Relaxed Locally Correctable Codes with Nearly-Linear Block Length and Constant Query Complexity*

Alessandro Chiesa
alexch@berkeley.edu
UC Berkeley

Tom Gur†
tom.gur@warwick.ac.uk
University of Warwick

Igor Shinkar
ishinkar@sfu.ca
Simon Fraser University

Abstract

Locally correctable codes (LCCs) are codes $C\colon \Sigma^k \to \Sigma^n$ which admit local algorithms that can correct any individual symbol of a corrupted *codeword* via a minuscule number of queries. One of the central problems in algorithmic coding theory is to construct $O(1)$-query LCC with minimal block length. Alas, state-of-the-art of such codes requires exponential block length to admit $O(1)$-query algorithms for local correction, despite much attention during the last two decades.

This lack of progress prompted the study of *relaxed LCCs*, which allow the correction algorithm to *abort* (but not err) on small fraction of the locations. This relaxation turned out to allow constant-query correction algorithms for codes with polynomial block length. Specifically, prior work showed that there exist $O(1)$-query relaxed LCCs that achieve nearly-quartic block length $n = k^{4+\alpha}$, for an arbitrarily small constant $\alpha > 0$.

We construct an $O(1)$-query relaxed LCC with *nearly-linear* block length $n = k^{1+\alpha}$, for an arbitrarily small constant $\alpha > 0$. This significantly narrows the gap between the lower bound which states that there are no $O(1)$-query relaxed LCCs with block length $n = k^{1+o(1)}$. In particular, this resolves an open problem raised by Gur, Ramnarayan, and Rothblum (ITCS 2018).

Keywords: algorithmic coding theory; sublinear algorithms; tensor codes; consistency tests using random walks

*Extended abstract of this work was presented at the ACM-SIAM 31st Symposium on Discrete Algorithms (SODA 2020).
†Tom Gur is supported by UKRI Future Leaders Fellowship MR/S031545/1.

Contents

1 Introduction **3**
- 1.1 Main result . . . 4
- 1.2 Discussion . . . 4
- 1.3 Organization . . . 5

2 Techniques **5**
- 2.1 Challenges for reducing block length . . . 5
- 2.2 Building block I: consistency checks using random walks . . . 6
- 2.3 Building block II: relaxed-correctable PCPs . . . 7
- 2.4 A composition theorem using CTRW . . . 8
- 2.5 Composing tensored Reed–Solomon codes with relaxed-correctable PCPs . . . 9

3 Preliminaries **10**
- 3.1 Basic coding theory . . . 11
- 3.2 Relaxed locally correctable codes . . . 11
- 3.3 Canonical PCPs of proximity . . . 12
- 3.4 Relaxed-correctable PCPs . . . 13

4 Consistency tests using random walks **13**
- 4.1 Composition theorem using CTRW . . . 14
- 4.2 Local correction algorithms for composed codes . . . 15
- 4.3 Analysis of the algorithm . . . 17

5 CTRW for tensor codes **19**

6 Relaxed-correctable canonical PCPPs **21**

7 Proof of main theorem **23**

References **24**

A Proof of Theorem 1 for binary codes **26**

1 Introduction

Locally correctable codes (LCCs) are error-correcting codes that exhibit local-to-global phenomena, allowing for correction of individual symbols of a noisy codeword via a small number of queries. More precisely, a code $C\colon \Sigma^k \to \Sigma^n$ is an LCC with correcting radius τ if there exists a probabilistic algorithm, called *local corrector*, that is given an index $i \in [n]$ and query access to an input $w \in \Sigma^n$ such that if w is τ-close to a valid codeword $C(x)$, and outputs $C(x)_i$ (the i-th bit of $C(x)$) with high probability.

The notion of local codes, such as LCCs and the closely-related notion of locally decodable codes (LDCs), are central to algorithmic coding theory. Indeed, their study led to progress in several areas of theoretical computer science, including complexity theory, program checking, data structures, and cryptography (see surveys [Tre04; Yek12; KS17] and references therein), and also led to practical applications in distributed storage systems (notably, to Microsoft Azure [Hua+12]).

Unfortunately, the state of the art of LCC demands a steep price for the redundancy required to obtain good locality. Despite two decades of extensive study, the best construction of $O(1)$-query LCCs is obtained via a parameterization of the Reed–Muller code [Mul54], where the block length is *exponential* in the message length of the code. This state of affairs is even worse than for $O(1)$-query LDC, for which *sub-exponential* block length was achieved using a highly non-trivial construction of matching vector codes [Yek08; Efr12].

This lack of progress prompted the study of relaxed LCCs [GRR18] (following the relaxation of LDCs that was introduced in the seminal work of Ben-Sasson, Goldreich, Harsha, Sudan, and Vadhan [Ben+06]). This natural relaxation arguably captures the essence of local correction, yet allows for more efficient constructions: an exponential improvement in block length. In a recent line of works [GGK15; GG16; BDG17; GG18; GR18; Blo+18; GL20; RR19] relaxed LDCs and relaxed LCC have been studied and used to obtain applications to data structures [CGW09], PCPs [MR10; DH13], property testing [CG18], and probabilistic proofs [GGK15; GR17; GR18].

Loosely speaking, this relaxation of LCCs allows the local corrector to *abort* (i.e., output "don't know") on a small fraction of the indices while still avoiding correction errors. More accurately, a *relaxed LCC* $C\colon \Sigma^k \to \Sigma^n$ with *correcting radius* τ and *success rate* ρ is a code that admits a randomized algorithm, called the *local corrector*, which receives an oracle access to a string $w \in \Sigma^n$ and (a direct access to) an index $i \in [n]$. The corrector makes a small number of queries to w and is required to satisfy the following conditions:[1]

1. *Completeness:* If the input is a valid codeword (i.e., $w = C(x)$), the corrector must always output $C(x)_i$.
2. *Relaxed correction:* If w is τ-close to a codeword $C(x)$, with high probability, the corrector must either output $C(x)_i$ or a special *abort* symbol $\perp$ (indicating it detected a corruption and is unable to correct it).

This seemingly modest relaxation turns out to allow using powerful tools from the PCP literature. Using such techniques, it was recently shown that there exist $O(1)$-query relaxed LCCs with polynomial block length [GRR18]; more precisely, with nearly quartic block length $n = k^{4+\alpha}$, for an arbitrarily small constant $\alpha > 0$. On the other hand, it was recently shown that there *cannot* exist $O(1)$-query relaxed LCCs with block length $n = k^{1+o(1)}$ [GL20],[2] so we know that the block length must be super-linear.

It is natural to ask whether it is possible to reduce block length, ideally as close as possible to the lower bound, while preserving $O(1)$-query relaxed local correction. In particular, in [GRR18] the following open problem was raised:

Do there exist $O(1)$-query relaxed LCCs with nearly-linear block length?

[1] By standard transformations [Ben+06], the following two conditions imply a "success rate" property: the corrector will only output $\perp$ on a ρ-fraction of the coordinates, for an arbitrarily small constant $\rho > 0$.

[2] The bound in [GL20] is stated for relaxed LDCs, but it extends to relaxed LCCs via standard arguments (see, e.g., [BGT17]).

1.1 Main result

Our main result provides a strong positive answer to the above open problem, giving a simple, explicit construction of $O(1)$-query relaxed LCCs with block length $n = k^{1+\alpha}$, for an arbitrarily small constant $\alpha > 0$.

Theorem 1 (Informal, see Theorem 7.1)**.** *For every $q \in \mathbb{N}$ and field $\mathbb{F}$ there exists a $O(q)$-query relaxed LCC $C\colon \mathbb{F}^K \to \mathbb{F}^N$ with constant relative distance and decoding radius that has block length*

$$N = q^{O(\sqrt{q})} \cdot K^{1+O(1/\sqrt{q})} \ .$$

Note that the dependency of the block length in the query complexity allows a tradeoff ranging from block length $N = K^{1+\alpha}$ with query complexity $O(1/\alpha^2)$ for constant $\alpha > 0$, and down to block length $N = K \cdot 2^{\tilde{O}(\sqrt{\log(K)})}$ with query complexity $q = \log(K)/\log\log(K)$.

1.2 Discussion

Below we make several remarks that highlight several aspects of our results.

Matching state-of-the-art of relaxed LDCs. The notion of relaxed locally *correctable* codes (relaxed LCCs) is closely related to that of relaxed locally *decodable* codes (relaxed LDCs), wherein the goal is to locally decode a symbol of the *message* rather than locally correct a symbol of the *codeword*. Obtaining the stronger notion of relaxed local correctability is, in general, more challenging than obtaining relaxed local decodability.

Our relaxed LCCs not only achieve constant query complexity and nearly-linear length, but rather *exactly* match the parameters of the relaxed *LDC* in [Ben+06] (achieving block length $N = K^{1+\alpha}$ and query complexity $O(1/\alpha^2)$ for any constant $\alpha > 0$), which remains the state-of-the-art for over 15 years.

In addition, a recent work [GGK15] achieved a construction of $O(1)$-query, nearly-linear length relaxed LDC, which are also strongly locally testable. Our construction of a relaxed LCC can also be made strongly locally testable, via standard techniques at the cost of increasing the query complexity by a constant factor (see discussion in Section 3). Our work thus extends the result in [GGK15] to the setting of relaxed LCCs.

In sum, our work implies that the efficiency of state-of-the-art constructions of relaxed LDCs can also be achieved by the stronger notion of relaxed LCCs. Note that the lower bound of [GL20], stating that any q-query RLCC must have block length $N \geq K^{1+\Omega(1/q^2)}$, allows room for improving both upper and lower bounds. This discussion raises several natural open problems that we leave for future work.

Open Problem 1. *Is it possible to obtain an RLCC with a better block length vs query complexity tradeoff?*

Open Problem 2. *Is there a separation between the parameters that can be achieved by relaxed LCC and by relaxed LDC?*

Open Problem 3. *Do there exist non-trivial RLCCs in the list correcting regime?*

Open Problem 4. *Is it possible to obtain RLCCs with improved block length by allowing poly-logarithmic query complexity?*

Application to PCPs. A key component that we introduce in our proof of Theorem 1 is the notion of *relaxed-correctable PCPs* (cPCP). Loosely speaking, these PCPs can be thought of as extending the decodable PCPs (dPCP) of Dinur and Harsha [DH13] to the setting of local correctability, allowing for recovering any symbol of the proof, rather than just the witness encoded in it. See Section 3.4 for a precise definition.

