclusively using approximate arithmetic. All of the verifier's checks are performed approximately, and likewise the prover, using purely approximate arithmetic, is able to compute messages that satisfy these checks. See Section 3.

Second, we describe and analyze how this protocol can be instantiated for the common settings of approximate arithmetic over $\mathbb{R}$ (Theorem 4.4) and $\mathbb{C}$ (Theorem 4.7). To do this, we construct approximate analogs of the Factor Theorem (commonly known as the univariate case of the Schwartz–Zippel lemma) over $\mathbb{R}$ (Theorem 4.2) and $\mathbb{C}$ (Theorem 4.3). This result applies classical and modern results from complex analysis and approximation theory. See Section 4.

Third, we analyze the Fiat–Shamir transform of our protocol and uncover a new form of round-by-round (RBR) soundness that is intermediate between the classical sum-check and constant-round protocols (Theorems 4.5 and 4.8). This intermediate behavior reflects the ability of a cheating prover to make gradual progress, round by round, toward making an initially large error appear smaller over successive rounds. This is the first non-trivial protocol which we are aware of with this intermediate RBR soundness. See Section 4.

Additionally, we show a practical application of our approximate sum-check protocol in a fully linear interactive oracle proof for the inner product of vectors over $\mathbb{R}$ that are distributed or secret-shared. We use this as an example to make our results explicit and discuss concrete parameters. See Section 5. More generally, we intend this paper as a showcase of tools from other areas of mathematics (e.g. approximation theory, complex analysis), which previously have not seen significant use in cryptography.

Our approach. We generalize the sum–check protocol so that it reasons about approximation directly. In each round the verifier replaces exact equalities from the classical sum-check protocol with comparisons up to a prescribed tolerance, so messages need only be consistent within that margin. This eliminates arithmetization: rather than encoding the computation over a finite field, the prover and verifier work directly with the original approximate computation using ordinary approximate arithmetic at an application–chosen precision.

This approach yields a black–box feasibility result: the protocol compiler is independent of how arithmetic is implemented; the protocol merely requires the parties to evaluate specified expressions to within stated error bounds. Section 3 formalizes the protocol and its guarantees over a general metric domain; Section 4 instantiates the framework over the reals and the complex numbers and derives the resulting soundness bounds.

Analytic methods replacing algebraic techniques. Our soundness analysis departs fundamentally from the classical sum-check protocol, which relies on algebraic properties—specifically the Factor Theorem that bounds the size of polynomial zero-sets. Instead, we control the *sublevel sets* of low-degree polynomials (sets of the form $\{x : |p(x)| \leq \delta\}$), using tools from approximation theory and complex analysis.

The key challenge is understanding how errors can propagate across rounds. In the classical setting, a cheating prover either succeeds immediately (if two distinct polynomials happen to agree at the random challenge) or fails—there's no middle ground. In our approximate setting, however, a more subtle attack becomes possible: the prover can make gradual progress round-by-round, slowly reducing an initially large error Δ down to the acceptance threshold δ.

Our soundness analysis addresses this by considering all rounds holistically rather than applying a union bound over individual rounds. The key idea is that Remez-style inequalities let us conclude that substantial error contraction is unlikely across the protocol. For the error to contract significantly in any given round, there must be a polynomial—specifically, the difference between the honest polynomial q_k that the prover should have sent and the dishonest polynomial p_k that was actually sent—that has a large magnitude at one or more points in the challenge space S but a small magnitude at the specific random challenge r_k chosen by the verifier. Our Remez-style lemmas (Theorems 4.2 and 4.3) bound the probability that such "unlucky" challenges occur, establishing that the probability this happens repeatedly in a way that reduces the initial error Δ down to δ (whether gradually over many rounds or suddenly in one or two rounds) is vanishingly small.

This yields a soundness error of roughly

$$\frac{vd}{|S|} + 2^v \sqrt[c_d]{\frac{v\delta}{\Delta}},$$

where v and d are the number of variables and the degree (respectively) of the polynomial g to which the approximate sum-check applies, $c_d \approx 2d$ for $\mathbb{C}$ and $c_d \approx 3d$ for $\mathbb{R}$. The first term, $\frac{vd}{|S|}$, is inherited from the classical sum-check protocol, while the second captures the probability of sufficient error contraction through our approximation-theoretic analysis. A simpler round-by-round analysis (Appendix C) gives weaker bounds by treating each round in isolation, but our tighter analysis in Section 4.2 uses a novel inductive argument that tracks the cumulative distribution of error reductions across all rounds.

We extend this soundness analysis to round-by-round (RBR) soundness via another novel inductive proof. This leads to two surprising results. First, to our knowledge, all prior non-contrived r-round interactive protocols with soundness ϵ either have RBR soundness roughly ϵ or $\epsilon^{1/r}$; however, our approximate sum-check protocol, when instantiated over $\mathbb{R}$ and $\mathbb{C}$ has RBR soundness roughly $\epsilon^{3/r}$ and $\epsilon^{2/r}$ respectively. Second, this unusual RBR soundness is tight. We show this by demonstrating a prover strategy which closely matches this bound (Appendix B). This malicious prover makes gradual progress, round by round, toward making an initially large error appear smaller over successive rounds. As expected, this attack differs significantly from the optimal malicious strategy for the classical sum-check protocol.

To the best of our knowledge, this is the first use of such approximation-theoretic tools in the analysis of interactive proofs. Beyond serving our immediate purposes, it highlights that the robustness of sum-check does not fundamentally rely on algebraic vanishing (that is, on the geometry of the zero-set of polynomials) or error-correcting codes, but can also emerge from purely analytic control over low-degree polynomials.

To illustrate our soundness results, consider the following simplified theorem describing the soundness of an approximate sum-check protocol over $\mathbb{R}$.

Theorem 1.1 (Informal version of Theorem 4.4)**.** *Consider the classical sum-check protocol, depicted in Figure 1, instantiated over $\mathbb{R}$. Suppose that $B = \{0, 1\}$ and S is a set of at least $4vd$ equispaced points over the interval $[0, 1]$. Additionally, suppose that all operations by the verifier are performed using approximate arithmetic with absolute error ε. In this model, the prover and verifier exchange approximate real numbers and the verifier checks approximate rather than exact equality (described in detail for each round in Figure 2). Then, in the resulting protocol the verifier is convinced that the prover's claim about the sum has error at most Δ with probability at least $1/2$, where $\Delta \in O\left(2^{O(vd)} vd \cdot \varepsilon\right)$. In other words, if the error in the prover's claim Δ is larger than this bound, then the verifier rejects with probability at least $1/2$.*

A simpler soundness analysis, which directly adapts the proof of soundness for the classical sum-check protocol by using an approximate variant of the Schwartz–Zippel lemma over $\mathbb{R}$ (Theorem 4.2), only allows for proving the claim with $\Delta \in O((4v)^{O(vd)} d \cdot \varepsilon)$ and has a concretely larger constants hidden by the asymptotic notation. The tighter bound is based on a careful inductive argument that employs different results from

approximation theory. The tight soundness analysis follows from our results in Section 4.2. The simpler proof of the above weaker soundness bound is presented in Appendix C.

Applications. As discussed above, the sum-check protocol is a crucial building block in protocols for verifiable computation, including the GKR protocol [GKR08] and many follow-up works. We expect our approximate sum-check protocol and its analysis to lead to asymptotic and concrete improvements in this space, and leave a detailed exploration of this direction to future work. In Section 5, we present a different and more direct application of approximate sum-check that implies low-communication proofs of simple arithmetic statements on distributed or secret-shared data, captured by the notion of a *fully linear IOP* [BBC+19].

A black-box compiler perspective. Beyond enabling more efficient verification of approximate computations, our results establish a qualitatively new type of feasibility result: the possibility of *representation-independent* verification of approximate computation.

Traditional approaches to verifying approximate computations require non-black-box access to the computation's representation. For instance, quantization-based methods must know the bit-representation of inputs to embed the computation in a finite field whose size depends on the required precision. The choice of field, the (approximate) encoding of real values, and the arithmetization strategy all depend intimately on the precision requirements and the structure of the computation being verified.

In contrast, our protocol can be viewed as a black-box compiler with the following signature:

Input: A statement of the form $\sum_{x\in\{0,1\}^v} g(x) \approx H$, represented by an arithmetic circuit for g

Output: A tuple of arithmetic circuits implementing:

- Prover's next-message functions (size $\text{poly}(S, 2^v)$ each)
- Verifier's next-message functions (size $\text{poly}(S)$ each)

Crucially, the compiler is *representation-oblivious*: it does not depend on the precision level ε, the desired accuracy Δ, or even the underlying integral domain $\mathbb{U}$. The compiler outputs abstract arithmetic circuits using operations $\{+, -, \times\}$ without specifying how these operations are implemented or represented.

Our completeness and soundness analysis (Theorems 3.1, 4.4, and 4.7) guarantees that these circuits achieve the claimed security properties whenever the approximation error of the prover's and verifier's arithmetic operations is sufficiently small. The required precision follows from the parameters $(d, v, \Delta, \varepsilon)$, but the circuits themselves are independent of these choices. An implementer can select any representation (floating-point, fixed-point, interval arithmetic, symbolic computation) and any precision level, and the protocol's correctness follows from our analysis.

This is fundamentally different from quantization-based approaches, which produce *different* finite-field computations for different precision requirements. Our compiler is black-box in the same sense that classical upper and lower bounds for arithmetic circuits are black-box: the analysis does not "look inside" the representation of field elements, only at the circuit structure.

This black-box property has practical implications. A single implementation of the prover and verifier circuits can be instantiated at different precision levels for different applications without modifying the protocol structure. Most importantly, reasoning about security and correctness cleanly separates from reasoning about numerical precision—the former is handled by our protocol analysis, the latter by standard numerical analysis of approximate arithmetic.

1.2 Related work

There is a growing body of work which explores verifiable computation protocols for approximate numerical computations, particularly in the context of machine learning (an area which is broadly termed zkML) [KDZ24, Lab24, GGPZ17]. Most existing approaches operate by encoding approximate real-valued operations as exact arithmetic over finite fields, using fixed-point or quantized representations. While this enables the use of established SNARK frameworks, it introduces a major overhead which stems from the fact that the proved computation often differs significantly from the originally intended computation due to quantization, or alternatively significantly restricts the types of computations. While existing systems pay

heavily for quantization to operate over finite fields, our work generalizes the sum-check protocol itself to support approximation *natively*, without encoding real-valued computations into exact algebraic structures.

Recent work has begun to explore other strategies for handling approximation in proof systems. These include protocols over characteristic-zero domains such as $\mathbb{Z}$ or $\mathbb{Q}$ [CHA24, GWHD25], as well as exact arithmetic over $\mathbb{R}$ or $\mathbb{C}$ [V$^+$22]. Our work departs from all of these directions. Rather than verifying exact statements about approximate computations, we propose an interactive proof protocol that directly incorporates approximation into its soundness condition. A detailed comparison with these alternative approaches is deferred to Section 6.

While our focus is on interactive proofs for approximate computation, there is also a loosely related body of work on property testing of real-valued functions. This line of research [Rub91, REK96, KMS03, KMS99, ABCG93, KMS02, Mag00, GLR$^+$91, ABF$^+$23] studies how to efficiently test whether a real-valued function approximately satisfies a desired algebraic or analytic property. In particular, Chebyshev polynomials, which arise in our soundness analysis, also appear in the soundness analysis of the low-degree test for real-valued functions in [ABF$^+$23]. Whereas these works share our focus on reasoning about both numerical precision and approximate correctness, they do not address our setting of proof systems with possibly dishonest provers.

2 Preliminaries

Our work draws on tools from complex analysis, complexity theory, and cryptography. Here we introduce unified notation and provide the minimal background from each area necessary to understand and contextualize our results.

2.1 Notation

We use $\mathbb{U}$ to denote an arbitrary integral domain. The real numbers are $\mathbb{R}$, the complex numbers are $\mathbb{C}$, and, for a prime q, $\mathbb{F}_q$ is the finite field of size q.

When working with the complex numbers, $\mathbb{T} = \{x \in \mathbb{C} : |x| = 1\}$ denotes the complex unit circle, $\mathbb{D} = \{x \in \mathbb{C} : |x| \leq 1\}$ denotes the complex unit disk, $\omega_n = \exp(2\pi i/n)$ denotes a primitive n'th root of unity, and $\Omega_n = \{\omega_n^j : 0 \leq j < n\}$ denotes the full set of n'th roots of unity.

We use the following different notions for the size of a set S. For finite S, we write $\#(S)$ for the *cardinality*, the number of elements, of S. If $S \subset \mathbb{R}$ is a measurable subset of a measurable set J, we use $\lambda_J(S)$ to denote the *normalized Lebesgue measure* of S over J. That is $\lambda_J(S) = \frac{\text{Lebesgue measure of } S}{\text{Lebesgue measure of } J}$ where the Lebesgue measure captures the intuitive notion of the sum of the length of the intervals that comprise a set. If $S \subseteq \mathbb{T}$ and Lebesgue measurable, we use $\mu(S)$ to denote its *normalized arc-length measure* over $\mathbb{T}$. Informally, if S is a set of arcs on the unit circle, $\mu(S)$ is the sum of the lengths of these arcs, divided by 2π. Note that we will exclusively apply $\#$, λ, and μ for sets which satisfy the required properties for them to be well-defined. Additionally, for a function f and a set S, we define the *uniform norm* as $\|f\|_S := \sup_{x \in S} |f(x)|$. That is maximum *magnitude* of f when evaluated over all points in S.

To denote sampling r uniformly from a set S, we write $r \sim U(S)$ and use $U(0,1)$ as a shorthand for the uniform distribution over the interval $[0,1] \in \mathbb{R}$.

2.2 Integral domains and metrics

Definition 1 (Integral domain)**.** An integral domain is a set $\mathbb{U}$ equipped with two binary operations $+$ and $\times$ such that $(\mathbb{U}, +, \times)$ forms a commutative ring with unity and has no zero divisors. It is common to call $\mathbb{U}$ itself an integral domain and leave the operations implicit.