We observe that the relaxed LCCs in Theorem 1 can be used to derive the first construction of a cPCP with nearly-linear length and query complexity $O(1)$ (see Section 6). We find this result to be of independent interest, and due to the importance of dPCPs in facilitating PCP composition, it is possible that cPCPs will be useful in making progress on related problems.

On the alphabet of relaxed LCCs In the main body of the paper we prove Theorem 1 only for large alphabet (yet sublinear in the block length K). It is quite straightforward (though a bit tedious) to adapt our techniques to obtain a relaxed LCC with the same tradeoff between the query complexity and the block length for binary alphabet. The details on the binary alphabet appear in Appendix A.

1.3 Organization

The rest of the paper is organized as follows. In Section 2 we present a high-level overview of the proof of Theorem 1. In Section 3 we provide the necessary preliminaries. In Section 4 we introduce the notion of *consistency tests using random walks* and prove a composition theorem for codes that admit such tests. In Section 5 we prove a theorem which states that tensor codes admit consistency tests using random walks. In Section 6, we provide a new construction of canonical PCPs of proximity that are relaxed locally correctable. Finally, in Section 7 we combine these building blocks and prove Theorem 1 by composing tensored Reed–Solomon codes with relaxed-correctable PCPs, via the composition theorem.

2 Techniques

Our starting point is the construction of the relaxed LCC in [GRR18], which below we refer to as "GRR". We begin by briefly recalling the high-level approach underlying GRR and the bottlenecks that lead to a nearly quartic block length (i.e., $N = K^{4+\alpha}$ for any small constant $\alpha > 0$).

We then outline our new construction, which at a high level can be viewed as a length efficient variant of [GRR18]. We introduce the notions of *consistency check using random walks* (CTRW) and *relaxed correctable PCPs*, which underlie our construction and deem to be of independent interest. We show that tensor codes admit CTRW, and we construct new relaxed-correctable PCPs. Then, we prove a composition theorem using CTRW, and apply it to compose tensored Reed–Solomon codes with our relaxed-correctable PCPs.

2.1 Challenges for reducing block length

To present the challenges in obtaining $O(1)$-query relaxed LCCs with near-linear length, we first briefly recall the previous state of the art; namely, the construction in [GRR18].

The relaxed LCC in GRR is obtained via the paradigm of *PCP composition* [AS98] by composing (i) a relaxed LCC C with large query complexity with (ii) a special type of PCP (more accurately, special type of PCP of proximity). The result of the composition is a relaxed LCC C' that (roughly) inherits the query complexity of the PCP, with a controlled overhead in block length.

The codewords of the composed code C' are constructed by taking each codeword of the relaxed LCC C and concatenating it with a PCP for each local view that the relaxed local corrector for C could query, asserting that this local view is one that would lead the corrector to successfully recover the value at the desired coordinate.[3]

A straightforward argument shows that the foregoing approach allows to locally correct the bits of the original relaxed LCC C without querying the entire local view that would have been queried by the original corrector. Instead, we invoke the PCP verifier with respect to that local view and the corresponding PCP. The relaxed corrector for the composed code C' takes the queries made by the original corrector as input, and uses the PCP verifier to test that this local view would have lead the original corrector to output the right value.

However, more involved machinery is required to correct the PCP part of C'. This is the main reason why constructing relaxed LCC is significantly harder than constructing relaxed *LDC*, for which $O(1)$-query decoding with nearly-linear block length was already shown in [Ben+06]. Indeed, the quartic blowup in the GRR construction originates from the mechanism for correcting the part of the composed code C' that consists of PCPs. In more detail, to correct the PCPs in the code in GRR, it was observed that both:

1. the Reed–Muller based PCP [BFL91; Bab+91], which has polynomial length and polylogarithmic query complexity, and
2. the Hadamard based PCP [Aro+98], which has exponential length and constant query complexity,

can be made locally correctable if they are restricted to linear languages, since in essence, these PCPs are based on codes which are (non-relaxed) LCCs themselves.[4] However, known PCP composition techniques do not preserve this property, and so we still do not know of polynomial-length PCP constructions that are locally correctable using $O(1)$ queries. (This is not surprising as, of course, we do not know of any polynomial-length LCC with $O(1)$ queries.)

Thus, to obtain $O(1)$-query relaxed LCC with polynomial block length, GRR starts with the Reed–Muller code, and composes it with the self-correctable PCP with polynomial length and polylogarithmic query complexity, and then performs another composition with the self-correctable PCP with exponential length and constant query complexity. It is not hard to see that each such composition entails a quadratic blowup (e.g., for composing with the Reed–Muller local corrector, we have to provide a PCP for each of the possible $O(n^2)$ lines), and thus these two compositions result in a quartic blow up.

In order to avoid this blowup and obtain nearly-linear length, we will introduce and construct a new type of PCP, which allows us to reduce the number of compositions to just a single one. In addition, we show a new composition theorem which relies on the notion of consistency checks using random walks to perform a more efficient composition that does not incur a quadratic overhead.

Our relaxed LCC is constructed via a composition theorem that takes codes that admit "consistency tests using random walks" and composes them with a special type of "relaxed-correctable" PCPs to obtain the relaxed LCC with the desired parameters. We begin by discussing the former component.

2.2 Building block I: consistency checks using random walks

We introduce the notion of *consistency tests using random walks* (CTRW). These are structured local tests that assert the consistency of global object with an individual point inside it. We will later capitalize on the specific structural properties of CTRW to obtain an efficient composition.

[3] For this idea to work, several important details must be addressed: the codeword of C should be replicated many times for C' to have distance, the relaxed LCC must be robust, and others. However, for the sake of simplicity, we ignore these issues here.

[4] Several other properties were required for the GRR construction, such as strong soundness, canonicity, linearity, and robustness. We discuss the ones that are also relevant to us in Section 2.3.

In our setting, we define CTRW as follows. An error correcting code $C \subseteq \Sigma^n$ admits a *consistency test using a random walk* if there exists a test, which gets a word $w \in \Sigma^n$ that is close to C and a coordinate $i \in [n]$, that checks that the symbol w_i is consistent with the rest of the codeword closest to w, by performing a random walk over intersecting local constraints as follows.

At each step of the random walk, we check that the current constraint is satisfied, and that it is consistent with the previous constraint on their intersection. That is, the test chooses a sequence of local constraints on the purported codeword such that the $(j+1)$-th constraint involves coordinates that intersect those involved in the j-th constraint, and checks that w satisfies all local constraints, in which case it declares that w_i is globally consistent. See Section 4 for the precise definition of CTRW.

We require that on a valid codeword, the test will pass with probability 1, and that given a codeword that is "close" (typically, within distance that is proportional to the distance of the code) to a valid codeword, with high probability not only should the test reject, but also that the local view of the test should be "far" from an accepting view (i.e., satisfying the *robustness* condition).

When constructing a CTRW, to minimize the query complexity of the test, we want the constraints to be of small locality, as well as for the random walk to converge to the uniform distribution after a small number of steps. Note that there is tension between these two properties. In addition, observe that this notion implies *relaxed* local correctability with appropriate parameters.

Building on techniques from [GGK15], in Section 5 we prove that the structural properties of tensor codes admit consistency tests using random walks. We provide a refined analysis of the robustness of these tests, as in our application we require the codes to have relative distance close to 1.

Theorem 2.1 (informal, see Theorem 5.1)**.** *Let $C\colon \mathbb{F}^k \to \mathbb{F}^n$ be an arbitrary linear code of relative distance δ_C, and let $C^{\otimes m}$ be the m-dimensional tensor product of C. Then, $C^{\otimes m}$ admits an CTRW that given a word that is τ-close to a valid codeword, has ρ-robust soundness $\epsilon = (\delta_C - 2\rho)^m - \tau$. Furthermore, all predicates defined by the CTRW for $C^{\otimes m}$ check that a restriction of a given tensor word to a line belongs to C.*

2.3 Building block II: relaxed-correctable PCPs

The second component in our composition theorem is a *relaxed-correctable PCP*, which is a special type of PCP where proof oracles can be relaxed locally corrected, similarly to relaxed LCCs. Informally, for a language L, a relaxed-correctable PCP system consists of PCP $\pi(x)$ for each $x \in L$, and an algorithm that on input that gets an input x, a purported proof $\tilde{\pi}$ of length $|\pi(x)|$, and a coordinate $i \in [|\pi(x)|]$. The algorithm makes a small number of queries to the purported proof $\tilde{\pi}$ and satisfies the following guarantees. If $\tilde{\pi}_i = \pi(x)_i$, then the algorithm returns $\pi(x)_i$, and if $\tilde{\pi}$ is close to some $\pi(x)$, then with high probability, the algorithm either outputs $\pi(x)_i$ or a special *abort* symbol $\perp$.

For technical reasons, the relaxed-correctable PCPs that we need must additionally be *canonical PCPs of proximity* [GS06]. Informally, this means that the PCP satisfies two additional requirements:

- for every true statement there exists a unique canonical proof that the verifier is required to accept, and;
- the verifier is required to reject any pair of statement and proof with probability that is roughly proportional to its distance from a true statement and its corresponding canonical proof.

These properties are important for local codes as they create a one-to-one correspondence between valid statements and valid proofs. We discuss how these properties are used in our composition theorem in the next subsection. For now, we merely note that PCPs of proximity (PCPPs) [Ben+06] where introduced to facilitate PCP composition, and immediately proved to be a powerful tool for local codes, since on the one hand, in the

setting of local codes we cannot afford reading an entire PCP oracle, whereas on the other hand the distance property of codes is often compatible with the approximated decision problems that PCPPs can deal with.

Relying on recent progress on canonical PCPPs [DGG19; Par20], we provide new, simple constructions of *relaxed-correctable* $O(1)$-query canonical PCPPs of polynomial length.

Theorem 2.2 (informal, see Theorem 6.1)**.** *For any language L in the class $\mathcal{P}$, there exists a canonical relaxed-correctable PCP of proximity with polynomial length, soundness $1/2$, and query complexity $O(1)$. Furthermore, the PCP oracle is relaxed-correctable with $O(1)$ queries.*

In order to construct a relaxed-correctable PCPP we compose the relaxed LCC in [GRR18] (which has polynomial block length and query complexity $O(1)$) with state-of-the-art canonical PCPPs of polynomial block length and query complexity $O(1)$. We stress that this composition is in the *reversed direction* compared to our main composition theorem. Namely, we compose a PCPP with a relaxed LCC to obtain a relaxed-correctable PCPP.[5]

2.4 A composition theorem using CTRW

We outline our composition theorem, which takes a code that admits CTRW (Component I, see Section 2.2) and composes it with relaxed-correctable canonical PCPP (Component II, see Section 2.3) to obtain a relaxed LCC that (roughly) inherits the query complexity from the PCPP, at the cost of increasing the block length by a multiplicative sublinear factor. (See Theorem 4.3 for the exact parameters that our composition theorem obtains.) Later on, in Section 2.5, we explain how to use our composition theorem, together with the components that we constructed, to obtain our main result.

Our composition theorem roughly follows the outline of the composition theorem in [GRR18], however, it is optimised to make use of the structural properties of consistency test via random walks, as well as on the properties of our new PCPP. It, thus, suffices to perform a single composition to obtain a $O(1)$-query relaxed LCC of nearly-linear length (rather than two compositions for quartic length, as in [GRR18]).

Let C be a code that admits a CTRW with constraint set $\mathcal{C} = \{\mathcal{C}_j\}_{j\in[n]}$, where each $\mathcal{C}_j$ is a collection of constraints that depend on the j'th coordinate. Let π the encoding function of a relaxed-correctable canonical PCPP. We compose C with π by constructing a new code C', where for each message x the corresponding codeword consists of two parts:

1. a *core* that consists of (repetitions of) the original encoding $C(x)$, where the number of repetitions asserts that the core dominates the size of the composed codeword, and thus provides distance;
2. a *PCP bundle* that for each coordinate $j \in [n]$ and constraint $\mathcal{P} \in \mathcal{C}_j$ includes a PCPP $\pi_{(j,\mathcal{P})}$ which asserts that $C(x)$ satisfies the constraint $\mathcal{P}$.

Correcting a point in the core of an alleged codeword $C(x)$ is fairly straightforward. We perform a random walk over intersecting constraints in a randomly chosen copy of the original codeword $C(x)$ in the composed codeword $C'(x)$, and check each constraint by invoking the corresponding PCPP verifier, rather than querying the entire local view that the constraint refers to. The robustness of the CTRW ensures that this preserves the soundness condition. Correcting the PCP bundle is more involved, however, as we discuss next.

The naive approach for correcting a point p in the PCP bundle of a purported codeword $C'(x)$ is by relying on the relaxed-correctablity of the PCPP that we use and invoking the relaxed corrector on the relevant PCPP oracle to recover p. Unfortunately, this does *not* suffice, as we discuss next.

[5] It might be interesting to note the total depth of the composition: our relaxed LCC indeed contains PCP that themselves contain encodings of the (weaker) relaxed LCC in [GRR18], which yet again contains copies of simpler PCP that are based on (non-relaxed) LCC.

Let π' be the purported PCPP oracle that contains the point p. The fact that π' is relaxed locally correctable tells us that if π' is close to a valid PCPP $\pi_{(j,\mathcal{P})}$, we can correct (or detect corruption and abort) any of the points it contains, including p. However, we encounter two problems: (i) if the purported PCPP oracle π' is far from a valid encoding (which it might, since the size of each PCPP oracle is negligible with respect to the total size of the codeword), there is no guarantee regarding the success of the relaxed corrector. (ii) Even if π' is a perfectly valid PCPP encoding, it might still not be an encoding that is consistent with the rest of the word. Indeed, note that since the size of π' is negligible, one can replace it with a completely different PCPP oracle while remaining within the correcting radius.

This is where the power of canonical PCPP, wherein only the canonical proof for a correct statement is accepted, kicks in. To deal with the first problem, note that if π' is far from a valid encoding, then it is also far from the prescribed canonical proof. Thus, the canonical PCPP verifier will reject, and the relaxed corrector can return $\perp$.

Dealing with the second problem requires a bit more work. Suppose that π' is some perfectly valid PCPP encoding (or close to one) that is not consistent with the rest of the codeword. Recall that π' is a proof stating that some local constraint on the core of the composed code C' is satisfied. Denote this local view of the core by w. If w is not corrupted (or only slightly corrupted), then it is far from being consistent with π', and so again, we detected a corruption, and we can safely output $\perp$. However, how can we detect a corruption if w was also heavily corrupted such that it matches (the corrupted) π'? The crux is that w is a subset of the *core* of C', which we already showed how to correct using CTRW. Since w is heavily corrupted, we can choose a random point in w, invoke the relaxed-corrector on it, which will find that the random point in inconsistent with the rest of the word.

2.5 Composing tensored Reed–Solomon codes with relaxed-correctable PCPs

We are now ready to describe the construction behind Theorem 1. We start with the Reed–Solomon code parameterized such that its distance is roughly $1 - 1/m$, for a constant $m \in \mathbb{N}$ to be determined later. We consider the m-th tensor power of the Reed–Solomon code, which we denote by $\mathsf{RS}^{\otimes m}$, and denote its block length by n. The reason we think of $\mathsf{RS}^{\otimes m}$ as a tensor code, rather than as the standard Reed–Muller code, is because we wanted a length efficient consistency test using random walks, which only makes use of the $O(mn)$ axis-parallel lines out of all possible $O(n^2)$ lines. Details follow.

By Theorem 2.1, we know that $\mathsf{RS}^{\otimes m}$ admits a CTRW. For concreteness, we spell out this CTRW. The constraints correspond to the consistency of each coordinate p of $\mathsf{RS}^{\otimes m}$ with each axis-parallel line ℓ that is incident with p. To check the consistency of a point p with the global codeword, we perform a random walk of roughly m steps (without repeating directions). That is, we first check the constraint that corresponds to p and a random line ℓ_1 that is incident with it. Then we choose a random point p_1 on ℓ_1 and check the constraint that corresponds to p_1 and a random line ℓ_2 that is incident with it, and so on. Theorem 2.1 guarantees that if p is inconsistent with the global codeword, then this test succeeds with high probability.

We stress that we use the tensor of the Reed–Solomon code, rather than an arbitrary asymptotically good code, in order to increase the robustness, which would, in turn, reduce the query complexity in the composition theorem. To get some intuition for that, note that if we use a code with distance, say $1/4$, then each line incident with p has distance $1/4$ from the correct codeword. Hence, it is possible that the line ℓ_1 can be changed in at most $1/4$ fraction of the coordinates and to become consistent with p_1 and consistent with the global codeword. Similarly, if the chosen point p_1 is inconsistent with the global codeword, then any line ℓ_2 can be changed in at most $1/4$ fraction of the coordinates to become consistent with p_1 and consistent with the global codeword. Thus implies that the CTRW will catch an inconsistency with probability only $1/4^m$. On the other hand, when tensoring the Reed–Solomon code, which has distance roughly $1 - 1/m$,

this probability becomes $(1-1/m)^m$, which is a constant independent of m.

Finally, we invoke our composition theorem with respect to $\mathsf{RS}^{\otimes m}$ (and its corresponding CTRW that is implied by Theorem 2.1), whose block length we denote by n, together the $O(1)$-query canonical relaxed-correctable PCPP from Theorem 2.2 with proofs of polynomial length. By applying the composition theorem we obtain a relaxed corrector with query complexity $O(m^2)$. As for the length, the composition adds a PCPP oracle of size $n^{O(1/m)}$ for each one of the $m \cdot n$ axis-parallel lines. Hence the total block length of the construction is roughly $m \cdot n \cdot n^{O(1/m)} = O(mn^{1+(1/m)})$, which gives us the desired block length by setting a sufficiently large constant m.

3 Preliminaries

We begin with standard notation. The relative distance between two strings $x, y \in \Sigma^n$ is defined as

$$\operatorname{dist}(x,y) \coloneqq |\{i \in [n] : x_i \neq y_i\}|/n \ .$$

If $\operatorname{dist}(x,y) \leq \epsilon$, we say that x is *ϵ-close* to y; otherwise we say that x is *ϵ-far* from y. For a non-empty set $S \subseteq \Sigma^n$ define the distance of x from S as $\operatorname{dist}(x,S) \coloneqq \min_{y \in S} \operatorname{dist}(x,y)$. If $\operatorname{dist}(x,S) \leq \epsilon$, we say that x is *ϵ-close* to S; otherwise we say that x is *ϵ-far* from S.

We will also need a more general notion of a distance, allowing different coordinates to have different weight. In particular, we will need the distance that gives constant weight to one of the coordinates, and spreads the rest of the weight uniformly between all coordinates.

Definition 3.1. *Fix $n \in \mathbb{N}$ and an alphabet Σ. For a coordinate $k \in [n]$ define the distance dist_k between two strings $x, y \in \Sigma^n$ as*

$$\operatorname{dist}_k(x,y) = \frac{\mathbf{1}_{[x_k \neq y_k]}}{2} + \frac{\{i \in [n] : x_i \neq y_i\}}{2n} \ .$$

That is, if $x_k \neq y_k$, then $\operatorname{dist}_k(x,y)$ is at least $1/2$.

Analogously to the distance with respect to the uniform measure, we define dist_k between a string $x \in \Sigma^n$ and a set $S \subseteq \Sigma^n$ as

$$\operatorname{dist}_k(x,S) = \min_{y \in S} \operatorname{dist}_k(x,y) \ .$$

Let $\mathcal{P} \colon \Sigma^n \to \{0,1\}$ be a predicate. Typically, the predicate will depend on a small number of coordinates (a local view), in which case we denote by $\operatorname{Dom}(\mathcal{P})$ the set of coordinates on which $\mathcal{P}$ depends, and identify $\mathcal{P}$ with $\mathcal{P} \colon \Sigma^{\operatorname{Dom}(\mathcal{P})} \to \{0,1\}$. Finally, we denote by $\mathsf{sat}(\mathcal{P}) = \{y \in \Sigma^{\operatorname{Dom}(\mathcal{P})} : \mathcal{P}(y) = 1\}$ the set of assignments satisfying $\mathcal{P}$. Then, for $x \in \Sigma^{\operatorname{Dom}(\mathcal{P})}$ we have

$$\operatorname{dist}(x, \mathsf{sat}(\mathcal{P})) = \min_{y \in \Sigma^{\operatorname{Dom}(\mathcal{P})} : \mathcal{P}(y)=1} \frac{\{i \in \operatorname{Dom}(\mathcal{P}) : x_i \neq y_i\}}{|\operatorname{Dom}(\mathcal{P})|} \ ,$$

and analogously

$$\operatorname{dist}_k(x, \mathsf{sat}(\mathcal{P})) = \min_{y \in \Sigma^{\operatorname{Dom}(\mathcal{P})} : \mathcal{P}(y)=1} \frac{\mathbf{1}_{[x_k \neq y_k]}}{2} + \frac{\{i \in \operatorname{Dom}(\mathcal{P}) : x_i \neq y_i\}}{2|\operatorname{Dom}(\mathcal{P})|} \ .$$

We will also need the following concentration of measure inequality.