Common examples of integral domains include the integers $\mathbb{Z}$, the rationals $\mathbb{Q}$, the reals $\mathbb{R}$, the complex numbers $\mathbb{C}$, and finite fields (e.g., $\mathbb{F}_q$ for a prime q).

Definition 2 (Metric)**.** A metric on a set S is a function $D : S \times S \to \mathbb{R}_{\geq 0}$ such that for every $x, y, z \in S$,

- $D(x, y) = 0$ if and only if $x = y$,
- $D(x, y) = D(y, x)$, and
- $D(x, z) \leq D(x, y) + D(y, z)$ (This last property is known as the triangle inequality).

There are two common metrics, discrete and absolute value. The discrete metric is explicitly defined as

$$D(x,y) = \begin{cases} 0, & x = y \\ 1, & x \neq y \end{cases}.$$

The absolute value metric over $\mathbb{R}$ and $\mathbb{C}$ is defined as $D(x,y) = |x-y|$. Both of these trivially satisfy the properties of a metric.

Distances between polynomials. In order to analyze the soundness of our protocols, we need to define a notion of distance not only between two points, but also between two polynomials p, q. In our main applications, namely over the integral domains $\mathbb{R}$ or $\mathbb{C}$, we use the distance function $D(p,q) = \sup_{x \in I} |p(x) - q(x)|$ for a specified subdomain I. That is, for our work, the most relevant notion of distance between polynomials is their *maximum* pointwise difference over an evaluation domain I. This choice of distance function appears formally in Equation (12) in Section 4.

2.3 Interactive proofs and the classical sum-check protocol

Public-coin interactive proofs. A public-coin interactive proof is a protocol between a prover and a verifier in which the verifier's messages consist of uniformly random strings. The prover responds to each such message, and after some number of rounds (which may grow with the size of the verifier's input), the verifier decides whether to accept or reject.

A protocol has perfect completeness if the verifier always accepts when interacting with an honest prover on a true statement. It has soundness error ϵ if, for any false statement and any (possibly cheating) prover, the verifier accepts with probability at most ϵ.

The sum-check protocol. We now briefly recall the classical sum-check protocol of Lund, Fortnow, Karloff, and Nisan [LFKN92] in Figure 1, which serves as the starting point for our work. The classical sum-check protocol is an interactive proof that allows a prover to convince a verifier that

$$H = \sum_{(b_1,\ldots,b_v) \in B^v} g(b_1, \ldots, b_v)$$

for a v-variate polynomial g over $\mathbb{F}_q$ and a finite set $B \subset \mathbb{F}_q$ (in most applications of the sum-check protocol $B = \{0,1\}$).

Throughout, we refer to this original protocol as the *classical sum-check protocol*. We refer to our approximate version, which relaxes the verifier's checks to tolerate bounded approximation errors, as the *approximate sum-check protocol*.

Theorem 2.1 (Completeness and soundness of the classical sum-check protocol [LFKN92])**.** *Let $g \colon \mathbb{U}^v \to \mathbb{U}$ be a v-variate polynomial such that the degree of g in each variable is at most d and let $B, S \subseteq \mathbb{U}$ be finite sets. Then the classical sum-check protocol (Figure 1), when applied to g, satisfies perfect completeness and has soundness error at most $vd/|S|$.*

The standard proof of this theorem relies crucially on the Factor Theorem: if $p \neq q$ are distinct degree-d univariate polynomials, they can agree on at most d points. This algebraic fact underlies the $d/|S|$ per-round error probability that leads to the $vd/|S|$ overall soundness error via a union bound across rounds.

2.4 Round-by-round soundness and the Fiat-Shamir transformation

The Fiat-Shamir transform [FS87] is a standard technique for transforming public-coin interactive proofs into non-interactive protocols. Roughly speaking, each random verifier challenge in the interactive protocol is replaced in the transformed protocol with a cryptographic hash of the prover's messages up to that point in the protocol. The security of the transformed proof is deeply linked to a notion referred to as the round-by-round soundness error of the interactive protocol, defined below.

Definition 3 (Round-by-round soundness)**.** A public-coin interactive proof $\Pi = (P, V)$ for a language $\mathcal{L}$ is said to have round-by-round soundness error ϵ_{RBR} [CCH+19, Hol19] if there exists a "doomed set" of partial transcripts $\mathcal{D}$ such that:

Setup. The prover and verifier agree on g, a v-variate polynomial over $\mathbb{U}$ of degree at most d in each variable and $B, S \subseteq \mathbb{U}$, finite sets (in most applications of the sum-check protocol $B = \{0, 1\}$ and $\mathbb{U}$ is finite so $S = \mathbb{U}$). If any of the verifier's checks fail during the protocol, it immediately rejects.

Round 1. The prover sends the verifier H claimed to equal

$$\sum_{(b_1,\ldots,b_v)\in B^v} g(b_1, \ldots, b_v)$$

and a polynomial p_1 claimed to equal

$$\sum_{(b_2,\ldots,b_v)\in B^{v-1}} g(x, b_2, \ldots, b_v).$$

The verifier checks that p_1 is a univariate polynomial of degree $\leq d$ and that

$$H = \sum_{b_1\in B} p_1(b_1),$$

and subsequently sends a random challenge $r_1 \sim U(S)$ to the prover.

Round $k \in \{2, \ldots, v-1\}$**.** The prover sends the verifier a polynomial p_k claimed to equal

$$\sum_{(b_{k+1},\ldots,b_v)\in B^{v-k}} g(r_1, \ldots, r_{k-1}, x, b_{k+1}, \ldots, b_v).$$

The verifier checks that p_k is a univariate polynomial of degree $\leq d$ and that

$$p_{k-1}(r_{k-1}) = \sum_{b_k\in B} p_k(b_k),$$

and subsequently sends a random challenge $r_k \sim U(S)$ to the prover.

Round v**.** The prover sends the verifier a polynomial p_v claimed to equal

$$g(r_1, \ldots, r_{v-1}, x).$$

The verifier checks that p_v is a univariate polynomial of degree $\leq d$ and that

$$p_{v-1}(r_{v-1}) = \sum_{b_v\in B} p_v(b_v),$$

and draws a random challenge $r_v \sim U(S)$ to check that

$$p_v(r_v) = g(r_1, \ldots, r_v).$$

If this final check passes, the verifier accepts the prover's claim.

Figure 1: A modern presentation of the classical sum-check protocol of Lund, Fortnow, Karloff, and Nisan [LFKN92].

1. If $x \notin \mathcal{L}$, then $(x, \emptyset) \in \mathcal{D}$.

2. For every partial transcript τ, prover message α, and subsequent verifier message β, if $(x, \tau) \in \mathcal{D}$, then

$$\Pr_{\beta}[(x, \tau|\alpha|\beta) \notin \mathcal{D}] \leq \epsilon_{RBR}.$$

3. For every complete transcript τ, if $(x, \tau) \in \mathcal{D}$, then $V(x, \tau) = \text{reject}$.

Theorem 2.2 (Follows from [BSCS16, CCH+19, BGTZ25]). *Let Π be a v-round public coin interactive proof and $\text{FS}[\Pi]$ be the non-interactive protocol obtained by applying the Fiat-Shamir transformation to Π in the Random Oracle (RO) Model. If Π has round-by-round soundness ϵ_{RBR}, Q is an upper bound on the number of queries a malicious prover can make to the random oracle, and κ is the number of bits output by the random oracle whenever it is evaluated, then $\text{FS}[\Pi]$ has soundness error at most $O(Q \cdot \epsilon_{RBR} + Q^2 \cdot 2^{-\kappa})$.*

The $Q^2 \cdot 2^{-\kappa}$ term is the probability of finding a collision in the random oracle (i.e., two inputs to the random oracle that produce the same output). Meanwhile, the term $Q \cdot \epsilon_{RBR}$ is the success probability of a standard "grinding attack" on Fiat-Shamir'ed protocols. This attack performs a linear search over prover messages, attempting to find one that is "lucky for the attacker" in the sense that it hashes to a verifier challenge that the attacker is prepared to answer.

In the context of proof systems in the RO model, the term "bits of security" refers to the logarithm of the number of random oracle queries that an attacker needs to perform to achieve a constant success probability. According to Theorem 2.2, if $\kappa > 2\lambda$ (e.g., if $\lambda = 128$ and one uses a cryptographic hash function with at least 256 bits of output), then the number of bits of security achieved in the RO model after applying the Fiat-Shamir transformation is roughly $\min(\log(1/\epsilon_{RBR}), \lambda)$.
For the classical sum-check protocol, the standard soundness and the round-by-round soundness (RBR) essentially coincide: its RBR soundness is d/q [CCH+19], and its standard soundness error is vd/q, which can be derived by combining the RBR soundness with a union bound over the v rounds of interaction.

2.5 Remez theory and complex analysis

Let $\mathfrak{T}_n$ be the n'th Chebyshev polynomial of the first kind restricted to the domain $[1, \infty)$ and range $[1, \infty]$. Recall that $\mathfrak{T}_n$ is a degree n polynomial which is strictly increasing on the domain $[1, \infty)$. This can be written explicitly as $\mathfrak{T}_n(x) = \cosh(n \cdot \text{arccosh}(x))$ where $\cosh(x) := \cos(ix)$ is the hyperbolic cosine and arccosh is its inverse. Note that under this definition, $\mathfrak{T}_n^{-1}(x)$ is well-defined for all x in its range $[1, \infty)$, as for any such x there is a unique $y \in [1, \infty)$ such that $\mathfrak{T}_n(y) = x$. It may be helpful to observe that, for $y \in [1, \infty]$, $\mathfrak{T}_n(y) \in O(y^n)$. Indeed, $\mathfrak{T}_n(y)$ equals $2^{n-1} \cdot y^n +$ [lower degree terms].

The starting point for our analysis concerns a pair of Remez inequalities, one in $\mathbb{R}$ and one in $\mathbb{C}$. The first is the traditional Remez Inequality [Rem36, BE95, Gan01]:

Theorem 2.3 (Remez inequality, [Rem36]). *Let $p(x) := \sum_{k=0}^{d} a_k x^k$ be a polynomial of degree d with real coefficients. Let J be a finite interval in $\mathbb{R}$. Then, for any measurable $E \subseteq J$,*

$$\|p\|_J \leq \mathfrak{T}_d\left(\frac{2}{\lambda(E)} - 1\right) \|p\|_E. \tag{1}$$

Note that this is stronger than the usual presentation of the inequality as

$$\|p\|_J \leq \left(\frac{4}{\lambda(E)}\right)^d \|p\|_E.$$

The second is a more modern result that says:

Theorem 2.4 (Theorem 1.2, [TY20]). *Let $p(x) := \sum_{k=0}^{d} a_k x^k$ be a polynomial of degree d with complex coefficients. Then, for any measurable $E \subseteq \mathbb{T}$,*

$$\|p\|_{\mathbb{T}} \leq \mathfrak{T}_d\left(\csc\left(\frac{\pi}{2}\mu(E)\right)\right) \|p\|_E. \tag{2}$$

We note that since $\mu(E) \in [0,1]$, $\csc(\pi/2 \cdot \mu(E))$ is in $[1,\infty]$ and therefore in the domain of $\mathfrak{T}_d$, ensuring that the right hand side of the inequality above is well-defined.

Informally, these theorems state that if p is large in magnitude over a particular domain and small on a subset, then the size of the subset is bounded above by the degree of p and the ratio between "small" and "large." To illustrate, in the case of $\mathbb{C}$, Theorem 2.4 states that if p is large in magnitude over the unit circle (denoted by $\|p\|_{\mathbb{T}}$) and small on a subset $E \subseteq \mathbb{T}$ (smallness denoted by $\|p\|_E$), then the size of this subset (size denoted by $\mu(E)$) is directly related to the degree of p and the ratio between $\|p\|_T$ and $\|p\|_E$.

Although $\mathfrak{T}_d\left(\csc\left(\frac{\pi}{2}x\right)\right)$ and $\mathfrak{T}_d\left(\frac{2}{x}-1\right)$ are particularly thorny expressions, it may be helpful to think of them as $\approx 1/x^d$ terms. The intuition behind this coarse asymptotic approximation is that $\mathfrak{T}_d(x) \in O(x^d)$ and $\csc(\pi x/2)$ and $2/x-1$ behave like $1/x$. When relevant, we make this very rough intuition precise in the form of upper and lower bounds. For example, if the ratio between $\|p\|_{\mathbb{T}}$ and $\|p\|_E$ is large, then Theorem 2.4 states that $\mathfrak{T}_d\left(\csc\left(\frac{\pi}{2}\mu(E)\right)\right)$ must be small. Since this expression is roughly characterized by $1/\mu(E)^d$, we conclude that $1/\mu(E)^d$ is large and therefore (assuming d is a small constant) that $\mu(E)$ must be small.

Remark 1. Expression (2) is tight (i.e. holds with equality) for any fixed choice of $\mu(E)$ and there is a nice closed form for polynomials p for which it is tight. We discuss this in more detail and formally describe the polynomials that fit this criterion in Appendix B.

3 Approximate arithmetic computations and the approximate sum-check protocol over integral domains

In this section, we define our approximate sum-check protocol. But first, to reason formally about protocols that tolerate approximation, we introduce a model of approximate arithmetic circuits where each operation introduces bounded error ε.

Definition 4. Given an integral domain $\mathbb{U}$, associated with a metric $D : \mathbb{U} \times \mathbb{U} \to \mathbb{R}$, we define approximate computations, with respect to an approximation parameter $\varepsilon > 0$, as "approximate arithmetic circuits", which are circuits consisting of "approximate gates", denoted by $\oplus$, $\ominus$, $\otimes$ and $\oslash$, where for every $u, v \in \mathbb{U}$

$$D(u \oplus v, u+v), D(u \ominus v, u-v), D(u \otimes v, u \times v) \leq \varepsilon$$

and

$$u \oslash_\delta v := 1_{D(u,v)\leq\delta}$$

for any δ which is a positive multiple of ε. Here, $1_{D(u,v)\leq\delta}$ denotes the function that maps (u,v) to 1 if $D(u,v) \leq \delta$, and otherwise maps (u,v) to zero. [4]

In the approximate sum-check protocol, it is possible to formally model the prover and verifier as approximate circuits, where the verifier also has the ability to sample random elements from a finite subset of $\mathbb{U}$. Note that all values sent by the prover and verifier can have some error, induced by the approximate gates.