Theorem 3.2 (McDiarmid's inequality)**.** *Let $X_1, X_2, \ldots, X_n$ be independent random variables such that, for all i we have $a_i \leq X_i \leq b_i$, $c_i \coloneqq b_i - a_i \leq C$. Let $S_n \coloneqq \sum_{i=1}^n X_i$ be their sum, $E_n \coloneqq \mathrm{E}[S_n] = \sum_{i=1}^n \mathrm{E}[X_i]$ be its expected value, and $V_n \coloneqq \operatorname{Var}[S_n] = \sum_{i=1}^n \operatorname{Var}[X_i]$ be its variance. Then,*

$$\Pr\left[|S_n - E_n| > t\right] < 2\exp\left(-\frac{t^2/2}{V_n + C \cdot t/3}\right).$$

3.1 Basic coding theory

Let $k < n$ be positive integers, and let Γ, Σ be two alphabets. A *code* $C\colon \Gamma^k \to \Sigma^n$ is an *injective* mapping from messages of length k, over the alphabet Γ, to codewords of length n, over the alphabet Σ. Typically it is the case that $\Gamma = \Sigma$, in which case we simply say that the code C is over the alphabet Σ. The *message length* of the code is k, its *block length* is n (which we view as a function of k), and its *rate* is k/n. The *relative distance* of the code is the minimum, over all distinct messages $x, y \in \Gamma^k$, of $\mathrm{dist}(C(x), C(y))$. We sometimes abuse notation and use C to denote the set of all of its codewords, $\{C(x)\}_{x\in\Gamma^k} \subseteq \Sigma^n$.

Linear codes. Let $\mathbb{F}$ be a finite field. A code $C\colon \mathbb{F}^k \to \mathbb{F}^n$ is *linear* if it is a $\mathbb{F}$-linear map from $\mathbb{F}^k$ to $\mathbb{F}^n$. In this case the set of codewords C is a subspace of $\mathbb{F}^n$. A basic result about linear codes [Jus72] is that there exist (explicit) constructions of linear codes that are *binary* ($\mathbb{F} = \mathbb{F}_2$) and *good* (have constant rate and constant relative distance).

Concatenation. Code concatenation is an operation on codes typically used to reduce the alphabet size. Fix alphabets Σ, Ξ, Γ. Fix an *outer* code $C\colon \Gamma^k \to \Xi^n$ with relative distance δ_C and rate r_C, and an *inner* code $D\colon \Xi \to \Sigma^r$ with relative distance δ_D and rate r_D. The concatenation of C with D is the code $C_{comp}\colon \Gamma^k \to \Sigma^{r\cdot n}$ such that each $x \in \Gamma^k$ is first encoded with C, and then each symbol of the resulting codeword is encoded with D. The relative distance of C_{comp} is $\delta_C \cdot \delta_D$ and the rate is $r_C \cdot r_D$.

Focusing on linear codes, we have the following fact. Let $\mathbb{F}$ be a field, and $\mathbb{G}$ an extension of $\mathbb{F}$; so that $\mathbb{G} \cong \mathbb{F}^m$ for some $m \in \mathbb{N}$. Let $C\colon \mathbb{F}^k \to \mathbb{G}^n$ and $D\colon \mathbb{G} \to \mathbb{F}^r$ be codes that are $\mathbb{F}$-linear (we identify $\mathbb{G}$ with $\mathbb{F}^m$). Then the code $C_{comp}\colon \mathbb{F}^k \to \mathbb{F}^{r\cdot n}$ obtained by concatenating C and D is $\mathbb{F}$-linear.

Tensor product. Let $C\colon \Gamma^k \to \Sigma^n$ be a code with rate r and relative distance δ, and let $m \in \mathbb{N}$. The *tensor product code* $C^{\otimes m}\colon \Gamma^{k^m} \to \Sigma^{n^m}$ is the code with message length k^m, block length n^m, rate r^m, and relative distance δ^m that comprises all functions $c\colon \Gamma^{k^m} \to \Sigma^{n^m}$ whose restriction to any axis-parallel line is in C. Namely, for every $j \in \{1, \ldots, m\}$ and $a_1, \ldots, a_{j-1}, a_{j+1}, \ldots, a_m \in \Sigma^k$, the function $c'\colon \Gamma^k \to \Sigma^n$ defined by $c'(i) \coloneqq c(a_1, \ldots, a_{j-1}, i, a_{j+1}, \ldots, a_m)$ belongs to C.

3.2 Relaxed locally correctable codes

Following the discussion in the introduction, we provide a formal definition of relaxed LCCs, and state some basic facts and known results.

Definition 3.3 (Relaxed LCC)**.** *Let $C \subseteq \Sigma^n$ be an error correcting code with relative distance δ, and let $q \in \mathbb{N}$, $\tau_{\mathrm{cor}} \in (0, \delta/2)$, $\rho \in (0, 1)$, and $\epsilon \in (0, 1]$ be parameters. Let $\mathcal{D}$ be a randomized algorithm that gets an oracle access to an input $w \in \Sigma^n$ and an explicit access to an index $i \in [n]$. We say that $\mathcal{D}$ is a* q-local relaxed correction algorithm for C with correction radius τ_{cor} and soundness ϵ *if for all inputs the algorithm $\mathcal{D}$ reads explicitly the coordinate $i \in [n]$, reads at most q (random) coordinates in w, and satisfies the following conditions.*

1. *For every $w \in C$, and every coordinate $i \in [n]$ it holds that $\Pr[\mathcal{D}^w(i) = w_i] = 1$.*
2. *For every $w \in \Sigma^n$ that is τ_{cor}-close to some codeword $c \in C$ and every coordinate $i \in [n]$ it holds that $\Pr[\mathcal{D}^w(i) \in \{c_i, \perp\}] \geq \epsilon$, where $\perp \notin \Sigma$ is a special* abort *symbol.*

The code C is said to be a $(\tau_{\mathrm{cor}}, \epsilon)$-relaxed locally correctable code (RLCC) *with query complexity q if it admits a* q-query relaxed local correction algorithm with correction radius τ_{cor} and soundness ϵ.

The following remark explains why in our setting we can omit the third condition of Definition 3.3.

Remark 3.4 (On success rate)**.** For relaxed LCC with query complexity $O(1)$ it is known that, via standard transformations [Ben+06], the third condition of relaxed LCCs follows directly from the first two conditions in Definition 3.3. Since all of our constructions have query complexity $O(1)$, we can restrict our attention to only the first two conditions.

It would be also useful to note the following trivial observation.

Observation 3.5. *Note that if C is a $(\tau_{\mathrm{cor}}, \epsilon)$-RLCC, then every subset of C is also an RLCC with the same parameters.*

Finally, we will need the following theorem of Gur, Ramnarayan, and Rothblum [GRR18].

Theorem 3.6 ([GRR18])**.** *There exists an explicit construction of a systematic binary linear $(\tau_{\mathrm{cor}}, \epsilon)$-RLCCs with constant relative distance, block length $n = \mathrm{poly}(k)$, query complexity $q = O(1)$, with correction radius $\tau_{\mathrm{cor}} = \Omega(1)$, and soundness $\epsilon = \Omega(1)$*

3.3 Canonical PCPs of proximity

A *canonical PCP of proximity* strengthens the notion of a *PCP of proximity* (PCPP), which in turn strengthens the notion of a *PCP*. We recall these latter notions before formally defining canonical PCPs of proximity. For an intuitive discussion of these notions, see Section 2.3.

PCPs. A PCP for a language L is a polynomial-time randomized oracle algorithm V that receives direct access to an input x and oracle access to a proof π. The algorithm V is allowed to make a small number of queries to π such that the following holds: for every $x \in L$ there exists a proof π such that $\Pr[V^{\pi}(x) = 1] = 1$, and for every $x \notin L$ and every proof π it holds that $\Pr[V^{\pi}(x) = 1] \leq 1/2$.

PCP of proximity. A PCP of proximity for a language L, with respect to proximity parameter ρ, is a polynomial-time randomized oracle algorithm V that receives oracle access to *both* an input x and a proof π. The algorithm V is allowed to make a small number of queries to x and to π such that the following holds: for every $x \in L$ there exists a proof π such that $\Pr[V^{\pi}(x) = 1] = 1$, and for every x that is ρ-far from the language L and every proof π it holds that $\Pr[V^{\pi}(x) = 1] \leq \epsilon_{\mathrm{PCPP}}$ for some soundness parameter $\epsilon_{\mathrm{PCPP}} < 1$.

Canonical PCPs of proximity. A *canonical PCPP* is a PCPP in which every instance in the language has a unique (thus canonical) accepting proof.

Definition 3.7 (canonical PCPP)**.** *A canonical PCPP for a language $L \subseteq \Sigma^*$ with soundness ϵ_{PCPP} with respect to proximity parameter ρ, is a polynomial-time randomized oracle algorithm V satisfying the following conditions with respect to some polynomial $\ell \colon \mathbb{N} \to \mathbb{N}$.*

- Canonical completeness: *For every $x \in L$ there exists a* unique *(canonical) proof $\pi^*(x) \in \Sigma^{\ell(|x|)}$ for which* $\Pr[V^{x,\pi^*(x)} = 1] = 1$.
- Canonical soundness: *For every $x' \in \Sigma^n$ and proof $\pi' \in \Sigma^{\ell(|x|)}$ such that*

$$\delta(x', \pi') \triangleq \min_{x \in \Sigma^n} \left\{ \max \left(\frac{\mathrm{dist}(x', x)}{n} \; ; \; \frac{\mathrm{dist}(\pi', \pi^*(x))}{\ell(n)} \right) \right\} > \rho \ , \tag{1}$$

it holds that $\Pr[V^{x',\pi'} = 1] \leq \epsilon_{\mathrm{PCPP}}$.

Above, for any $x \notin L$ we define $\pi^*(x) = \perp$ and say that any proof π' is 1-far from $\perp$.

The polynomial ℓ denotes the length of the canonical proof in the PCPP, and its query complexity is the maximum number of queries that V makes to the instance and its (supposed) proof.

In this paper we use the following theorem due to Dinur, Goldreich, and Gur [DGG19]. (Alternatively, we could have used the construction by Paradise [Par20].)

Theorem 3.8 ([DGG19])**.** *Let $\rho > 0$ be a proximity parameter. For every language in $L \in \mathcal{P}$ there exists a polynomial $\ell\colon \mathbb{N} \to \mathbb{N}$ and a canonical PCPP verifier for L satisfying the following properties.*

1. *For all $x \in L$ of length $|x| = n$ the length of the canonical proof $\pi(x)$ is $|\pi(x)| = \ell(n)$.*
2. *The query complexity of the PCPP verifier is $q = O(1/\rho)$.*
3. *The PCPP verifier for L has perfect completeness and soundness $\epsilon = 1/2$ for proximity parameter ρ (with respect to the uniform distance measure).*

3.4 Relaxed-correctable PCPs

We introduce a new notion of PCPs, to which we refer to as *relaxed-correctable PCPs* (cPCP). A cPCP is a PCP system that is both locally checkable and relaxed locally correctable; that is, in addition to allowing local verification of the validity of the given proof like a standard PCP, a cPCP also allows for relaxed local correction of the PCP oracle itself.

The notion of cPCP is closely related to the decodable PCPs (dPCP) introduced by Dinur and Harsha [DH13], which play an important role in PCP composition. In fact, cPCPs can be thought of as extending the definition of dPCPs to the setting of local correctability, as they allow recovery of any symbol in the PCP oracle, rather than just the encoded witness. We remind that similarly to our case, dPCPs admit *relaxed* decoding proceedures, i.e., the decoder is allowed to abort (output, say, $\perp$) in case it detects corruption.

In this work we focus on canonical PCPs of proximity, and so we define relaxed-correctable canonical PCPPs; in short ccPCPPs.

Definition 3.9. *[relaxed locally correctable cPCP of proximity] A language $L \subseteq \Sigma^*$ is said to admit a ccPCPP with query complexity q and soundness ϵ_{PCPP} with respect to the proximity parameter ρ for the canonical proof $\pi(\cdot)$, and correcting soundness ϵ_{RLCC} for correcting radius τ_{cor} if it satisfies the following conditions.*

1. *There exists a is a q-query canonical PCPP for L that satisfies the conditions in Definition 3.7 with soundness ϵ_{PCPP} with respect to the proximity parameter ρ for the canonical proof $\pi(\cdot)$.*
2. *The code $\Pi_L = \{w \circ \pi(w) : w \in L\}$ is a $(\tau_{\mathrm{cor}}, \epsilon_{RLCC})$-RLCC with query complexity q.*

4 Consistency tests using random walks

We define the notion of *consistency tests using random walks*. Intuitively, given an error correcting code $C \subseteq \Sigma^n$, a consistency test using a random walk gets a word $w \in \Sigma^n$ that is close to some codeword $c^* \in C$, and a coordinate $i \in [n]$. The test checks that the symbol w_i is equal to c^*_i, i.e., w_i is consistent with the codeword closest to w. To this end, the test chooses a sequence of local constraints on the purported codeword such that the $(j+1)$-th constraint involves coordinates that intersect those involved in the j-th constraint, and checks that w satisfies all local constraints, in which case it declares w_i is globally consistent.

We note that this notion implies *relaxed* local correctability with appropriate parameters (see Remark 4.2).

Definition 4.1. *[Consistency test using random walks] Let $C \subseteq \Sigma^n$ be an error correcting code. A (q,t)-consistency test using random walks (CTRW) for C is a randomized algorithm that gets as input a string $w \in \Sigma^n$ and a coordinate $i \in [n]$. For each coordinate $j \in [n]$ CTRW defines a collection of constraints $\mathcal{C}_j$ such that each predicate $\mathcal{P}\colon \Sigma^n \to \{0,1\}$ in $\mathcal{C}_j$ depends on at most q coordinates. The test works as follows.*

Algorithm 1 Consistency test using random walks

Input: $w \in \Sigma^n, i \in [n]$

1: $k_1 = i$
2: **for** $r = 1$ **to** t **do**
3: Sample a predicate $\mathcal{P}_r \in \mathcal{C}_{k_r}$ according to a distribution $\mathcal{D}_r$ that may depend on the previous steps
4: Let $\mathrm{Dom}_r = \mathrm{Dom}(\mathcal{P}_r)$ be the set of coordinates on which $\mathcal{P}_r$ depends
5: Sample $k_{r+1} \sim \mathrm{Dom}_r$ uniformly at random
6: **end for**
7: Read w in the coordinates $\cup_{r\in[t]}\mathrm{Dom}_r$
8: **if** $P_r(w_{|\mathrm{Dom}_r}) = 1\ \forall r \in [t]$ **then**
9: **return** ACCEPT
10: **else**
11: **return** REJECT
12: **end if**

We say that CTRW has perfect completeness and (τ,ρ,ϵ)-robust soundness if it satisfies the following guarantees.

Perfect completeness: *If $w \in C$, then $\Pr[\mathsf{CTRW}^w(i) = ACCEPT] = 1$ for all $i \in [n]$.*

(τ,ρ,ϵ)-robust soundness: *If w is τ-close to some $c^* \in C$, but $w_i \neq c^*_i$, then*

$$\Pr[\exists r \in [t] \text{ such that } \mathrm{dist}_{k_r}(w_{|\mathrm{Dom}_r}, \mathsf{sat}(\mathcal{P}_r)) \geq \rho] \geq \epsilon \ .$$

Here dist_{k_r} is defined as $\mathrm{dist}_{k_r}(w_{|\mathrm{Dom}_r}, c_{|\mathrm{Dom}_r}) = \frac{1}{2}\cdot\mathbf{1}_{[w_{k_r}\neq c_{k_r}]} + \frac{\{\ell\in\mathrm{Dom}_r : w_\ell \neq c_\ell\}}{2|\mathrm{Dom}_r|}$, and $\mathrm{dist}_{k_r}(w_{|\mathrm{Dom}_r}, \mathsf{sat}(\mathcal{P}_r)) = \min_{c:\mathcal{P}_r(c)=1}\{\mathrm{dist}_{k_r}(w_{|\mathrm{Dom}_r}, c_{|\mathrm{Dom}_r}\}$. See Definition 3.1 and the following discussion in Section 3.

Remark 4.2. Note that if a code C admits a (q,t)-CTRW with perfect completeness and (τ,ρ,ϵ)-robust soundness for any $\rho > 0$, then it is an (τ,ϵ)-RLCC with query complexity qt. Indeed, given a word $w \in \Sigma^n$ and a coordinate $j \in [n]$ the local correction algorithm for C runs the (q,t)-CTRW on input (w,j). The algorithm outputs w_j if CTRW accepts, and outputs $\perp$ otherwise. We omit the straightforward details.

4.1 Composition theorem using CTRW

Below we prove that if a code admits a (q,t)-CTRW then it can be composed with an appropriate PCPP to obtain an RLCC with query complexity $O(t)$.

Theorem 4.3 (Composition theorem)**.** *Consider the following components.*

- Outer code: *A code $C_{base}\colon \Sigma^k \to \Sigma^n$ with the following properties.*
  1. *C_{base} admits a (n',t)-CTRW with perfect completeness and $(\tau,\rho,\epsilon_{RW})$-robust soundness.*
  2. *For each $j \in [n]$ and each $\mathcal{P} \in \mathcal{C}_j$ it holds that $\mathsf{sat}(\mathcal{P})$ has distance $\delta_{\mathcal{P}} \geq 4\rho$.*
  3. *$|\mathcal{C}_j| = s$ for all $j \in [n]$.*

- Inner PCPP: *A q_{PCPP}-query canonical PCPP system for each $\mathcal{P} \in \mathcal{C}_j$ for all $j \in [n]$ with the following properties.*

  1. *For all $x \in \mathsf{sat}(\mathcal{P})$ of length n' the length of the canonical proof is $\ell(n')$.*
  2. *The PCPP verifier for $\mathsf{sat}(\mathcal{P})$ has soundness ϵ_{PCPP} for proximity parameter $\rho \leq \delta_{\mathcal{P}}/4$ with respect to the distance measure dist_j.*
  3. *The code $\Pi_{\mathcal{P}} = \{w \circ \pi(w) : w \in \mathsf{sat}(\mathcal{P})\}$ is a (τ, ϵ_{in})-RLCC with query complexity q_{PCPP}.*

Then, there exists a code $C_{comp} \colon \Sigma^k \to \Sigma^N$ for $N = 2n \cdot s \cdot \ell(n')$. The code C_{comp} is a $(\tau_{\mathrm{cor}}, \epsilon_{RLCC})$-RLCC with query complexity $(t+1) \cdot q_{\mathrm{PCPP}}$, where the decoding radius of C_{comp} is $\tau_{\mathrm{cor}} = \tau/4$ and the soundness is $\epsilon_{RLCC} = \min(\frac{\epsilon_{RW} \cdot (1-\epsilon_{\mathrm{PCPP}}) \cdot \delta_{\mathcal{P}}}{2}, \frac{\epsilon_{in}}{4})$.

Constructing the composed code C_{comp}: Given the components in the statement of Theorem 4.3 the composed code $C_{comp} \colon \Sigma^k \to \Sigma^N$ is obtained by combining C_{base} with the PCPP as follows. Each $c^* \in C_{base}$ corresponds to a unique codeword $c \in C_{comp}$. The codeword c consists of two parts $c = c_{base} \circ \Pi$ defined as follows.

1. c_{base} consists of $s \cdot \ell(n')$ repetitions of c^*. The repetition of c^* is somewhat artificial, and is done so that this part will constitute a constant fraction of the length of the codeword c. The exact reason for this will be clear in the proof of correctness.

2. Π is the concatenation of proofs of proximity $\pi_{(j,\mathcal{P})}$ (as per the inner PCPP in the hypothesis of the theorem) for each $j \in [n]$ and each constraint $\mathcal{P} \in \mathcal{C}_j$ asserting that $c_{|\mathrm{Dom}(\mathcal{P})} \in \mathsf{sat}(\mathcal{P})$ (or rather $c_{|\mathrm{Dom}(\mathcal{P})}$ is close to $\mathsf{sat}(\mathcal{P})$ with respect to the distribution dist_j). That is, each $\pi_{(j,\mathcal{P})}$ is a proof asserting that (i) the restriction of some $w \in \Sigma^n$ to $\mathrm{Dom}(\mathcal{P})$ satisfies $\mathcal{P}$, and that (ii) the value w_j is indeed the correct value.