Practical considerations. Over $\mathbb{R}$ and $\mathbb{C}$ with the absolute value metric, the definition of approximate circuits captures a model of fixed-point arithmetic. If an upper bound on the magnitude of values in the circuit is known apriori, then its semantics can be implemented using floating-point arithmetic. Moreover, it is possible to extend the precision of native floating-point hardware to support nearly arbitrary precision using work from computational geometry [RS97, HLB01]. For arithmetic operations, this extension incurs polylogarithmic costs in the ratio of the desired precision to that of machine precision. In principle, this allows for an implementation of our approximate sum-check protocol that makes significant usage of a floating-point processor.

3.1 The approximate sum-check protocol

We now state our main soundness result for the approximate sum-check protocol. The theorem applies to any approximate implementation of an integral domain equipped with a distance function satisfying

[4] For clarity, in the remainder of the paper, we write $D(u,v) \leq \delta$ rather than $u \oslash_\delta v$, particularly when δ is itself an expression that does not lend itself to fitting nicely in a subscript.

natural properties. The soundness analysis requires a "distance amplification function" κ that controls how polynomial distance behaves under evaluation—roughly, $\kappa(\rho)$ represents the minimum possible ratio between the distance of two polynomials and the distance of their values at a typical evaluation point.

The key insight is that soundness depends not just on the classical probability that polynomials disagree at a random point (as in the exact sum-check), but also on how polynomial distance concentrates under the metric structure. This leads to a graceful degradation: a prover making a claim with error Δ can convince the verifier of a claim with smaller error δ only with probability related to the ratio δ/Δ, modulated by the function κ.

In the theorem statement below, the soundness analysis requires a function κ that controls how polynomial distance can contract under evaluation—roughly, $\kappa(\rho)$ bounds the factor by which the distance between two polynomials can be reduced when evaluating at a random point ρ. Smaller values of κ correspond to greater potential distance reduction, which benefits a cheating prover.

Theorem 3.1. *Consider an approximate implementation of an integral domain $\mathbb{U}$ with a metric $D : \mathbb{U} \times \mathbb{U} \to \mathbb{R}$ and with approximation parameter ε as defined in Definition 4. Let $B, S \subset \mathbb{U}$ be finite subsets called the* evaluation *and* sampling *domains, respectively.*

If the distance function, D (extended to polynomials) satisfies the property that for every $p, q \in \mathbb{U}_d[x]$ and $r \in S \cup B$

$$D(p(r), q(r)) \leq D(p, q), \tag{4}$$

then the protocol described in Figure 2 satisfies the following completeness and soundness guarantees.
***Completeness:** If $\delta \in \Omega(d\,\#(B)^v\varepsilon)$, that is, if δ is sufficiently large relative to ε, then the protocol has perfect completeness.*
***Soundness:** Suppose there exists a monotonically increasing $\kappa : [0, 1] \to \mathbb{R}_{\geq 0}$ that satisfies:*

$$\Pr_{r \sim U(S)}[D(p(r), q(r)) \leq \delta] \leq \frac{d}{|S|} + \Pr_{\rho \sim U(0,1)}[D(p, q) \cdot \kappa(\rho) \leq \delta]. \tag{5}$$

Then we have soundness error at most

$$\frac{vd}{|S|} + \Pr_{\rho_1, \cdots, \rho_v}\left[\prod_{j=1}^{v} \kappa(\rho_j) \leq \frac{v\delta}{\Delta}\right]$$

and round-by-round soundness error at most

$$\frac{d}{|S|} + \Pr_{\rho}\left[\kappa(\rho) \leq 2\sqrt[v]{\frac{\delta}{\Delta}}\right].$$

Note that Δ is purely a soundness parameter which does not impact the behavior of an honest prover or verifier in the protocol.

Interpreting the soundness bound. The soundness probability decomposes into two terms:

1. The classical term $vd/|S|$ inherited from the exact sum-check protocol, representing the probability that two distinct polynomials of total degree d polynomials agree at a random challenge point.
2. A new term measuring the probability that random "distance contraction factors" $\kappa(\rho_j)$ conspire to make a large initial error Δ appear smaller than the threshold δ.

The function κ depends on the choice of domain and distance function. For instance, we show later that:

- Over $\mathbb{R}$ with equispaced sampling points and absolute value distance, $\kappa(x) \approx 1/\mathcal{T}_d(2/x - 1)$ where $\mathcal{T}_d$ is the d-th Chebyshev polynomial (Theorem 4.2).
- Over $\mathbb{C}$ with roots of unity and absolute value distance, $\kappa(x) \approx 1/\mathcal{T}_d(\csc(\pi x/2))$ (Theorem 4.3).

In both cases, κ is an increasing function on the domain $[0, 1]$, so smaller values of the ratio δ/Δ make the soundness error smaller, as expected.

Setup. The prover and verifier agree on g, a v-variate polynomial over $\mathbb{U}$ of degree at most d in each variable, finite sets B and S (in most applications of the sum-check protocol $B = \{0, 1\}$), and δ, an error tolerance. The verifier also selects an approximate correctness parameter Δ which only affects soundness (the choice of both δ and Δ will be specified in Section 3.2.2). If any of the verifier's checks fail during the protocol, it immediately rejects.

Round 1. The prover computes the sum-check value $H \in \mathbb{U}$ with error at most δ.[a] It sends the verifier H with the claim:

$$D\left(H, \sum_{(b_1,\ldots,b_v)\in B^v} g(b_1, \ldots, b_v)\right) \leq \Delta \tag{3}$$

and a polynomial $p_1(x) \in \mathbb{U}_d[x]$ claimed to approximately equal

$$\sum_{(b_2,\ldots,b_v)\in B^{v-1}} g(x, b_2, \ldots, b_v).$$

The verifier checks that $p_1(x)$ is a univariate polynomial of degree $\leq d$ and that

$$D\left(H, \sum_{b_1\in B} p_1(b_1)\right) \leq \delta,$$

and subsequently sends a random challenge $r_1 \sim U(S)$ to the prover. This choice of r_1 can be communicated with $\log |S|$ bits by sending $r \in \{1, 2, \ldots, |S|\}$.

Round $k \in \{2, \ldots, v-1\}$. The prover sends the verifier a polynomial $p_k(x) \in \mathbb{U}_d[x]$ claimed to approximately equal

$$\sum_{(b_{k+1},\ldots,b_v)\in B^{v-k}} g(r_1, \ldots, r_{k-1}, x, b_{k+1}, \ldots, b_v).$$

The verifier checks that $p_k(x)$ is a univariate polynomial of degree $\leq d$ and that

$$D\left(p_{k-1}(r_{k-1}), \sum_{b_k\in B} p_k(b_k)\right) \leq \frac{\delta}{\#(B)^{k-1}},$$

and subsequently sends a random challenge $r_k \sim U(S)$ to the prover.

Round v. The prover sends the verifier a polynomial $p_v(x) \in \mathbb{U}_d[x]$ claimed to approximately equal

$$g(r_1, \ldots, r_{v-1}, x).$$

The verifier checks that $p_v(x)$ is a univariate polynomial of degree $\leq d$ and that

$$D\left(p_{v-1}(r_{v-1}), \sum_{b_v\in B} p_v(b_v)\right) \leq \frac{\delta}{\#(B)^{v-1}},$$

and draws a random challenge $r_v \sim U(S)$ to check that

$$D\left(p_v(r_v), g(r_1, \ldots, r_v)\right) \leq \frac{\delta}{\#(B)^v}.$$

If this final check passes, the verifier accepts the prover's claim.

[a] Completeness only holds if the distance in Equation 3 is bounded by δ rather than Δ. See Section 3.2.1.

Figure 2: The approximate sum-check protocol. It closely mirrors the classical sum-check protocol (Figure 1), with the key modifications that all verifier checks test *approximate* rather than exact equality (a notion that only makes sense for domains $\mathbb{U}$ with metric structure).

3.2 Proof of Theorem 3.1

For ease of notation in the proofs, for $k = 1, \ldots, v-1$, let

$$q_k(x) := \sum_{(b_{k+1},\ldots,b_v)\in B^{v-k}} g(r_1, \ldots, r_{k-1}, x, b_{k+1}, \ldots, b_v),$$

and $q_v(x) := g(r_1, \ldots, r_{v-1}, x)$.

3.2.1 Completeness

If the verifier was capable of exact operations over $\mathbb{U}$ then completeness would be trivial; however, the introduction of approximate operations slightly complicates the analysis.

Moreover, it is not necessarily the case that completeness holds for all H which satisfy

$$D\left(H, \sum_{(b_1,\ldots,b_v)\in B^v} g(b_1, \ldots, b_v)\right) \leq \Delta.$$

To see why, consider two protocols, one with Δ_1 and the other with Δ_2 with the property that $\Delta_1 > \Delta_2$ and all other parameters held equal. If the second protocol has any soundness, then the verifier should, in some executions, reject an initial claim H s.t. $\Delta_1 \geq H > \Delta_2$. However, for the first protocol to have perfect completeness, it must never reject that claim.

Thus when we describe perfect completeness, we restrict ourselves to a "well-behaving" prover who makes an honestly computed initial claim for the sum H (which differs by at most $O(\delta)$ from the true sum). As the prover can compute H in $O(\#(B)^v)$ operations, this is not prohibitive. Completeness should then be thought of as, for any g, an honest prover and verifier, implemented using approximate operations, can reliably engage in the approximate sum-check protocol and the verifier can always be made to accept by the prover.

Proof. For completeness, it must be the case that in every round k, an honest prover can ensure that

$$D\left(p_{k-1}(r_{k-1}), \sum_{b_k\in B} p_k(b_k)\right) \leq \frac{\delta}{\#(B)^k} \tag{6}$$

However, there are two immediate issues. First, by the nature of the prover being approximate, it is constrained to sending a p_k that might not exactly equal q_k. Second, the verifier can only compute $p_{k-1}(r_{k-1})$ and $\sum_{b_k\in B} p_k(b_k)$ using approximate operations.

To be sure that Expression (6) holds, the verifier cannot simply *compute* this expression as written. After all, it could be the case that the LHS is greater than the RHS, but that the *computed* approximate LHS is smaller. To avoid this and to preserve soundness, the verifier needs to check that the computed LHS, when added to the worst case error possible in computing the LHS, does not exceed the RHS. As every operation introduces a chance for the introduction of at most ε additive error, this is a simple accounting. Note that this *does not change the semantics* of the check the verifier is specified to perform in Figure 2. This only concerns the *implementation* of a verifier with approximate arithmetic that conforms to these semantics.

Supposing that p_{k-1} and p_k were transmitted as $d+1$ coefficients (where each at worst can be ε away from the true coefficient), via Horner's method, the verifier requires $2d$ operations to evaluate one of these polynomials at a single point. Thus the LHS can be computed in $O(d\,\#(B))$ operations. The verifier is thus assured that its computed LHS has error at most $O(d\,\#(B)\varepsilon)$.

To avoid accepting a p_k which violates Expression (6), the implemented verifier check must take the form

$$\text{computed LHS} + O(d\,\#(B)\varepsilon) \leq \text{RHS}.$$

Note that, because the prover's computation has some error, there must be a "good enough" p_k the prover can compute can send. This introduces a bound from the other direction and an additional $O(d\,\#(B)\varepsilon)$ term and is only relevant when computing δ exactly (rather than asymptotically).

For completeness to hold, it suffices that

$$D\left(q_{k-1}(r_{k-1}), \sum_{b_k \in B} q_k(b_k)\right) + O(d\,\#(B)\varepsilon) = \frac{\delta}{\#(B)^k}.$$

By definition,

$$D\left(q_{k-1}(r_{k-1}), \sum_{b_k \in B} q_k(b_k)\right) = 0,$$

so it is the case that

$$O(d\,\#(B)\varepsilon) \leq \frac{\delta}{\#(B)^k}.$$

By rewriting terms, and considering the max over all k, we have that the protocol is complete if $\delta \in \Omega(d\,\#(B)^v\varepsilon)$ which is precisely what is assumed. □

Note that in certain approximate integral domains, the bound on δ required to ensure completeness can be tightened as a result of the cancellation of errors in the computation and alternative methods of transmitting/evaluating polynomials.

As an example of such an alternative, when working over $\mathbb{C}$, the prover can send p_k as $d+1$ evaluations over the roots of unity, and the verifier can evaluate $p_k(r_k)$ using the Discrete Fourier transform.

Additionally, in a practical implementation of a prover that only performs approximate operations, the gradual shift from tolerance δ to $\delta/\,\#(B)^v$ is helpful in lowering the precision required by the prover. In each round, the denominator is inversely proportional to the number of values being summed by a typical prover in that round (and thus the accumulated additive error in p_k).