   Remark 4.4. Note that if every two assignments in $\mathsf{sat}(\mathcal{P})$ are $\delta_{\mathcal{P}}$-far from each other, then requiring that w has the "correct" value in the j-th coordinate is non-trivial in the following sense: for any w such that $w_{|\mathrm{Dom}(\mathcal{P})} \in \mathsf{sat}(\mathcal{P})$ and w' obtained from w by changing only the j-th coordinates it holds that $\mathrm{dist}_j(w_{|\mathrm{Dom}(\mathcal{P})}, w'_{|\mathrm{Dom}(\mathcal{P})}) \geq 0.5$ and $\mathrm{dist}_j(w'_{|\mathrm{Dom}(\mathcal{P})}, \mathsf{sat}(\mathcal{P})) \geq \Omega(\delta_{\mathcal{P}})$. Hence, no proof will convince the PCPP verifier that w' is close to $\mathsf{sat}(\mathcal{P})$ with high probability.

Parameters of C_{comp}: For the block length of $C_{comp} \colon \Sigma^k \to \Sigma^N$ note that each codeword $c' \in C_{comp}$ consists of $s \cdot \ell(n') \cdot n$ symbols (of the $s \cdot \ell(n')$ copies) of the base codeword $c \in \Sigma^n$, concatenated with $n \cdot s$ proofs of proximity, each of length $\ell(n')$. Therefore, total block length of C_{comp} is $2n \cdot s \cdot \ell(n')$.

4.2 Local correction algorithms for composed codes

Before presenting the correction algorithm for C_{comp} we first consider a (slightly simpler) code that contains only one copy of C_{base} and all proofs as in Item 2 of the definition of C_{comp}. For this code we describe a local correction algorithm for correcting the symbols in the part corresponding to the base code.

Algorithm 2 Local correction for the base part without repetitions

Input: $w^* \in \Sigma^n, \Pi = (\pi_{j,\mathcal{P}})_{j\in[n],\mathcal{P}\in\mathcal{C}_j} \in (\Sigma^{\ell(n')})^{ns}, i^* \in [n]$

1: Run the CTRW for C_{base} on input (w^*, i^*)
2: Let $\mathcal{P}_1, \ldots, \mathcal{P}_t$ be the constraints sampled by the CTRW, and let $k_1, \ldots, k_t$ be the coordinates defined by $\mathcal{P}_r$'s
3: **for** $r = 1$ **to** t **do**
4: Run the PCPP verifier on $\pi_{(k_r,\mathcal{P}_r)}$ to check that $w^*_{|\mathrm{Dom}_r}$ is close to $\mathsf{sat}(\mathcal{P}_r)$ with respect to dist_{k_r}
5: **end for**
6: **if** all iterations of Step 4 accept **then**
7: **return** $w^*_{i^*}$
8: **else**
9: **return** $\perp$
10: **end if**

Claim 4.5. *Let $w^* \in \Sigma^n, \Pi = (\pi_{j,\mathcal{P}})_{j\in[n],\mathcal{P}\in\mathcal{C}_j} \in \Sigma^{ns\ell(n')}$ and $i^* \in [n]$ be input to Algorithm 2. Suppose that the base part of w^* is τ_{cor}-close to some base codeword $c^* \in C_{base}$, but $w^*_{i^*} \neq c^*_{i^*}$. Then Algorithm 2 will output $\perp$ with probability at least $\epsilon_{RW} \cdot (1 - \epsilon_{\mathrm{PCPP}})$.*

Proof. If we choose the constraints as per Step 2 of Algorithm 2, then with probability at least ϵ_{RW} there will be some $r \in [t]$ such that $\mathrm{dist}_{k_r}(w_{|\mathrm{Dom}_r}, \mathsf{sat}(P_r)) \geq \rho$. If that happens, then the PCPP verifier applied to the predicate P_r in Step 4 will accept with probability at most ϵ_{PCPP}, and thus the algorithm will output $\perp$ with probability at least $\epsilon_{RW} \cdot (1 - \epsilon_{\mathrm{PCPP}})$. □

We are now ready to present the local correction algorithm for C_{comp}. The algorithm has two cases depending on whether (1) the coordinate corresponds to (one of the copies of) a symbol in the base code, or (2) it belongs to the PCPP part $\pi_{i^*,\mathcal{P}}$ for some $i^* \in [n]$ and $\mathcal{P} \in \mathcal{C}_j$.

1. Suppose first that the coordinate $i \in [N]$ that corresponds to a coordinate of the base code. In this case the local correction algorithm works as follows.

Algorithm 3 Local correction for the base part of C_{comp}

Input: $w \in \Sigma^N, i \in [N]$

1: Let $i^* \in [n]$ be the coordinate of C_{base} that corresponds to the coordinate $i \in [N]$
2: Sample $r \in [s \cdot \ell(n')]$ uniformly at random
3: Let $w^* \in \Sigma^n$ be the substring of w whose coordinates correspond to the r-th copy of the base codeword
4: Let $\Pi = (\pi_{(j,\mathcal{P})})_{j\in[n],\mathcal{P}\in\mathcal{C}_j}$ be the part of w corresponding to the concatenation of all proofs of proximity
5: **return** Algorithm 2 on the input (w^*, Π, i^*)

2. Suppose now that $i \in [N]$ is a coordinate that belongs to a proof of proximity $\pi_{(i^*,\mathcal{P})}$ for some coordinate i^* corresponding to a symbol from the base code, and a constraint $\mathcal{P} \in \mathcal{C}_{i^*}$ of the form $\mathcal{P} \colon \Sigma^{\mathrm{Dom}(\mathcal{P})} \to \{0, 1\}$. In this case the local correction algorithm works as follows.

Algorithm 4 Local correction for the PCPP part of C_{comp}

Input: $w \in \Sigma^N, i \in [N]$

1: Let $i^* \in [n]$ and $\mathcal{P} \colon \Sigma^{\mathrm{Dom}(\mathcal{P})} \to \{0,1\}$ be the coordinate and the constraint corresponding to the proof containing the i-th coordinate
2: Sample $r \in [s \cdot \ell(n')]$ uniformly at random
3: Let $w^* \in \Sigma^n$ be the substring of w whose coordinates correspond to the r-th copy of the base codeword
4: Run the PCPP verifier for $\mathcal{P}$ on $\pi_{(i^*,\mathcal{P})}$ to check that $\mathrm{dist}_{i^*}(w^*_{|\mathrm{Dom}(\mathcal{P})}, \mathsf{sat}(\mathcal{P})) \leq \rho$
5: **if** Step 4 rejects **then**
6: **return** $\perp$
7: **end if**
8: Let $\Pi = (\pi_{(j,\mathcal{P})})_{j \in [n], \mathcal{P} \in \mathcal{C}_j}$ be the part of w corresponding to the concatenation of all proofs of proximity
9: Run Algorithm 2 on the input (w^*, Π, j') for a uniformly random coordinate j' of $w^*_{|\mathrm{Dom}(\mathcal{P})}$
10: **if** Step 9 returns $\perp$ **then**
11: **return** $\perp$
12: **end if**
13: Run the local corrector of the inner PCPP on $(w^*_{|\mathrm{Dom}(\mathcal{P})} \circ \pi_{(i^*,\mathcal{P})})$ to correct the i^*-th coordinate
14: **return** the value obtained in Step 13

Query complexity: It is clear that the query complexity is upper bounded by that of Algorithm 4. The total number of queries in upper bounded by (i) q_{PCPP} queries in Step 4, (ii) at most $t \cdot q_{\mathrm{PCPP}}$ queries in Step 9, and (iii) at most q_{RLCC} queries in Step 13.

4.3 Analysis of the algorithm

It is obvious that given a codeword $w \in C_{comp}$ and an index $i \in [N]$ the local correction always returns the correct answer w_i.

Suppose now that the input is some $w \in \Sigma^N$ that is τ_{cor}-close to some codeword $c \in C_{comp}$. We show below that $\Pr[ALG^w(i) \in \{c^*_i, \perp\}] \geq \epsilon_{RLCC}$.

Let c^* be the base codeword of c, that is, c is obtained from repetitions of c^* concatenated with the corresponding canonical proofs. Denote by $w_{base} \in \Sigma^{ns\ell(n')}$ the restriction of w to the coordinates containing the $s \cdot \ell(n')$ repetitions of the base codeword. Since w is τ_{cor}-close to C_{comp}, it follows that w_{base} is $2\tau_{\mathrm{cor}}$-close to $s \cdot \ell(n')$ repetitions of some base codeword $c^* \in C_{base}$. Denote by W_{close} the event that $\mathrm{dist}(w^*, c^*) \leq 4\tau_{\mathrm{cor}} = \tau$, where w^* is a substring of w corresponding to a random copy of the basecode. Then $\Pr[W_{\mathrm{close}}] \geq 1/2$.

Consider the two cases depending on the coordinate i.

1. Suppose first that $i \in [N]$ is a coordinate in some copy of the base code, and let w^* be a random copy of the base codeword. By the discussion above with probability at least $1/2$ the event W_{close} holds, i.e., the w^* is τ-close to $c^* \in C_{base}$. Therefore,

$$\Pr[ALG^w_3(i) \in \{c_i, \perp\}] \geq 1/2 \cdot \Pr[ALG^w_2(i) \in \{c^*_{i^*}, \perp\} | W_{\mathrm{close}}] \ .$$

Analysing the term $\Pr[ALG^w_2(j) \in \{c^*_{i^*}, \perp\} | W_{\mathrm{close}}]$, we have

$$\begin{aligned} \Pr[ALG^w_2(i^*) \in \{c^*_{i^*}, \perp\} | W_{\mathrm{close}}] \quad = \quad & \Pr[ALG^w_2(i^*) \in \{c^*_{i^*}, \perp\} | w^*_{i^*} = c^*_{i^*} \wedge W_{\mathrm{close}}] \cdot \Pr[w^*_{i^*} = c^*_{i^*} | W_{\mathrm{close}}] \\ & + \Pr[ALG^w_2(i^*) \in \{c^*_{i^*}, \perp\} | w^*_{i^*} \neq c^*_{i^*} \wedge W_{\mathrm{close}}] \cdot \Pr[w^*_{i^*} \neq c^*_{i^*} | W_{\mathrm{close}}] \ . \end{aligned}$$

For the first term note that Algorithm 2 always returns either $w^*_{i^*}$ or $\perp$. Therefore if $w^*_{i^*} = c^*_{i^*}$, then $\Pr[ALG^w_2(i^*) \in \{c^*_{i^*}, \perp\} | w^*_{i^*} = c^*_{i^*} \wedge W_{\text{close}}] = 1$.

Suppose now that $w^*_{i^*} \neq c^*_{i^*}$, then if we choose the constraints as per Step 2 of Algorithm 2, then with probability at least ϵ_{RW} there will be some $r \in [t]$ such that $\text{dist}_{k_r}(w_{|\text{Dom}_r}, \mathsf{sat}(P_r)) \geq \rho$. If that happens, then the PCPP verifier applied to the predicate P_r in Step 4 will accept with probability at most ϵ_{PCPP}. Therefore, the second term is lower bounded by

$$\Pr[ALG^w_2(i^*) = \perp | w^*_{i^*} \neq c^*_{i^*} \wedge W_{\text{close}}] \geq \epsilon_{RW}(1 - \epsilon_{\text{PCPP}}) \ .$$

Combining the two terms we conclude that

$$\Pr[ALG^w_3(i) \in \{c_i, \perp\}] \geq \frac{\epsilon_{RW} \cdot (1 - \epsilon_{\text{PCPP}})}{2} \geq \epsilon_{RLCC} \ .$$

2. Next, let $i \in [N]$ be a coordinate that belongs to a proof of proximity $\pi_{(i^*,\mathcal{P})}$ for some coordinate i^* corresponding to a symbol from the base code, and a predicate $\mathcal{P} \in \mathcal{C}_{i^*}$. We prove that if $\Pr[ALG^w_4(i) = \perp] < \epsilon_{RLCC}$, then $\Pr[ALG^w_4(i) = c_i] \geq \epsilon_{in}/4$. This clearly suffices as by the choice of ϵ_{RLCC} we have $\epsilon_{in}/4 \geq \epsilon_{RLCC}$.

   Again, by the discussion above, the event W_{close} holds with probability at least $1/2$, i.e., a random copy of the base codeword w^* is $4\tau_{\text{cor}} = \tau$-close to $c^* \in C_{base}$. We will assume that in Step 3 we choose such w^*.

   If Algorithm 4 returns $\perp$ with probability less than ϵ_{RLCC} in Step 6, then the PCPP verifier for $\pi_{(i^*,\mathcal{P})}$ in Step 4 accepts with probability at least $1 - \epsilon_{RLCC} > \epsilon_{\text{PCPP}}$. Therefore, there is some $c_{\mathcal{P}} \in \mathsf{sat}(\mathcal{P})$ (not necessarily equal to $c^*_{|\text{Dom}(\mathcal{P})}$) such that $\text{dist}_{i^*}(w^*_{|\text{Dom}(\mathcal{P})}, c_{\mathcal{P}}) \leq \rho$, and $\pi_{(i^*,\mathcal{P})}$ is ρ-close to $\pi(c_{\mathcal{P}})$. Therefore, since $\rho < 1/2$, it follows that (i) $w^*_{i^*} = (c_{\mathcal{P}})_{i^*}$, (ii) $\text{dist}(w^*_{|\text{Dom}(\mathcal{P})}, c_{\mathcal{P}}) \leq 2\rho$ with respect to the uniform distance, (iii) and $\pi_{(i^*,\mathcal{P})}$ is ρ-close to $\pi(c_{\mathcal{P}})$.

   Suppose that Algorithm 2 in Step 11 returns $\perp$ with probability less than ϵ_{RLCC}. Consider Step 9 in Algorithm 4, where we run Algorithm 2 on the input w^* for a random coordinate j' in $\text{Dom}(\mathcal{P})$. For each $j' \in \text{Dom}(\mathcal{P})$ let $p_{j'}$ be the probability that step 9 accepts on j'. Then $\mathbb{E}[p_{j'}] > 1 - \epsilon_{RLCC}$, and hence for more than $(1 - \delta_{\mathcal{P}}/2)|\text{Dom}(\mathcal{P})|$ coordinates $j' \in \text{Dom}(\mathcal{P})$ we have $p_{j'} \geq 1 - 2\epsilon_{RLCC}/\delta_{\mathcal{P}} \geq 1 - \epsilon_{RW} \cdot (1 - \epsilon_{\text{PCPP}})$.

   Therefore, by the analysis of Algorithm 3 it follows that for more than $(1 - \delta_{\mathcal{P}}/2)|\text{Dom}(\mathcal{P})|$ coordinates $j' \in \text{Dom}(\mathcal{P})$ it holds that $w^*_{j'} = c^*_{j'}$, i.e., $\text{dist}(w^*_{|\text{Dom}(\mathcal{P})}, c^*_{|\text{Dom}(\mathcal{P})}) < \delta_{\mathcal{P}}/2$. Combining with the conclusion from the previous step that $\text{dist}(w^*_{|\text{Dom}(\mathcal{P})}, c_{\mathcal{P}}) \leq 2\rho$ it follows that $\text{dist}(c^*_{|\text{Dom}(\mathcal{P})}, c_{|\text{Dom}(\mathcal{P})}) < 2\rho + \delta_{\mathcal{P}}/2 \leq \delta_{\mathcal{P}}$. Therefore, since $\mathsf{sat}(\mathcal{P})$ has distance $\delta_{\mathcal{P}}$ we conclude that $c^*_{|\text{Dom}(\mathcal{P})} = c_{|\text{Dom}(\mathcal{P})}$.

   So far we showed that if Algorithm 4 returns $\perp$ with probability less than ϵ_{RLCC} and w^* is τ-close to c^* (which happens with probability at least $1/2$), then $w^*_{|\text{Dom}(\mathcal{P})}$ is 2ρ-close to $c^*_{|\text{Dom}(\mathcal{P})}$, and $\pi_{(i^*,\mathcal{P})}$ is ρ-close to $\pi(c^*_{|\text{Dom}(\mathcal{P})})$. Hence, the local correction algorithm for the inner PCPP applied on $(w^*_{|\text{Dom}(\mathcal{P})} \circ \pi_{(i^*,\mathcal{P})})$ in Step 13 will return c^*_i or $\perp$ with probability at least ϵ_{in}. Therefore,

$$\Pr[ALG^w_4(i) \in \{c^*_i, \perp\}] \geq \Pr[\text{Step 14 returns } c^*_i \text{ or } \perp | W_{\text{close}}] \cdot \Pr[W_{\text{close}}] \geq \epsilon_{in}/2 \ .$$

   However, by the assumption Algorithm 4 returns $\perp$ with probability at most $\epsilon_{RLCC} \leq \epsilon_{in}/4$, and thus $\Pr[ALG^w_4(i) = c_i] \geq \epsilon_{in}/2 - \epsilon_{RLCC} \geq \epsilon_{in}/4$. This completes the proof of correctness.

5 CTRW for tensor codes

In this section we prove a general theorem saying that the tensor product of any code with a good distance code admits a CTRW with appropriate parameters. Indeed, this theorem will be used as one of the components required for composition as per Theorem 4.3 described in the previous section.

Theorem 5.1. *Let $C \subseteq \mathbb{F}^n$ be an arbitrary linear code of relative distance δ_C, and let $C^{\otimes m}$ be the m-dimensional tensor product of C. Then, $C^{\otimes m}$ admits an (n, m)-CTRW with (τ, ρ, ϵ)-robust soundness, where the soundness parameter is $\epsilon = (\delta_C - 2\rho)^m - \tau$.*

Furthermore, all predicates defined by the CTRW for $C^{\otimes m}$ check that the restriction of the given tensor word to a line belongs to C.

Proof. We describe the consistency test for $C^{\otimes m}$. Fix a word $w \in \mathbb{F}^{[n]^m}$ and an index $\bar{z} = (z_1, \ldots, z_m) \in [n]^m$. The consistency test works as follows.

Algorithm 5 Consistency test for the m-dimensional tensor product of $C^{\otimes m}$

Input: $w \in \mathbb{F}^{[n]^m}, \bar{z} = (z_1, \ldots, z_m) \in [n]^m$

1: Let $z^{(1)} = \bar{z}$.
2: Sample $\bar{r} = (r_1, \ldots, r_m) \in [n]^m$ uniformly at random.
3: **for** $j = 1$ **to** m **do**
4: Let $z^{(j+1)} = (r_1, \ldots, r_j, z_{j+1}, \ldots, z_m) \in [n]^m$.
5: Let $\ell_j = (r_1, \ldots, r_{j-1}, *, z_{j+1}, \ldots, z_m)$ be the line that passes through $z^{(j)}$ and $z^{(j+1)}$.
6: Let $\mathcal{P}_j$ be the predicate that accepts if and only if $w(\ell_j)$ is a codeword in C.
7: **end for**
8: **return** ACCEPT if and only if $\mathcal{P}_j$ is satisfied for all $j \in [m]$.

By construction, if $w \in C^{\otimes m}$, then the consistency test always accepts. For the soundness analysis, let $w \in \mathbb{F}^{[n]^m}$ be a word that is τ-close to some codeword $c^* \in C$, and let $\bar{z} = (z_1, \ldots, z_m) \in [n]^m$ be such that $w_{\bar{z}} \neq c^*_{\bar{z}}$. Note that we may assume without loss of generality that w is τ-close to the all-zero codeword, i.e., $c^* = 0$. Indeed, suppose that $w_{\bar{i}}$ is τ-close to some codeword $c^* \in C$ that is not all-zeros, and consider the word $w' = w - c^*$. It is easy to verify that w' is τ-close to 0, and the behaviors of the consistency test on w and w' are the same.

Next we show that if w is τ-close to the zero codeword but $w_{\bar{z}} \neq 0$, then

$$\Pr[\exists j \in [m] \text{ such that } \mathrm{dist}_{z^{(j)}}(w_{|\ell_j}, C) \geq \rho] \geq (\delta_C - 2\rho)^m - \tau \ .$$

We introduce notation that will be used throughout the proof. For each point $x \in [n]^m$ and $j \in [m]$ denote by $L_{x,j}$ the line in direction j that passes through x, i.e., $L_{x,j} = \{(x_1, \ldots, x_{j-1}, t, x_{j+1}, \ldots, x_m) : t \in [n]\}$. For each $j \in [m]$ denote by F_j the event that $w_{z^{(j)}} \neq 0$ and w has more than $(\delta_C - 2\rho)n$ non-zeros on the line $L_{z^{(j)},j}$; denote by E_j the event that $w_{z^{(j)}} \neq 0$ and w has at most $(\delta_C - 2\rho)n$ non-zeros on the line $L_{z^{(j)},j}$. Informally, the events F_j "contribute" to the distance of w from the zero codeword, and the events E_j "contribute" to the probability that $\mathrm{dist}_{z^{(j)}}(w_{|\ell_{j+1}}, C) \geq \rho$.