3.2.2 Soundness

Proof. We will show that

$$\Pr_{r_1,\ldots,r_v}\left[D(p_v(r_v), q_v(r_v)) \leq \frac{\delta}{\#(B)^v}\right] \leq \frac{vd}{|S|} + \Pr_{\rho_1\cdots\rho_v}\left[\prod_{j=1}^{v} \kappa(\rho_j) \leq \frac{v\delta}{\Delta}\right].$$

To establish this, we proceed by induction showing that

$$\Pr_{r_1,\ldots,r_v}\left[D(p_v(r_v), q_v(r_v)) \leq \frac{\delta}{\#(B)^v}\right]$$

is bounded above (for all $k > 1$) by

$$\frac{(v-k+1)d}{n} + \Pr_{\substack{r_1,\ldots,r_{k-1}\\ \rho_k,\ldots,\rho_v}}\left[D(p_{k-1}(r_{k-1}), q_{k-1}(r_{k-1}))\prod_{j=k}^{v} \kappa(\rho_j) \leq \frac{(v-k+1)\delta}{\#(B)^{k-1}}\right].$$

Suppose this inductive hypothesis holds from v to $k+1$.
In round k, the prover was forced to send a p_k such that

$$D\left(p_{k-1}(r_{k-1}), \sum_{b_k \in B} p_k(b_k)\right) \leq \frac{\delta}{\#(B)^{k-1}}.$$

By the triangle inequality

$$D\left(q_{k-1}(r_{k-1}), \sum_{b_k \in B} p_k(b_k)\right) \geq D\left(p_{k-1}(r_{k-1}), q_{k-1}(r_{k-1})\right) - \frac{\delta}{\#(B)^{k-1}}.$$

Since $q_{k-1}(r_{k-1}) = \sum_{b_k \in B} q_k(b_k)$, we conclude that

$$D\left(\sum_{b_k \in B} q_k(b_k), \sum_{b_k \in B} p_k(b_k)\right) \geq D\left(p_{k-1}(r_{k-1}), q_{k-1}(r_{k-1})\right) - \frac{\delta}{\#(B)^{k-1}}.$$

Thus, again by the triangle inequality,

$$\sum_{b_k \in B} D\left(p_k(b_k), q_k(b_k)\right) \geq D\left(p_{k-1}(r_{k-1}), q_{k-1}(r_{k-1})\right) - \frac{\delta}{\#(B)^{k-1}}.$$

In a nonempty set of values there must exist a value greater than or equal to the average, thus there exists a $b_k \in B$ such that

$$D(p_k(b_k), q_k(b_k)) \geq \frac{D(p_{k-1}(r_{k-1}), q_{k-1}(r_{k-1}))}{\#(B)} - \frac{\delta}{\#(B)^k}.$$

By Expression (4), we have

$$D(p_k, q_k) \geq \frac{D(p_{k-1}(r_{k-1}), q_{k-1}(r_{k-1}))}{\#(B)} - \frac{\delta}{\#(B)^k}. \tag{7}$$

By our inductive hypothesis we have that

$$\Pr_{r_1,\ldots,r_v}\left[D(p_v(r_v), q_v(r_v)) \leq \frac{\delta}{\#(B)^v}\right]$$

is bounded above by

$$\frac{(v-k)d}{|S|} + \Pr_{\substack{r_1,\ldots,r_k,\\ \rho_{k+1},\ldots,\rho_v}}\left[D(p_k(r_k), q_k(r_k)) \prod_{j=k+1}^{v} \kappa(\rho_j) \leq \frac{(v-k)\delta}{\#(B)^k}\right].$$

By Expression (5), this is bounded above by

$$\frac{(v-k+1)d}{|S|} + \Pr_{\substack{r_1,\ldots,r_{k-1},\\ \rho_k,\ldots,\rho_v}}\left[D(p_k, q_k) \cdot \prod_{j=k}^{v} \kappa(\rho_j) \leq \frac{(v-k)\delta}{\#(B)^k}\right]. \tag{8}$$

By Expression (7), this is bounded above by

$$\frac{(v-k+1)d}{|S|} + \Pr_{\substack{r_1,\ldots,r_{k-1},\\ \rho_k,\ldots,\rho_v}}\left[\left(\frac{D(p_{k-1}(r_{k-1}), q_{k-1}(r_{k-1}))}{\#(B)} - \frac{\delta}{\#(B)^k}\right) \cdot \prod_{j=k}^{v} \kappa(\rho_j) \leq \frac{(v-k)\delta}{\#(B)^k}\right].$$

Given that $0 \leq \kappa(\rho_k) \leq 1$, this is bounded by

$$\frac{(v-k+1)d}{|S|} + \Pr_{\substack{r_1,\ldots,r_{k-1},\\ \rho_k,\ldots,\rho_v}}\left[\frac{D(p_{k-1}(r_{k-1}), q_{k-1}(r_{k-1}))}{\#(B)} \cdot \prod_{j=k}^{v} \kappa(\rho_j) \leq \left(\frac{(v-k)\delta}{\#(B)^k} + \frac{\delta}{\#(B)^k}\right)\right].$$

This simplifies as

$$\frac{(v-k+1)d}{|S|} + \Pr_{\substack{r_1,\ldots,r_{k-1},\\ \rho_k,\ldots,\rho_v}}\left[D(p_{k-1}(r_{k-1}), q_{k-1}(r_{k-1})) \cdot \prod_{j=k}^{v} \kappa(\rho_j) \leq \frac{(v-k+1)\delta}{\#(B)^{k-1}}\right].$$

The case of $k = 1$ follows nearly identically, except $D(p_{k-1}(r_{k-1}), q_{k-1}(r_{k-1}))$ is replaced by Δ. This gives the result

$$\Pr_{r_1,\ldots,r_v}\left[D(p_v(r_v), q_v(r_v)) \leq \frac{\delta}{\#(B)^v}\right] \leq \frac{vd}{|S|} + \Pr_{\rho_1,\ldots,\rho_v}\left[\prod_{j=1}^{v} \kappa(\rho_j) \leq \frac{v\delta}{\Delta}\right].$$

□

3.2.3 Round-by-round soundness

The round-by-round soundness error is the sum of two terms, the first being the round-by-round soundness error of the classical sum-check protocol and the second being "additional soundness error" introduced by the notion of approximation we consider in this work.

Proof. Let $a(0) = \Delta$ and $a(v) = \frac{\delta}{\#(B)^v}$. We let $a(1), \ldots, a(v-1)$ denote values to be defined later (the precise definition of $a(i)$ for $i = 1, \ldots, v-1$ will not turn out to matter in the analysis, so long as each $a(i)$ is "not too much less than" $a(i-1)$).

Per Section 2.4, we define the doomed set $\mathcal{D}$ to be the set of all partial transcripts with last verifier message r_k where $D(p_k(r_k), q_k(r_k)) > a(k)$. That is

$$\mathcal{D} = \left\{ (H, \tau | p_k | r_k) : D\left(H, \sum_{b_1 \in B} q_1(b_1)\right) > \Delta \quad \text{and} \quad D(p_k(r_k), q_k(r_k)) > a(k) \right\}.$$

Intuitively, a transcript enters the doomed set if the prover's deviation from the true value becomes too large. The definition that $a(0) = \Delta$ ensures that Property 1 in the definition of doomed sets (Section 2.4) is by $\mathcal{D}$. The definition that $a(v) = \frac{\delta}{\#(B)^v}$ ensures that Property 3 in the definition of doomed sets is satisfied.

To show that this choice of $\mathcal{D}$ satisfies Property 2 in the definition of doomed sets, we need to show that

$$\max_k \Pr_{r_{k+1}} \left[(x, \tau | p_{k+1} | r_{k+1}) \notin \mathcal{D} \mid (x, \tau) \in \mathcal{D} \right] \leq \frac{d}{|S|} + \Pr_{\rho} \left[\kappa(\rho) \leq 2 \sqrt[v]{\frac{\delta}{\Delta}} \right].$$

This quantity itself acts as an *upper-bound* on the true round-by-round soundness error ϵ_{RBR}. By the definition of $\mathcal{D}$ we have

$$\begin{aligned} &\Pr_{r_{k+1}} \left[(x, \tau | p_{k+1} | r_{k+1}) \notin \mathcal{D} \mid (x, \tau) \in \mathcal{D} \right] \\ &= \Pr_{r_{k+1}} \left[D(p_{k+1}(r_{k+1}), q_{k+1}(r_{k+1})) \leq a(k+1) \mid (x, \tau) \in \mathcal{D} \right] \end{aligned} .$$

Applying Expression (5) allows us to upper bound this quantity as

$$\frac{d}{|S|} + \Pr_{\rho_{k+1}} \left[D(p_{k+1}, q_{k+1}) \cdot \kappa(\rho_{k+1}) \leq a(k+1) \mid (x, \tau) \in \mathcal{D} \right]. \tag{9}$$

A generalized form of Expression (7) allows us to bound this as

$$\frac{d}{|S|} + \Pr_{\rho_{k+1}} \left[\left(\frac{D(p_k(r_k), q_k(r_k))}{\#(B)} - \frac{\delta}{\#(B)^{k+1}} \right) \cdot \kappa(\rho_{k+1}) \leq a(k+1) \mid (x, \tau) \in \mathcal{D} \right].$$

This can be rearranged as

$$\frac{d}{|S|} + \Pr_{\rho_{k+1}} \left[D(p_k(r_k), q_k(r_k)) \leq a(k+1) \cdot \frac{\#(B)}{\kappa(\rho_{k+1})} + \frac{\delta}{\#(B)^k} \mid (x, \tau) \in \mathcal{D} \right].$$

From the definition of $\mathcal{D}$ this is bounded by

$$\frac{d}{|S|} + \Pr_{\rho_{k+1}} \left[a(k) \leq a(k+1) \cdot \frac{\#(B)}{\kappa(\rho_{k+1})} + \frac{\delta}{\#(B)^k} \right].$$

Which can rearranged as

$$\frac{d}{|S|} + \Pr_{\rho_{k+1}} \left[\rho_{k+1} \leq \kappa^{-1} \left(\frac{a(k+1)\,\#(B)}{a(k) - \frac{\delta}{\#(B)^k}} \right) \right].$$

This probability is explicitly $\frac{d}{|S|} + \kappa^{-1} \left(\frac{a(k+1)\,\#(B)}{a(k) - \frac{\delta}{\#(B)^k}} \right)$. Given that this is an upper bound on ϵ_{RBR} we can set the lhs to ϵ_{RBR}. Solving for an arbitrary $a(k+1)$ gives us

$$\left(\frac{a(k)}{\#(B)} - \frac{\delta}{\#(B)^{k+1}} \right) \cdot \kappa \left(\epsilon_{RBR} - \frac{d}{|S|} \right) \leq a(k+1).$$

Unrolling this recurrence gives us

$$a(0)\frac{\kappa(\epsilon_{RBR}-\frac{d}{|S|})^v}{\#(B)^v}-\delta\sum_{j=1}^{v}\frac{\kappa(\epsilon_{RBR}-\frac{d}{|S|})^j}{\#(B)^v}\leq a(v).$$

Simplifying and applying the boundary conditions on $a(\cdot)$, we get

$$\Delta\kappa\left(\epsilon_{RBR}-\frac{d}{|S|}\right)^v-\delta\sum_{j=1}^{v}\kappa\left(\epsilon_{RBR}-\frac{d}{|S|}\right)^j\leq\delta.$$

Rewriting this gives us

$$\Delta\kappa\left(\epsilon_{RBR}-\frac{d}{|S|}\right)^v\leq\delta\sum_{j=0}^{v}\kappa\left(\epsilon_{RBR}-\frac{d}{|S|}\right)^j.$$

The range of κ is $[0,1]$ so the right side is bounded above by $\delta(v+1)$. Thus we can isolate ϵ_{RBR} as

$$\epsilon_{RBR}\leq\frac{d}{|S|}+\kappa^{-1}\left(\sqrt[v]{\frac{(v+1)\delta}{\Delta}}\right). \tag{10}$$

Given that $\sqrt[v]{v+1}\leq 2$ for all $v\geq 1$ and that κ is monotonically increasing, we can simplify this as

$$\epsilon_{RBR}\leq\frac{d}{|S|}+\Pr_{\rho}\left[\kappa(\rho)\leq 2\sqrt[v]{\frac{\delta}{\Delta}}\right].$$

□

3.3 Relation to the classical sum-check protocol

Note that when the protocol in Figure 2 is instantiated with an approximate implementation of an integral domain $\mathbb{U}$ with the discrete metric $D(x,y)=1_{x\neq y}$ and $\varepsilon,\delta=0$, it corresponds exactly to the classical sum-check protocol and its soundness, round-by-round soundness, and completeness are identical to that of the classical sum-check protocol. This result follows trivially from $\delta=0$.

4 The approximate sum-check protocol implemented over $\mathbb{R}$ and $\mathbb{C}$

The previous section provides a very general approximate sum-check protocol, the results of which depend on a function κ which is determined by the choice of domain $\mathbb{U}$, distance function D, and finite set S from which the verifier draws its random challenges. Here we derive two specific instantiations of the general protocol for $\mathbb{U}=\mathbb{R}$ and $\mathbb{C}$ with D being the absolute value metric. Both of these results rely on an approximate Schwartz–Zippel-like lemma.

4.1 Approximate Schwartz–Zippel lemmas over $\mathbb{R}$ and $\mathbb{C}$

The soundness analysis of the classical sum-check protocol relies on the fact that a nonzero degree-d polynomial can vanish on at most d points. This yields the following corollary, often referred to as the univariate Schwartz–Zippel lemma.

Theorem 4.1 (Schwartz–Zippel lemma [Sch80, Zip79])**.** *Let $\mathbb{U}$ be an arbitrary integral domain. Let $p(x):=\sum_{k=0}^{d}a_kx^k$ be a polynomial of degree at most d over $\mathbb{U}$. Let $S\subseteq\mathbb{U}$ be a finite set.*

$$\Pr_{r\sim U(S)}[p(r)=0]\leq\begin{cases}\frac{d}{|S|}, & p\neq 0\\ 1, & p=0\end{cases}.$$

Theorem 4.1 underlies many exact verifiable computing protocols: if two degree-d polynomials are distinct, then they can agree on at most d out of $|S|$ evaluation points. In our setting, however, we must tolerate approximation. Rather than asking whether two polynomials are exactly equal at many points, we ask how often they can be *approximately equal*.

Theorems 4.2 and 4.3 provide approximate versions of Schwartz–Zippel over $\mathbb{R}$ and $\mathbb{C}$ respectively. Both bound the probability that a degree-d complex polynomial takes value at most δ in magnitude when evaluated at a random element of a set $S \subset \mathbb{U}$. The bound depends not only on d and $|S|$, but also on how large the polynomial is elsewhere on the unit circle. Put differently, these state that if two degree-d polynomials differ substantially at even one point in a specific domain (i.e., their difference has large norm somewhere), then their difference cannot remain small on more than a limited fraction of the domain.

The role these theorems play in our approximate setting is the same role that the classical Schwartz–Zippel lemma plays in the exact sum-check protocol.

Theorem 4.2. *Let $p(x) := \sum_{k=0}^{d} a_k x^k$ be a polynomial of degree d with real coefficients. Let n be a positive integer. Let S be n equispaced points in $\mathbb{R}$. Let J be the smallest interval which covers all points in S. Let $r \sim U(S)$. Let $\rho \sim U(0,1)$. For any non-negative constant $c \leq \|p\|_J$,*

$$\Pr_r[|p(r)| \leq c] \leq \frac{d}{n} + \Pr_\rho[\|p\|_J \cdot \kappa(\rho) \leq c]$$

where

$$\kappa(x) = \frac{1}{\mathfrak{T}_d\left((2/x) - 1\right)}.$$

Theorem 4.3. *Let $p(x) := \sum_{k=0}^{d} a_k x^k$ be a polynomial of degree d with complex coefficients. Let $S = \Omega_n$. Let $r \sim U(S)$. Let $\rho \sim U(0,1)$. For any non-negative constant $c \leq \|p\|_{\mathbb{T}}$,*

$$\Pr_r[|p(r)| \leq c] \leq \frac{d}{n} + \Pr_\rho[\|p\|_{\mathbb{T}} \cdot \kappa(\rho) \leq c]$$

where

$$\kappa(x) = \frac{1}{\mathfrak{T}_d\left(\csc\left(\frac{\pi}{2}x\right)\right)}. \tag{11}$$

The proofs of these theorems follow a nearly identical structure, so we opt to only directly present the proof of Theorem 4.3 here, leaving the other as an exercise to the reader. Additionally, let $s := p - q$ and $I = \mathbb{T}$ when working over $\mathbb{C}$, $I = J$ when working over $\mathbb{R}$, then it trivially follows that $D(p(r), q(r)) = |p(r) - q(r)| = |s(r)|$ and

$$D(p, q) = \|p - q\|_I = \|s\|_I \tag{12}$$

Thus these theorems are in the requisite form for the approximate sum-check soundness analysis.

Proof of Theorem 4.3. Let E_c be defined as $E_c = \{x \in \mathbb{T} : |p(x)| \leq c\}$. By the fundamental theorem of algebra, E_c must comprise the union of at most d continuous arcs on the complex unit circle. Moreover, if any such arc I contains ℓ different n'th roots of unity, then $\mu(I) \geq (\ell - 1)/n$. It follows that

$$\Pr_r[|p(r)| \leq c] = \Pr_r[r \in E_c] = \frac{\#(\Omega_n \cap E_c)}{n} \leq \frac{d}{n} + \mu(E_c).$$

From Theorem 2.4, we have that

$$\|p\|_{\mathbb{T}} \leq \mathfrak{T}_d\left(\csc\left(\frac{\pi}{2}\mu(E_c)\right)\right)\|p\|_{E_c}.$$

Rewriting gives us

$$\mu(E_c) \leq \frac{2}{\pi}\csc^{-1}\left(\mathfrak{T}_d^{-1}\left(\frac{\|p\|_{\mathbb{T}}}{\|p\|_{E_c}}\right)\right). \tag{13}$$

Note that the right hand side of Expression (13) is well-defined, and exclusively takes on values in the range $[0, 1]$. This holds by the following reasoning: $\|p\|_{\mathbb{T}}/\|p\|_{E_c} \geq 1$, and hence $\mathfrak{T}_d^{-1}(\|p\|_{\mathbb{T}}/\|p\|_{E_c}) \geq 1$.

This ensures that $\csc^{-1}(\mathfrak{T}_d^{-1}(\|p\|_{\mathbb{T}}/\|p\|_{E_c}))$ is well-defined. Moreover, $\csc^{-1}$ is a strictly decreasing function over the interval $[1, \infty)$, with $\csc^{-1}(1) = \pi/2$. Hence,

$$\csc^{-1}(\mathfrak{T}_d^{-1}(\|p\|_{\mathbb{T}}/\|p\|_{E_c})) \leq \pi/2.$$

Accordingly, the right hand side of Expression (13) is in $[0, 1]$.

So by substitution we have

$$\Pr_r[|p(r)| \leq c] \leq \frac{d}{n} + \frac{2}{\pi}\csc^{-1}\left(\mathfrak{T}_d^{-1}\left(\frac{\|p\|_{\mathbb{T}}}{c}\right)\right) = \frac{d}{n} + \Pr_\rho\left[\frac{2}{\pi}\csc^{-1}\left(\mathfrak{T}_d^{-1}\left(\frac{\|p\|_{\mathbb{T}}}{c}\right)\right) \geq \rho\right].$$

Here, the final equality invokes the elementary fact that for any $C \in [0, 1]$, $\Pr_{\rho\sim[0,1]}[\rho \leq C] = C$.

The proof concludes by rewriting the inside of the probability

$$\Pr_\rho\left[\frac{2}{\pi}\csc^{-1}\left(\mathfrak{T}_d^{-1}\left(\frac{\|p\|_{\mathbb{T}}}{c}\right)\right) \geq \rho\right] = \Pr_\rho\left[\frac{\|p\|_{\mathbb{T}}}{\mathfrak{T}_d\left(\csc\left(\frac{\pi}{2}\rho\right)\right)} \leq c\right] = \Pr_\rho[\|p\|_{\mathbb{T}} \cdot \gamma(\rho) \leq c].$$

□

4.2 Application to $\mathbb{R}$

Theorem 4.4. *When the protocol in Figure 2 is instantiated over $\mathbb{R}$ with the absolute value metric, where S is a set of equispaced points and B is in the convex hull of S, it has soundness error at most*

$$\frac{vd}{|S|} + 2^v\sqrt[c_d]{\frac{v\delta}{\Delta}}$$

for $c_d \approx 3d$. Note, this c_d is defined precisely later; however, it is contained in the interval $[2.5d, 3.4d]$ and is strictly greater than $3d$ when $d \geq 3$.

Proof. Beginning where Theorem 3.1 left off, we have that if the prover begins the protocol with a claim that

$$\left|H - \sum_{b_1 \in B} q_1(b_1)\right| > \Delta,$$

then it is the case that

$$\Pr_{r_1,\ldots,r_v}\left[D(p_v(r_v), q_v(r_v)) \leq \frac{\delta}{\#(B)^v}\right] \leq \frac{vd}{|S|} + \Pr_{\rho_1,\ldots,\rho_v}\left[\prod_{j=1}^{v}\kappa(\rho_j) \leq \frac{v\delta}{\Delta}\right].$$

By Chernoff's bound, for arbitrary $t \geq 0$, the above expression equals:

$$\frac{vd}{|S|} + \Pr\left[e^{-t\cdot\ln\left(\prod_{j=1}^{v}\kappa(\rho_j)\right)} \geq e^{-t\cdot\ln\left(\frac{v\delta}{\Delta}\right)}\right].$$

This in turn is bounded above by:

$$\begin{aligned}
&\frac{vd}{|S|} + \frac{\mathbb{E}\left[e^{-t\cdot\ln\left(\prod_{j=1}^{v}\kappa(\rho_j)\right)}\right]}{e^{-t\cdot\ln\left(\frac{v\delta}{\Delta}\right)}} \\
&= \frac{vd}{|S|} + e^{t\cdot\ln\left(\frac{v\delta}{\Delta}\right)} \cdot \mathbb{E}\left[\prod_{j=1}^{v} e^{-t\cdot\ln(\kappa(\rho_j))}\right] \\
&= \frac{vd}{|S|} + e^{t\cdot\ln\left(\frac{v\delta}{\Delta}\right)} \cdot \prod_{j=1}^{v}\mathbb{E}\left[e^{-t\cdot\ln(\kappa(\rho_j))}\right].
\end{aligned}$$

By the independence of each ρ_k the above expression equals

$$\frac{vd}{|S|} + \left(\frac{v\delta}{\Delta}\right)^t \mathbb{E}\left[\kappa(\rho)^{-t}\right]^v.$$

By the definition of $\mathbb{E}$, this is equal to

$$\frac{vd}{|S|} + \left(\frac{v\delta}{\Delta}\right)^t \left(\int_0^1 \kappa(\rho)^{-t} d\rho\right)^v. \tag{14}$$

By the definition of κ for $\mathbb{R}$, this is equal to

$$\frac{vd}{|S|} + \left(\frac{v\delta}{\Delta}\right)^t \left(\int_0^1 \mathfrak{T}_d\left(\frac{2}{\rho} - 1\right)^t d\rho\right)^v.$$

Remark 2. The choice of t that yields the tightest bound on the soundness error can be optimized numerically as a function of v, d, δ, and Δ. A short Sage script for performing this optimization is provided in Appendix A. For the remainder of the proof, we proceed with a fixed, suboptimal choice of t to obtain a clean explicit bound.

For $x \geq 0$, $\mathfrak{T}_d(x) \leq 2^{d-1}x^d$ so the expression above is bounded by

$$\frac{vd}{|S|} + \left(\frac{v\delta}{\Delta}\right)^t \left(2^{t(d-1)} \int_0^1 \left(\frac{2}{\rho} - 1\right)^{td} d\rho\right)^v.$$

A straightforward substitution produces the integral

$$\frac{vd}{|S|} + \left(\frac{v\delta}{\Delta}\right)^t \left(2^{1+t(d-1)} \int_1^\infty \frac{u^{td}}{(1+u)^2} du\right)^v. \tag{15}$$

We will write $\int_1^\infty \frac{u^{td}}{(1+u)^2} du$ as

$$\int_0^\infty \frac{u^{td}}{(1+u)^2} du - \int_0^1 \frac{u^{td}}{(1+u)^2} du.$$

By taking $a = td + 1$ and $b = 1 - td$, This is defined to be

$$\beta(a, b) - \beta_{1/2}(a, b) = \beta(td + 1, 1 - td) - \beta_{1/2}(td + 1, 1 - td)$$

where β and β_x are the complete and incomplete beta functions respectively. A standard reflection identity is

$$\beta(x + 1, 1 - x) = \frac{td\pi}{\sin(\pi td)}$$

when $-1 < x < 1$, so if we define c_d to satisfy the following identity:

$$2^{1+\frac{d-1}{c_d}} \left(\frac{d\pi}{c_d \sin(\pi d/c_d)} - \beta_{1/2}\left(\frac{d}{c_d} + 1, 1 - \frac{d}{c_d}\right)\right) = 2, \tag{16}$$

and fix $t = 1/c_d$ (as mentioned earlier, this choice of t does not in general lead to the strongest possible error bound), substituting this value of t in Expression (15) yields the final bound:

$$\Pr[|p_v(r_v) - q_v(r_v)| \leq \delta] \leq \frac{vd}{|S|} + 2^v \sqrt[c_d]{\frac{v\delta}{\Delta}}.$$

□

4.2.1 Bounding c_d

We asserted that for small d, the value of c_d satisfying Expression (16) is approximately $3d$. Numerically, we find

$$c_1 \approx 2.521025,\ c_2 \approx 5.855944,\ c_3 \approx 9.208955,\ c_4 \approx 12.565672,\ c_5 \approx 15.923761.$$

More generally, we claim that the ratio c_d/d is strictly increasing and that $2.5d < c_d < 3.4d$ for all $d \geq 1$. Combined with the computed values above, this justifies the loose approximation $c_d \approx 3d$, and in particular shows that $c_d > 3d$ for $d \geq 3$. The proof is straightforward and omitted here in favor of a proof in the following subsection as working over $\mathbb{C}$ yields concretely stronger results and the two have the same general flavor.

4.2.2 Round-by-round soundness

Theorem 4.5. *When the protocol in Figure 2 is instantiated over $\mathbb{R}$ with the absolute value metric, where S is a set of equispaced points and B is in the convex hull of S,, it has round-by-round soundness error at most*

$$\frac{d}{|S|} + 8\sqrt[vd]{\frac{\delta}{\Delta}}.$$

Proof. This proof follows from Expression (10), where the generic proof of round-by-round soundness left off, namely that

$$\epsilon_{RBR} \leq \frac{d}{|S|} + \kappa^{-1}\left(2\sqrt[v]{\frac{\delta}{\Delta}}\right).$$

Given that for $x \in [0,1]$ $\kappa(x) \geq (x/4)^d$, we have

$$\epsilon_{RBR} \leq \frac{d}{|S|} + 4\sqrt[d]{2\sqrt[v]{\frac{\delta}{\Delta}}}.$$

Given that $\sqrt[d]{2} \leq 2$ for all $d \geq 1$, we have

$$\epsilon_{RBR} \leq \frac{d}{|S|} + 8\sqrt[vd]{\frac{\delta}{\Delta}}.$$

This completes the proof. □

4.2.3 A simple soundness corollary

Corollary 4.6 (Corollary of Theorem 4.4)**.** *If the protocol in Figure 2 is instantiated over $\mathbb{R}$ with the absolute value metric, $B = \{0,1\}$, and S is a set of at least $4vd$ equispaced values in the interval $[0,1]$, then it has soundness error at most $1/2$ when $\Delta \in O\left(vd2^{O(vd)}\varepsilon\right)$.*

Proof. This is a straightforward application of Theorem 4.4. The described protocol has soundness error at most

$$\frac{vd}{|S|} + 2^v\sqrt[c_d]{\frac{v\delta}{\Delta}}$$

for a $c_d < 3.4d$. To determine the Δ for which the protocol has soundness at least $1/2$ we can plug in $|S| = 4vd$ and solve

$$\frac{1}{4} + 2^v\sqrt[3.4d]{\frac{v\delta}{\Delta}} \leq \frac{1}{2}.$$

This gives us

$$\Delta \in O\left(v\delta 2^{O(vd)}\right).$$

Given that, by assumption, $\delta \approx d2^v\varepsilon$ (see the completeness condition in Theorem 3.1),

$$\Delta \in O\left(vd2^{O(vd)}\varepsilon\right).$$

This is precisely what we wanted to prove, so we are done. □

4.3 Application to $\mathbb{C}$

Theorem 4.7. *When the protocol in Figure 2 is instantiated over $\mathbb{C}$ with the absolute value metric where S is a set of roots of unity and $B \subset \mathbb{D}$, it has soundness error at most*

$$\frac{vd}{|S|} + 2^v \sqrt[c_d]{\frac{v\delta}{\Delta}}$$

for a $c_d \approx 2d$. Note, as with Theorem 4.4 this c_d is defined precisely later. Here it is contained in the interval $[1.6d, 2.4d]$ and strictly greater than $2d$ when $d \geq 3$.

Proof. This proof flows similarly to the proof for $\mathbb{R}$ and picks up at Expression (14). By the definition of κ for $\mathbb{C}$, we have

$$\frac{vd}{|S|} + \left(\frac{v\delta}{\Delta}\right)^t \left(\int_0^1 \kappa(\rho)^{-t} d\rho\right)^v = \frac{vd}{|S|} + \left(\frac{v\delta}{\Delta}\right)^t \left(\int_0^1 \mathfrak{T}_d\left(\csc\left(\frac{\pi}{2}\rho\right)\right)^t d\rho\right)^v.$$

A Sage script for the explicit calculation of this formula is contained in Appendix A. Here, we will use the fact that for $x \geq 0$, $\mathfrak{T}_d(x) \leq 2^{d-1}x^d$ to bound the expression above by

$$\frac{vd}{|S|} + \left(\frac{v\delta}{\Delta}\right)^t \left(\frac{2^{1+t(d-1)}}{\pi} \int_0^{\frac{\pi}{2}} \csc(\rho)^{td} d\rho\right)^v.$$

For small $0 < t < \frac{1}{d}$ (restricted to where the integral is defined) this is equal to

$$\frac{vd}{|S|} + \left(\frac{v\delta}{\Delta}\right)^t \left(\frac{2^{t(d-1)}}{\pi} \beta\left(\frac{1}{2}, \frac{1-td}{2}\right)\right)^v \tag{17}$$

where β is the beta function. For simplicity, we define c_d to satisfy the following identity:

$$\frac{2^{\frac{d-1}{c_d}}}{\pi} \beta\left(\frac{1}{2}, \frac{1-\frac{d}{c_d}}{2}\right) = 2, \tag{18}$$

and fix $t = 1/c_d$ (as mentioned earlier, this choice of t does not in general lead to the strongest possible error bound). By the impromptu definition of c_d in Expression (18), substituting this value of t yields the final bound:

$$\Pr[|p_v(r_v) - q_v(r_v)| \leq \delta] \leq \frac{vd}{|S|} + 2^v \sqrt[c_d]{\frac{v\delta}{\Delta}}.$$

This completes the proof. □

4.3.1 Bounding c_d

We asserted that for small d, the value of c_d satisfying Expression (18) is approximately $2d$. Numerically, we find

$$c_1 \approx 1.663516,\ c_2 \approx 3.969131,\ c_3 \approx 6.319634,\ c_4 \approx 8.679344,\ c_5 \approx 11.042419.$$

More generally, we claim that the ratio c_d/d is strictly increasing and that $c_d < 2.4d$ for all d. Combined with the computed values above, this justifies the loose approximation $c_d \approx 2d$, and in particular shows that $c_d > 2d$ for $d \geq 3$.

To prove the upper bound, consider the left-hand side of Expression (18) with the substitution $c_d = xd$:

$$\frac{2^{\frac{d-1}{xd}}}{\pi} \beta\left(\frac{1}{2}, \frac{1-\frac{1}{x}}{2}\right). \tag{19}$$

We claim this expression is non-negative, strictly increasing in d for fixed $x > 1$, and strictly decreasing in x for fixed $d > 1$. The monotonicity in d follows directly from the exponent $\frac{d-1}{xd}$ increasing with d. The

monotonicity in x follows from the facts that both the exponent and the beta function $\beta\left(\frac{1}{2}, \frac{1-\frac{1}{x}}{2}\right)$ decrease with x.

It therefore suffices to show that the limiting value of Expression (19) as $d \to \infty$ is less than 2 when $x = 2.4$. Indeed,

$$\frac{2^{\frac{d-1}{2.4d}}}{\pi}\beta\left(\frac{1}{2}, \frac{1-\frac{1}{2.4}}{2}\right) \leq \lim_{d\to\infty} \frac{2^{\frac{d-1}{2.4d}}}{\pi}\beta\left(\frac{1}{2}, \frac{1-\frac{1}{2.4}}{2}\right) < 1.98.$$

Hence, $c_d < 2.4d$ for all d. Tighter upper bounds on c_d can be obtained by solving Expression (18) numerically for larger d, or by optimizing the limiting expression.

4.3.2 Round-by-round soundness

Theorem 4.8. *When the protocol in Figure 2 is instantiated over $\mathbb{C}$ with the absolute value metric where S is a set of roots of unity and $B \subset \mathbb{D}$, it has round-by-round soundness error at most*

$$\frac{d}{|S|} + 2\sqrt[vd]{\frac{\delta}{\Delta}}.$$

Proof. This proof follows from Expression (10), where the generic proof of round-by-round soundness left off. Given that for $x \in [0,1]$ $\kappa(x) \geq x^d/d$, and then that for $d = 1$ or $d \geq 2$, $\sqrt[d]{2d} \leq 2$ we have

$$\epsilon_{RBR} \leq \frac{d}{|S|} + \kappa^{-1}\left(2\sqrt[v]{\frac{\delta}{\Delta}}\right) \leq \frac{d}{|S|} + \sqrt[d]{2d\sqrt[v]{\frac{\delta}{\Delta}}} \leq \frac{d}{|S|} + 2\sqrt[vd]{\frac{\delta}{\Delta}}.$$

This completes the proof. □

Remark 3. We conjecture the following general amplification property: Let Π be an r-round interactive protocol with round-by-round (RBR) soundness error ϵ. Then, $(k \cdot r)$-fold parallel repetition of Π results in a protocol with RBR soundness error at most ϵ^k. In particular, this conjecture implies that $(k \cdot r)$-fold parallel repetition boosts the Fiat-Shamir security level from $\log(1/\epsilon)$ bits to $k\log(1/\epsilon)$ bits (without requiring an increase in the precision of individual instances). This conjecture is known to hold in the special case of r-round protocols whose soundness error ϵ and RBR soundness error ϵ_{RBR} are maximally separated (i.e., $\epsilon_{RBR} \approx \epsilon^{1/r}$) [CCH+18, Corollary 5.7].

5 Application: Approximate fully linear IOPs

The sum-check protocol is a crucial building block in protocols for verifiable computation, including the GKR protocol [GKR08] and many follow-up works. However, its application to GKR in the approximate setting requires an extension of our analysis that we leave for future work. The core obstacle to this analysis is the existence of adaptive prover strategies applicable here but not to the classical protocol. Despite this, here we present a "low-end" application of the sum-check protocol, to verify simple arithmetic statements on distributed or secret-shared data. We use this as a proof of concept, demonstrating that our approximate sum-check protocol and its analysis can yield meaningful improvements for natural use cases with practically relevant parameters.

Consider the following motivating problem. A prover P owns two large data sets $U, V \in \mathbb{R}^N$ that are stored on two separate (trusted) servers, say two different hospitals. The data owner P, who may be malicious, wants to convince a verifier that the two data sets are essentially uncorrelated in the sense that their inner product satisfies $\langle U, V\rangle \approx 0$. (Alternatively, one can similarly consider proving any approximate inner-product value.) How can this be done with low communication complexity?

Note that even when the servers are fully honest, they cannot compute the inner product with low communication. Indeed, this requires $\Omega(N)$ bits of communication even when $U, V \in \{0,1\}^N$. On the other hand, the prover knows the correct inner product and can easily communicate it, but cannot be trusted.

Aaronson and Wigderson [AW08] considered an exact version of the problem over a finite field, and used a variant of the sum-check protocol to obtain a low-communication solution. This idea was generalized and

abstracted by Boneh et al. [BBC+19] using the notion of a *fully linear IOP* (FLIOP). An FLIOP for a language $L \subseteq \mathbb{U}^N$ is defined similarly to a standard interactive oracle proof (IOP) [BCS16, RRR16], except that the verifier does not have direct access to the input and instead can make few *linear queries* that apply jointly to the input and proofs. Such FLIOPs with short proofs and few linear queries directly imply low-communication protocols in the above distributed setting, since each linear query can be easily distributed between the two servers by having each respond with the part that depends on its own view of the input.

More precisely, let $L \subseteq \mathbb{U}^n$ be a language. An FLIOP for proving that $x \in L$ proceeds in rounds. In each round, there is a random public challenge r_k to which the prover responds with a proof vector $\pi_k \in \mathbb{U}^{\ell_k}$. At the end of the interaction, the verifier can make a small number of queries, where each query specifies a joint linear combination of the input x and the proofs π_i. Letting $\mathbb{U} = \mathbb{R}$ and extending to the approximate setting, we get the following theorem.

Theorem 5.1 (Approximate arithmetic FLIOP for inner product)**.** *Let* $L_{\text{IP}} = \{(U, V) \in \mathbb{R}^N \times \mathbb{R}^N : \langle U, V \rangle = 0\}$ *where* N *is a power of 2. Then, there is an approximate arithmetic FLIOP for* L_{IP} *that has* $\log N$ *rounds and total proof length and query description length (in field elements)* $O(\log N)$*. The completeness and soundness guarantees are the same as those of the approximate sum-check (Theorem 4.4) with* $B = \{0, 1\}$*,* $v = \log N$*, and* $d = 2$*.*

Proof. Let g_U, g_V be the multilinear extensions of U, V, namely the unique multilinear polynomials such that for any $b = (b_1, \ldots, b_v) \in B^v$ we have $g_U(b) = U_b$ and $g_V(b) = V_b$. The prover needs to convince the verifier that $\sum_{b \in B^v} g(b) \approx 0$ for the polynomial $g = g_U \cdot g_V$. Note that g has degree $d = 2$ in each of the $v = \log N$ variables.

The approximate sum-check for g from Figure 2 can be converted into an FLIOP for L_{IP} as follows. The FLIOP prover lets each proof π_k include the coefficients of the univariate polynomial p_k sent by the sum-check prover. Note that each intermediate check of consistency between p_k and p_{k-1} can be performed by the FLIOP verifier via a single linear query to the coefficients of the two polynomials. The final check requires the evaluation of g on a random public point $r \in \mathbb{R}^v$. This can be done by having the FLIOP verifier separately query $g_U(r)$ and $g_V(r)$, where each such query can be implemented via a single linear query to the input $x = (U, V)$ whose coefficients are determined by r. □

Analogously to [BBC+19], the above can be extended to more general languages and can be made non-interactive in the random oracle model using a distributed variant of the Fiat-Shamir transformation. We also expect a zero-knowledge variant of arithmetic FLIOPs to be useful for achieving security against malicious parties in information-theoretic approximate secure computation over the reals. Such applications of FLIOPs over finite fields were demonstrated in [BBC+19] and follow-up works. We leave these extensions to future work.

Concrete efficiency. Even though this FLIOP is defined over $\mathbb{R}$, we will instantiate it over $\mathbb{C}$ as $\mathbb{R}$ is a subfield of $\mathbb{C}$. We do that here to achieve concretely better parameters and as an explicit demonstration of the possibility of using one integral domain to seamlessly prove a statement about another. Suppose, for simplicity, that $N := 2^{30}$ and $|S| = 4vd = 240$. The protocol achieves soundness at least $1/2$ when δ/Δ is at least 2^{111} (that is, the error bound Δ in the prover's claim differs from the error bounds in the verifier's checks by 111 bits). These extra 111 bits of precision can be represented using just two additional 64-bit words. If the number of bits of precision slightly more than doubles, to 227, the soundness error drops below 2^{-30}. These bounds are derived numerically by optimizing the choice of t in Expression (17) using the script provided in Appendix A. A high-level comparison to other approaches is provided in the next section.

6 Detailed comparison with alternative approaches

Here we compare our generalized sum-check protocol, which treats approximation as a first-class concept, to alternative approaches that apply exact techniques to the verification of approximate computations. These alternatives fall into two broad categories, which we refer to as the *exact* and *localized* approaches (we call our new method the *approximate* approach). Each approach makes different trade-offs. Note that because our protocol applies in a fundamentally different model than alternative approaches (see discussion in Section 1.1), there are aspects in which they are incomparable. We do our best to bridge the gap at a high-level here.

The exact approach. Several recent works observe that the classical sum-check protocol can be instantiated over fields of characteristic zero, allowing exact arithmetic over $\mathbb{R}$ or $\mathbb{C}$ [SV22, CCKP19, CCKP22, BCS21]. In these constructions, the verifier's random challenges are sampled from a discrete subset of the ambient ring (e.g., $\mathbb{R}$ or $\mathbb{C}$), and the numerical precision required by the prover increases gradually round-by-round. While conceptually clean, this approach imposes a substantial burden on the prover in terms of the bits of precision needed for accurate computation.

The localized approach. Other recent works [CHA24, GWHD25] apply the classical sum-check protocol to exact statements over $\mathbb{Z}$ or $\mathbb{Q}$ by localizing to a large finite field $\mathbb{F}_q$ of random order. This allows computations to be carried out using finite-field arithmetic, without precision blow-up from round-to-round growth. However, the bit-lengths of the rational evaluations that must be represented before localization are often a constant factor larger than the desired security parameter λ [GWHD25, Section 2], and this constant can depend non-trivially on the structure of the computation (e.g., its multiplicative depth).

The approximate approach. Our generalized sum-check protocol takes a fundamentally different path: rather than forcing approximate computations into exact algebraic structures, it embraces approximation at the protocol level. The prover and verifier work directly with floating-point data at a tunable precision, and soundness is derived from the metric structure of low-degree complex polynomials. For a concrete comparison, we consider here its instantiation over $\mathbb{C}$ as described in Section 4.3.

Comparison of numerical precision. Let d be the degree of the polynomial in each variable, v the number of variables, and λ the desired number of bits of soundness error. The number of bits of precision required in each approach is:

- **Exact approach:** $O(vd\lambda + \log(vd))$ bits. Conceptually, the number of bits of precision increases by $d\lambda$ over each of the v rounds of the sum-check protocol.
- **Localized approach:** $O(\log(vd) + c\lambda)$ bits, where c depends on the application (e.g., reflecting multiplicative depth) and is larger than 1.
- **Approximate approach:** $O(\max(\log(vd)+\lambda, v+vd+d\lambda))$ bits, based on the need to control two sources of soundness error per Theorem 4.7: the classical error term vd/n (inherited from the classical sum-check protocol) and a new approximation-driven error term roughly $\sqrt[c]{v\delta/\Delta}$ (introduced by the relaxed checks in our protocol).

Although the asymptotic precision requirement for the approximate approach does not strictly improve over the localized approach, our prover performs a much smaller *number* of operations at the required precision level. This efficiency stems from our avoidance of arithmetization: we work directly with the original approximate computation, rather than encoding it into an exact algebraic form over a finite field, as discussed further below. Moreover, on a practical level, the prover and verifier can be implemented using native floating-point instructions, which are significantly faster and more efficiently parallelizable on modern hardware than large finite field operations.

In interactive settings, it is typical to set $\lambda = 40$ [ZKP22] to achieve less than one in a trillion odds of a successful attack. In this regime, our approximate protocol offers especially compelling performance, while avoiding the substantial operational overheads of finite field encodings.

Departing from arithmetization. Traditionally, applications of the sum-check protocol rely on arithmetization: the process of transforming discrete computations into exact algebraic claims over finite fields. This technique underpins a wide range of results in complexity theory, from IP = PSPACE to the PCP theorem [AW09]. More recently, it has been shown that the soundness of many interactive proof protocols, including sum-check, can be derived from the error-correcting properties of algebraic codes [Mei13] (specifically, *multiplication codes* [RR22, RR24], of which low-degree polynomials are a canonical example).

In contrast, our approximate sum-check protocol avoids both arithmetization of the target computation and reliance on error-correcting codes for soundness. Instead, our proofs rely on purely analytical arguments about metric structure and the sublevel set behavior of low-degree polynomials. This opens the door to interactive proof techniques for inherently approximate or numerical computations, where traditional arithmetization would obscure rather than clarify the underlying structure.

7 Discussion

We introduced a generalization of the sum-check protocol that supports approximate computation. The protocol proves claims of the form $\sum_{x\in\{0,1\}^v} g(x) \approx H$ with soundness that scales with the error in the claim, and avoids the need for exact arithmetic or error-correcting codes. Our analysis replaces classical code-based arguments with bounds on sublevel sets of complex polynomials, using tools from approximation theory. The approach applies broadly across domains and metrics, but is most natural over the complex numbers.

The protocol can be implemented with floating-point arithmetic, requiring only a constant-factor overhead in precision relative to the target security parameter for many applications. This provides a potential path around the large costs associated with existing verifiable computation protocols for numerical workloads. Our soundness analysis also identifies a phenomenon that appears inherent to approximate computation: after applying the Fiat-Shamir transform, a cheating prover can make an inaccurate answer appear slightly less erroneous in each round. For example, our approximate sum-check protocol, instantiated over $\mathbb{C}$, achieves RBR soundness roughly $\epsilon^{2/r}$, an intermediate regime between these two extremes.

Several directions remain open. Most SNARKs combine a *polynomial interactive oracle proof* (PIOP) with a polynomial commitment scheme. Our focus here is on the PIOP layer; realizing full SNARKs will require identifying polynomial commitment schemes that maintain soundness under approximation. While the prover is asymptotically efficient, implementation work is needed to understand concrete performance. On the theory side, our analysis relies on a Remez-type inequality for complex polynomials, but similar inequalities exist for broader function classes, such as algebraic functions, (s, p)-valent functions, and others [Bru03, Yom11, Fri21]. It is natural to explore whether the use of these richer function classes within the approximate sum-check protocol leads to any performance benefits. The mathematical tools from complex analysis and approximation theory that we applied have not previously seen use in cryptography. Future work may find broader applications of these tools in other information theoretic and cryptographic protocols.

Disclosures. Justin Thaler is a Research Partner at a16z crypto and is an investor in various blockchain-based platforms, as well as in the crypto ecosystem more broadly (for general a16z disclosures, see `https://www.a16z.com/disclosures/`).

8 Acknowledgments

Yuval Ishai was supported in part by ISF grants 2774/20 and 3527/24 and BSF grant 2022370. This work is not associated with Amazon. Yael Kalai is supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-25-C-0300. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA).

References

[ABCG93] S. Ar, M. Blum, B. Codenotti, and P. Gemmell. Checking approximate computations over the reals. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing*, STOC '93, page 786–795, New York, NY, USA, 1993. Association for Computing Machinery.

[ABF+23] Vipul Arora, Arnab Bhattacharyya, Noah Fleming, Esty Kelman, and Yuichi Yoshida. Low degree testing over the reals. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 738–792. SIAM, 2023.

[ALM+92] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 14–23, 1992.

[AS92] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 2–13, 1992.

[AW08] Scott Aaronson and Avi Wigderson. Algebrization: a new barrier in complexity theory. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 731–740. ACM, 2008.

[AW09] Scott Aaronson and Avi Wigderson. Algebrization: A New Barrier in Complexity Theory. *ACM Trans. Comput. Theory*, 1(1), February 2009.

[BBC+19] Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Zero-knowledge proofs on secret-shared data via fully linear pcps. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part III*, volume 11694 of *Lecture Notes in Computer Science*, pages 67–97. Springer, 2019.

[BCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive oracle proofs. In Martin Hirt and Adam D. Smith, editors, *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, volume 9986 of *Lecture Notes in Computer Science*, pages 31–60, 2016.

[BCS21] Jonathan Bootle, Alessandro Chiesa, and Katerina Sotiraki. Sumcheck arguments and their applications. In Tal Malkin and Chris Peikert, editors, *Advances in Cryptology – CRYPTO 2021*, pages 742–773, Cham, 2021. Springer International Publishing.

[BE95] Peter Borwein and Tamás Erdélyi. *Polynomials and Polynomial Inequalities*, volume 161 of *Graduate Texts in Mathematics*. Springer-Verlag New York, Inc., 1995.

[BGTZ25] Alexander R Block, Albert Garreta, Pratyush Ranjan Tiwari, and Michał Zając. On soundness notions for interactive oracle proofs. *Journal of Cryptology*, 38(1):4, 2025.

[Bru03] Alexander Brudnyi. On local behavior of analytic functions. *Journal of Functional Analysis*, 201(1):211–228, 2003.

[BSCS16] Eli Ben-Sasson, Alessandro Chiesa, and Nicholas Spooner. Interactive Oracle Proofs. In *Proceedings, Part II, of the 14th International Conference on Theory of Cryptography - Volume 9986*, page 31–60, Berlin, Heidelberg, 2016. Springer-Verlag.

[CCH+18] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, and Ron D. Rothblum. Fiat-shamir from simpler assumptions. Cryptology ePrint Archive, Paper 2018/1004, 2018.

[CCH+19] Ran Canetti, Yilei Chen, Justin Holmgren, Alex Lombardi, Guy N. Rothblum, Ron D. Rothblum, and Daniel Wichs. Fiat-Shamir: from practice to theory. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, page 1082–1090, New York, NY, USA, 2019. Association for Computing Machinery.

[CCKP19] Shuo Chen, Jung Hee Cheon, Dongwoo Kim, and Daejun Park. Verifiable computing for approximate computation. Cryptology ePrint Archive, Paper 2019/762, 2019.

[CCKP22] Shuo Chen, Jung Hee Cheon, Dongwoo Kim, and Daejun Park. Interactive proofs for rounding arithmetic. *IEEE Access*, 10, 2022.

[CHA24] Matteo Campanelli and Mathias Hall-Andersen. Fully succinct arguments over the integers from first principles. Cryptology ePrint Archive, Paper 2024/1548, 2024.

[CY24] Alessandro Chiesa and Eylon Yogev. Building cryptographic proofs from hash functions, 2024. CC BY-SA 4.0.

[Fri21] Omer Friedland. Remez-type inequalities and their applications. In Evgeny Abakumov, Anton Baranov, Alexander Borichev, Konstantin Fedorovskiy, and Joaquim Ortega-Cerdà, editors, *Extended Abstracts Fall 2019*, pages 73–79, Cham, 2021. Springer International Publishing.

[FS87] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *Advances in Cryptology — CRYPTO' 86*, pages 186–194, Berlin, Heidelberg, 1987. Springer Berlin Heidelberg.

[Gan01] Michael I. Ganzburg. Polynomial inequalities on measurable sets and their applications. *Constructive Approximation*, 17(2):275–306, Jan 2001.

[GGPZ17] Zeeshan Ghodsi, Sanjam Garg, Raluca Ada Popa, and Matei Zaharia. Safetynets: Verifiable execution of deep neural networks on an untrusted cloud. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[GKR08] Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for muggles. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, STOC, pages 113–122, Victoria, British Columbia, Canada, 2008.

[GLR+91] Peter Gemmell, Richard Lipton, Ronitt Rubinfeld, Madhu Sudan, and Avi Wigderson. Self-testing/correcting for polynomials and for approximate functions. In *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, STOC '91, page 33–42, New York, NY, USA, 1991. Association for Computing Machinery.

[GWHD25] Albert Garreta, Hendrik Waldner, Katerina Hristova, and Luca Dall'Ava. Zinc: Succinct arguments with small arithmetization overheads from IOPs of proximity to the integers. Cryptology ePrint Archive, Paper 2025/316, 2025.

[HL25] Horace He and Thinking Machines Lab. Defeating nondeterminism in llm inference. *Thinking Machines Lab: Connectionism*, 2025. https://thinkingmachines.ai/blog/defeating-nondeterminism-in-llm-inference/.

[HLB01] Y. Hida, X.S. Li, and D.H. Bailey. Algorithms for quad-double precision floating point arithmetic. In *Proceedings 15th IEEE Symposium on Computer Arithmetic. ARITH-15 2001*, pages 155–162, 2001.

[Hol19] Justin Holmgren. On round-by-round soundness and state restoration attacks. Cryptology ePrint Archive, Paper 2019/1261, 2019.

[KDZ24] Daniel Kang, Tri Dao, and Matei Zaharia. ZKML: An optimizing system for ML inference in zero-knowledge proofs. *Proceedings of the ACM SIGMOD Conference*, 2024.

[KMS99] Marcos Kiwi, Frédéric Magniez, and Miklos Santha. Approximate testing with relative error. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, STOC '99, page 51–60, New York, NY, USA, 1999. Association for Computing Machinery.

[KMS02] Marcos Kiwi, Frédéric Magniez, and Miklos Santha. *Exact and Approximate Testing/Correcting of Algebraic Functions: A Survey*, pages 30–83. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[KMS03] Marcos Kiwi, Frédéric Magniez, and Miklos Santha. Approximate testing with error relative to input size. *Journal of Computer and System Sciences*, 66(2):371–392, 2003.

[Lab24] Modulus Labs. Scaling intelligence: Verifiable decision forest inference with remainder. `https://github.com/Modulus-Labs/Papers/blob/master/remainder-paper.pdf`, February 2024.

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

[Mag00] Frédéric Magniez. Multi-linearity self-testing with relative error. In Horst Reichel and Sophie Tison, editors, *STACS 2000*, pages 302–313, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[Mei13] Or Meir. IP=PSPACE using error-correcting codes. *SIAM Journal on Computing*, 42(1):380–403, 2013.

[REK96] R. Rubinfeld, F. Ergun, and S.R. Kumar. Approximate checking of polynomials and functional equations. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, page 592, Los Alamitos, CA, USA, 1996. IEEE Computer Society.

[Rem36] Evgeny Yakovlevich Remez. Sur une propriété des polynômes de Tchebycheff. *Comm. Inst. Sci. Kharkow*, 13:93–95, 1936. English translation available at `https://history-of-approximation-theory.com/fpapers/remeztrans.pdf`.

[RR22] Noga Ron-Zewi and Ron D. Rothblum. Proving as fast as computing: Succinct arguments with constant prover overhead. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1353–1363. ACM, 2022.

[RR24] Noga Ron-Zewi and Ron D. Rothblum. Local proofs approaching the witness length. *Journal of the ACM*, 71(3):18:1–18:42, 2024.

[RRR16] Omer Reingold, Guy N. Rothblum, and Ron D. Rothblum. Constant-round interactive proofs for delegating computation. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 49–62. ACM, 2016.

[RS97] Jonathan Richard Shewchuk. Adaptive precision floating-point arithmetic and fast robust geometric predicates. *Discrete and Computational Geometry*, 18(3):305–363, 1997.

[Rub91] Ronitt A. Rubinfeld. *A mathematical theory of self-checking, self-testing and self-correcting programs*. PhD thesis, Berkeley, USA, 1991. UMI Order No. GAX91-26752.

[Sch80] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *J. ACM*, 27(4):701–717, October 1980.

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

[SV22] Eduardo Soria-Vazquez. Doubly Efficient Interactive Proofs over Infinite and Non-commutative Rings. In Eike Kiltz and Vinod Vaikuntanathan, editors, *Theory of Cryptography*, pages 497–525, Cham, 2022. Springer Nature Switzerland.

[Tha22] Justin Thaler. Proofs, arguments, and zero-knowledge. *Foundations and Trends in Privacy and Security*, 4(2–4):117–660, 2022.

[Tha25] Justin Thaler. Sum-check is all you need: An opinionated survey on fast provers in snark design. *Cryptology ePrint Archive*, 2025.

[TY20] Sergey Tikhonov and Peter Yuditskii. Sharp Remez Inequality. *Constructive Approximation*, 52(2):233–246, Oct 2020.

[V+22] Alejandro Vázquez et al. Doubly-efficient interactive proofs over infinite and non-commutative rings. In *ITCS*, 2022.

[Yom11] Yosef Yomdin. Generalized Remez Inequality for (s, p)-Valent Functions, 2011.

[Zip79] Richard Zippel. Probabilistic algorithms for sparse polynomials. In Edward W. Ng, editor, *Symbolic and Algebraic Computation*, pages 216–226, Berlin, Heidelberg, 1979. Springer Berlin Heidelberg.

[ZKP22] ZKProof. Zkproof community reference. version 0.3. `https://docs.zkproof.org/pages/reference/versions/ZkpComRef-0-3.pdf`, July 2022. Updated versions at `https://docs.zkproof.org/reference`.

A Numerical soundness analysis

As noted in the proof of Theorems 4.4 and 4.7, the bound on soundness error stated is a loose upper bound on the actual soundness error. Here we provide a Sage script that numerically calculates a more precise upper bound on the soundness error for various choices of parameters.

```
# sage script for the number of interactive bits of security
# for an approx-sum-check (assuming sufficiently large sample set)

from scipy.optimize import minimize_scalar

def T(x, d):
    return cosh(d * arccosh(x))

def R(x, d):
    return T((2 / x) - 1, d)

def C(x, d):
    return T(1 / sin(pi * x / 2), d)

def objective(t, d, v, k, F):
    if t <= 0:
        return float('inf')
    try:
        ival = numerical_integral(F(x, d)^t, 0, 1)[0]
    except Exception:
        return float('inf')
    if ival <= 0:
        return float('inf')
    return float(t * (log(v, 2) - k) + v * log(ival, 2))

# parameters
degree = 4
num_vars = 25
log_delta_ratio = 300 # log_2 of the ratio of Delta to delta
kappa = C # Use R for reals, C for complex, or your own distance function

# optimization
res = minimize_scalar(objective, bounds=(0, 1), method='bounded',
                      args=(degree, num_vars, log_delta_ratio, kappa))

# result
print(f"t = {res.x}")
print(f"Bits of security = {-res.fun}")
```

B Explicit malicious prover strategy

Consider a malicious prover in the classical sum-check protocol, that is one who begins with an incorrect claim for the sum. In round k, either the prover has fully convinced the verifier and can send honest messages q_k for the remainder of the protocol, or it must send a dishonest message p_k and hope that the verifier picks a point r_k where $p_k(r_k) = q_k(r_k)$. The notion of forward progress in fooling the verifier is binary, and there is no p_k the prover can pick which improves its chances of a future dishonest p_k fooling the verifier.

In the approximate sum-check protocol, things are very different. It is possible for the prover to strategically pick malicious messages which make steady progress, in expectation, towards fooling the verifier. We

discuss such a strategy here and show how it roughly matches the soundness bounds provided in Section 4.3.

To begin, in Remark 1, we alluded to a family of polynomials for which Theorem 2.4 is tight. Explicitly, consider the arc in $E_s \in \mathbb{T}$ from $e^{(1-s)i\pi}$ to $e^{(s-1)i\pi}$ (centered on -1) for a constant $s \in [0,1]$. This arc has size $\mu(E_s)$. Given this arc, the degree d polynomial [TY20]

$$\mathfrak{p}_{s,d}(x) = \mathfrak{p}_d\left(e^{iz}\right) = e^{i(dz/2)}\mathfrak{T}_d\left(\csc\left(\frac{\pi}{2}\mu(E_s)\right)\cos\left(\frac{z}{2}\right)\right)$$

is bounded by 1 on E_s, takes on its max value at $x = 1$, and makes the inequality in Theorem 2.4 an equality

$$\|\mathfrak{p}_{s,d}\|_{\mathbb{T}} = \mathfrak{T}_d\left(\csc\left(\frac{\pi}{2}\mu(E_s)\right)\right)\|\mathfrak{p}_{s,d}\|_{E_s}.$$

The family of polynomials emerges from considering all values of d and scaling and rotating $\mathfrak{p}_d$.

Given this family of polynomials, we can formulate an inverse Theorem 4.3.

Theorem B.1. *Let d and n be a positive integers, let $c \leq h$ be non-negative real numbers, let e^{iz} be an arbitrary point in $\mathbb{T}$, let $r \sim U(\Omega_n)$, and let $\rho \sim U(0,1)$. There exists a degree d polynomial of the form $p(x) := \sum_{k=0}^{d} a_k x^k$ with complex coefficients such that $|p(e^{iz})| = \|p\|_{\mathbb{T}} = h$ and*

$$\Pr_r[|p(r)| \leq c] \geq \Pr_\rho[\|p\|_{\mathbb{T}} \cdot \kappa(\rho) \leq c] - 1/n$$

where κ is defined as in Expression (11)*.*

Proof. Consider the polynomial $\mathfrak{p}_{s,d}$ as defined above with the set E_s with

$$s = \frac{2}{\pi}\csc^{-1}\left(\mathfrak{T}_d^{-1}\left(\frac{h}{c}\right)\right).$$

Let $p(x) = \mathfrak{p}_{s,d}(e^{iz}x)/h$. By the properties of the family of polynomials above, E_s is an arc, $\forall x \in E_s\ |p(x)| \leq c$, and it is that case that

$$\|p\|_{\mathbb{T}} = \mathfrak{T}_d\left(\csc\left(\frac{\pi}{2}\mu(E_s)\right)\right)\|p\|_{E_s}.$$

Any arc I of size $\mu(I)$, it must contain at least $n\mu(I) - 1$ roots of unity. It follows from the fact that E_s is an arc that

$$\Pr_r[|p(r)| \leq c] = \Pr_r[r \in E_s] = \frac{\#(\Omega_n \cap E_s)}{n} \geq \mu(E_s) - \frac{1}{n}.$$

By the definition of E_s we have

$$\Pr_r[|p(r)| \leq c] \geq \frac{2}{\pi}\csc^{-1}\left(\mathfrak{T}_d^{-1}\left(\frac{\|p\|_{\mathbb{T}}}{c}\right)\right) - \frac{1}{n} = \Pr_\rho\left[\frac{2}{\pi}\csc^{-1}\left(\mathfrak{T}_d^{-1}\left(\frac{\|p\|_{\mathbb{T}}}{c}\right)\right) \geq \rho\right] - \frac{1}{n}.$$

The proof concludes by rewriting the inside of the probability and is identical to the final step in proving Theorem 4.3

$$\Pr_\rho\left[\frac{2}{\pi}\csc^{-1}\left(\mathfrak{T}_d^{-1}\left(\frac{\|p\|_{\mathbb{T}}}{c}\right)\right) \geq \rho\right] = \Pr_\rho[\|p\|_{\mathbb{T}} \cdot \kappa(\rho) \leq c].$$

□ □

With this inverse Theorem 4.3 in hand, it is possible run the soundness arguments from the opposite direction, that is to provide a prover strategy which results in a lower bound on soundness. Here we walk through a simplified form of the argument in the context of round-by-round soundness which can be refined further with additional work, but suffices to illustrate the point.

Consider the protocol in Section 3 with $\#(B) \leq d$, and Suppose that the prover begins the protocol with a claim H such that

$$\left|H - \sum_{b_1 \in B} q_1(b_1)\right| = \Delta.$$

That is the claimed sum differs from the true sum by Δ.

A classical grinding attack would have the prover pick p_1 polynomials until $p_1(r_1)$ and $q_1(r_1)$ were sufficiently close. Such an attack might involve picking p_1 such that the difference $p_1 - q_1$ is roughly equal to a rescaled $\mathfrak{p}_{s,d}$ and then slightly perturbing it (to sample different r_1 values). This implies $\|p_1 - q_1\|_{\mathbb{T}} \leq \Delta$. By Theorem B.1 we have

$$\begin{aligned}\Pr_r\left[|p_1(r) - q_1(r)| \leq \frac{\delta}{\#(B)^v}\right] &\geq \Pr_\rho\left[\|p_1 - q_1\|_{\mathbb{T}} \cdot \kappa(\rho) \leq \frac{\delta}{\#(B)^v}\right] - \frac{1}{n} \\ &\geq \Pr_\rho\left[\Delta \cdot \kappa(\rho) \leq \frac{\delta}{\#(B)^v}\right] - \frac{1}{n} \\ &= \Pr_\rho\left[\kappa(\rho) \leq \frac{\delta}{\Delta\,\#(B)^v}\right] - \frac{1}{n}.\end{aligned}$$

and this probability can be explicitly evaluated.

However, the prover also has the option of doing a smaller amount of grinding in each round, making progress step by step. Informally, instead of attempting to get a lucky ρ such that $\kappa(\rho) \leq \frac{\delta}{\Delta\,\#(B)^v}$, the prover can attempt to get a sequence of ρ_i such that $\prod \kappa(\rho_i) \leq \frac{\delta}{\Delta\,\#(B)^v}$. One way to do that is to ensure that for each ρ_i

$$\kappa(\rho_i) \leq \left(\frac{\delta}{\Delta\,\#(B)^v}\right)^{1/v}.$$

Formally, suppose that in round k,

$$|p_{k-1}(r_{k-1}) - q_{k-1}(r_{k-1})| \leq \Delta \cdot \left(\frac{\delta}{\Delta\,\#(B)^v}\right)^{(k-1)/v}.$$

In that case, the prover can pick a p_k such that we have that

$$\begin{aligned}&\Pr\left[|p_k(r_k) - q_k(r_k)| \leq \Delta \cdot \left(\frac{\delta}{\Delta\,\#(B)^v}\right)^{k/v}\right] \\ &\geq \Pr\left[\Delta \cdot \left(\frac{\delta}{\Delta\,\#(B)^v}\right)^{(k-1)/v} \cdot \kappa(\rho_k) \leq \Delta \cdot \left(\frac{\delta}{\Delta\,\#(B)^v}\right)^{k/v}\right] - \frac{1}{n} \\ &= \Pr\left[\kappa(\rho_k) \leq \left(\frac{\delta}{\Delta\,\#(B)^v}\right)^{1/v}\right] - \frac{1}{n} \\ &= \kappa^{-1}\left(\#(B)\left(\frac{\delta}{\Delta}\right)^{1/v}\right) - \frac{1}{n}.\end{aligned}$$

Given that for $x \in [0,1]$ $\kappa(x) \leq \frac{\pi}{2}x^d$, this is bounded below by

$$\frac{\pi}{2}\sqrt[d]{\frac{1}{\#(B)}}\sqrt[vd]{\frac{\delta}{\Delta}} - \frac{1}{n}.$$

Now, since $\#(B) \leq d$, $\sqrt[d]{\frac{1}{\#(B)}} \geq \sqrt[d]{\frac{1}{d}} \geq \frac{2}{\pi}$, so this is in turn bounded below by

$$\sqrt[vd]{\frac{\delta}{\Delta}} - \frac{1}{n}.$$

This effectively provides a lower bound on the round-by-round soundness, giving us that.

$$\sqrt[vd]{\frac{\delta}{\Delta}} - \frac{1}{n} \leq \epsilon_{RBR} \leq \frac{d}{n} + 2\sqrt[vd]{\frac{\delta}{\Delta}}.$$

This demonstrates that, up to small constant factors, ϵ_{RBR} is tight!

C Simple soundness analysis

The standard proof of soundness for the classical sum-check protocol considers, in each round, the probability that the verifier selects a r_k such that $p_k(r_k) = q_k(r_k)$. This is $d/|S|$ by Schwartz–Zippel (Theorem 4.1). By taking the union bound over all v rounds, the resulting soundness error is at most $vd/|S|$.

It is natural to consider applying the same proof technique to the approximate sum-check protocol. Here we show that, while it can be applied, it results in a significantly larger precision requirement.

This is a high-level walk-through of the argument.

Consider a variant of the protocol described in Figure 2 where all computations and checks are exact up until the final check (to remove small additive factors which do not impact the higher order results here).

Given an integral domain $\mathbb{U}$, suppose we have that, for every $p, q \in \mathbb{U}_d[x]$

$$\Pr_{r \sim U(S)}\left[D(p(r), q(r)) \leq \frac{1}{c_1} \cdot \frac{D(\sum_{b\in B} p(b), \sum_{b\in B} q(b))}{\#(B)}\right] < c_2$$

for two constants c_1 and c_2.

If the prover lies about the initial sum by an additive factor at least c_1^v/δ then the probability that the verifier accepts is at most v/c_2. A short proof of this is as follows.

First consider a prover which makes a lie just smaller than $\frac{c_1^v}{\delta}$. If, in each round k,

$$D(p_k(r_k), q_k(r_k)) > \frac{1}{c_1} \cdot \frac{D(\sum_{b\in B} p_k(b), \sum_{b\in B} q_k(b))}{\#(B)},$$

the tightest possible lower bound on the final check is

$$D(p_v(r_v), q_v(r_v)) \geq \frac{\delta}{\#(B)^v}.$$

This leaves open the possibility of a verifier accepting.

Next, supposing instead that the lie was equal to c_1^v/δ, it is the case that

$$D(p_v(r_v), q_v(r_v)) > \frac{\delta}{\#(B)^v}.$$

This implies that, c_1^v/δ is the smallest lie for which the prover's success depends on, in at least one round, getting a lucky r_k from the verifier such that

$$D(p_k(r_k), q_k(r_k)) \leq \frac{1}{c_1} \cdot \frac{D(\sum_{b\in B} p_k(b), \sum_{b\in B} q_k(b))}{\#(B)}.$$

By a union bound over all rounds, the probability of such an event occurring is at most v/c_2. In the case of $\mathbb{R}$, by application of Theorem 4.2, $c_1 \approx \mathfrak{T}_d(2c_2 - 1)$. For 1/2 soundness, given this analysis, it must be the case that $c_2 \geq 2v$. Thus we have $c_1 \approx \mathfrak{T}_d(4v - 1)$.

By somewhat tedious algebraic manipulation, we have that

$$\Delta \in O((4v)^{O(vd)}\delta)$$

(concretely this is approximately $2^{v(2d+1)}v^{vd}$). Recalling from Section 3 that $\delta \in O(d\,\#(B)^v\varepsilon)$ and treating $\#(B)$ as a constant which is at least 2, we have the result

$$\Delta \in O((4v)^{O(vd)}d\varepsilon).$$

This is significantly worse than the $O(vd2^{O(vd)}\varepsilon)$ bound that follows from the tighter analysis in Theorem 4.4.