For each $j \in [m]$ denote

$$\epsilon_j = \Pr\left[(\wedge_{i=1}^{j-1} F_i) \bigwedge E_j\right] \ .$$

That is, ϵ_j is the probability that in Algorithm 5 j is the minimal index such that $w_{z^{(1)}}, \ldots w_{z^{(j)}}$ are non-zeros, and the line ℓ_j contains at most $(\delta_C - 2\rho)n$ non-zero points. Note that the events corresponding to ϵ_j's are

disjoint, and if they happen, then $w_{|\ell_j}$ is ρ-far from C with respect to $\mathrm{dist}_{z^{(j)}}$. Therefore,

$$\Pr[\exists j \in [m] \text{ such that } \mathrm{dist}_{z^{(j)}}(w_{|\ell_j}, C) \geq \rho] \geq \sum_{j=2}^{m} \epsilon_j \ .$$

Below we show that the distance of w from the zero codeword is at least $(\delta_C - 2\rho)^m - \sum_{j=2}^{m}(\delta_C - 2\rho)^{m-j}\epsilon_j$.

Note that $z^{(m+1)}$ is distributed uniformly in $[n]^m$, and thus

$$\mathrm{dist}(w, 0) = \Pr[w_{z^{(m+1)}} \neq 0] \geq \Pr[(\wedge_{j=2}^{m} F_j) \wedge w_{z^{(m+1)}} \neq 0] \ .$$

By definition if F_m holds, then $\Pr[w_{z^{(m+1)}} \neq 0 | F_m] \geq (\delta_C - 2\rho)$. Furthermore, also conditioning on $\wedge_{j=2}^{m} F_j$ it holds that $\Pr[w_{z^{(m+1)}} \neq 0 | \wedge_{j=2}^{m} F_j] \geq (\delta_C - 2\rho)$. Hence

$$\begin{aligned} \mathrm{dist}(w, 0) &\geq \Pr[(\wedge_{j=2}^{m} F_j) \wedge w_{z^{(m+1)}} \neq 0] \\ &= \Pr[\wedge_{j=2}^{m} F_j] \times \Pr[w_{z^{(m+1)}} \neq 0 | \wedge_{j=2}^{m} F_j] \\ &\geq \Pr[\wedge_{j=2}^{m} F_j] \times (\delta_C - 2\rho) \ . \end{aligned}$$

Therefore,

$$\mathrm{dist}(w, 0) \geq \Pr[\wedge_{j=2}^{m} F_j] \times (\delta_C - 2\rho) \ . \tag{2}$$

Next, we lower bound $\Pr[\wedge_{j=2}^{m} F_j]$ by "peeling off" one F_j at a time. Specifically, we show that

$$\Pr[\wedge_{j=2}^{m} F_j] \geq \Pr[\wedge_{j=2}^{m-1} F_j] \times (\delta_C - 2\rho) - \epsilon_m \ . \tag{3}$$

Indeed, observe that

$$\begin{aligned} \Pr[(\wedge_{j=2}^{m-1} F_j) \wedge w_{z^{(m)}} \neq 0] &= \Pr[\wedge_{j=2}^{m} F_j] + \Pr[(\wedge_{j=2}^{m-1} F_j) \wedge E_m] \\ &= \Pr[\wedge_{j=2}^{m} F_j] + \epsilon_m \ . \end{aligned}$$

On the other hand

$$\begin{aligned} \Pr[(\wedge_{j=2}^{m-1} F_j) \wedge w_{z^{(m)}} \neq 0] &= \Pr[(\wedge_{j=2}^{m-1} F_j)] \times \Pr[w_{z^{(m)}} \neq 0 | \wedge_{j=2}^{m-1} F_j] \\ &\geq \Pr[(\wedge_{j=2}^{m-1} F_j)] \times (\delta_C - 2\rho) \ , \end{aligned}$$

which proves Eq. (3).

By repeating the same argument, peeling off one F_j at a time, we get

$$\begin{aligned} \Pr[(\wedge_{j=2}^{m} F_j)] &\geq \Pr[(\wedge_{j=2}^{m-1} F_j)] \times (\delta_C - 2\rho) - \epsilon_m \\ &\geq (\Pr[(\wedge_{j=2}^{m-2} F_j)] \times (\delta_C - 2\rho) - \epsilon_{m-1}) \times (\delta_C - 2\rho) - \epsilon_m \\ &= \Pr[(\wedge_{j=2}^{m-2} F_j)] \times (\delta_C - 2\rho)^2 - (\delta_C - 2\rho)\epsilon_{m-1} - \epsilon_m \\ &\geq \cdots \\ &\geq \Pr[F_2] \times (\delta_C - 2\rho)^{m-2} - \sum_{j=3}^{m} (\delta_C - 2\rho)^{m-j}\epsilon_j \\ &\geq (\delta_C - 2\rho - \epsilon_2) \times (\delta_C - 2\rho)^{m-2} - \sum_{j=2}^{m-1} (\delta_C - 2\rho)^{m-j}\epsilon_j \\ &= (\delta_C - 2\rho)^{m-1} - \sum_{j=2}^{m} (\delta_C - 2\rho)^{m-j}\epsilon_j \ . \end{aligned}$$

Plugging this into Eq. (2) we get

$$\mathrm{dist}(w,0) \geq (\delta_C - 2\rho)^m - \sum_{j=2}^{m} (\delta_C - 2\rho)^{m+1-j}\epsilon_j \geq (\delta_C - 2\rho)^m - (\delta_C - 2\rho)\sum_{j=2}^{m}\epsilon_j \ .$$

Therefore, using the fact that $\Pr[\exists j \in [m] \text{ such that } \mathrm{dist}_{z^{(j)}}(w_{|\ell_j}, C) \geq \rho] \geq \sum_{j=1}^{m-1}\epsilon_j$ we conclude that

$$\Pr[\exists j \in [m] \text{ such that } \mathrm{dist}_{z^{(j)}}(w_{|\ell_j}, C) \geq \rho] \geq (\delta_C - 2\rho)^m - \mathrm{dist}(w,0) \geq (\delta_C - 2\rho)^m - \tau \ ,$$

as required. □

The following corollary, which we will use as a component in our composition theorem, follows immediately from Theorem 5.1.

Corollary 5.2. *For integer parameters $d, m \geq 2$ let $n = 2md$. Let $\mathsf{RS} = \mathsf{RS}(n,d)$ be the Reed–Solomon code of message length d and block length n over some field $\mathbb{F}$ of size at least n, so that its distance is $\delta_C = 1 - 1/2m$. Then $\mathsf{RS}^{\otimes m}$ satisfies the following properties.*

1. *The code $\mathsf{RS}^{\otimes m}$ has message length d^m and block length $n^m = (2m)^m \cdot d^m$.*
2. *The relative distance of $\mathsf{RS}^{\otimes m}$ is $(1 - \frac{1}{2m})^m \geq 0.5$.*
3. *$\mathsf{RS}^{\otimes m}$ admits an (n,m)-CTRW with $(\tau = 0.1, \rho = 1/4m, \epsilon_{RW} = 0.15)$-robust soundness.*
4. *For each coordinate $\bar{z} = (z_1, \ldots, z_m) \in [n]^m$ of $\mathsf{RS}^{\otimes m}$ the set of all constraints $\mathcal{C}_{\bar{z}}$ defined by the CTRW consists of m predicates, one for each axis-parallel line passing through $\bar{z}$, checking that the restriction of the tensor word to the line belongs to $\mathsf{RS}(n,d)$. In particular, the corresponding language is solvable in polynomial time and has distance $1 - 1/2m > 1/2$.*

Proof of Corollary 5.2. It suffices to invoke Theorem 5.1 as follows. For $n = 2md$ the code $C = \mathsf{RS}(n,d)$ has relative distance is $\delta_C = 1 - 1/2m$, and the tensor code $\mathsf{RS}(n,d)^{\otimes m}$ has relative distance $(1 - 1/2m)^m \geq 0.5$. Therefore, by Theorem 5.1 the tensor code $\mathsf{RS}(n,d)^{\otimes m}$ admits an (n,m)-CTRW with $(\tau = 0.1, \rho = 1/4m, \epsilon)$-robust soundness, for $\epsilon = (\delta_C - 2\rho)^m - \tau = (1 - 1/m)^m - 0.1 \geq 0.15$. The last item of Corollary 5.2 is immediate by definition of constraints in Algorithm 5. □

6 Relaxed-correctable canonical PCPPs

In this section we describe a new and simple construction of canonical PCPs of proximity that are relaxed locally correctable. Specifically, we show that for every language $L \in \mathcal{P}$ and all $j \in \mathbb{N}$ there exists a *canonical PCP of proximity* verifier such that for any $x \in L$ the length of $\pi(x)$, the canonical proof for x is $|\pi(x)| = \mathrm{poly}(|x|)$, and the language of the canonical proofs $\{x \circ \pi(x) : x \in L\}$ admits a local correction algorithm with correction radius $\tau_{\mathrm{cor}} = \Omega(1)$, soundness $\epsilon = 1/2$ for proximity parameter $\rho > 0$ with respect to distance measure dist_j, and query complexity $O(1/\rho)$.

Theorem 6.1. *Let $L \subseteq \Sigma^*$ be a language in $\mathcal{P}$, let $j \in \mathbb{N}$, and let $\rho > 0$ be a proximity parameter. There exists a canonical PCPP verfier with the following properties.*

1. *For all $n \geq j$ the PCPP verifier on inputs of length n has soundness $\epsilon_{\mathrm{PCPP}} = 0.5$ for proximity parameter ρ with respect to the distance measure dist_j.*

2. *The query complexity of the PCPP verifier is* $q_{\mathrm{PCPP}} = O(1/\rho)$.

3. *For every* $x \in L$ *of length* $|x| = n$ *and every coordinate* $j \in [n]$ *the length of the canonical proof is* $\ell(n) = |x|^{O(1)}$.

4. *The code* $\Pi_L = \{w \circ \pi(w) : w \in L\}$ *is a* (τ, ϵ_{in})*-RLCC with query complexity* $q_{RLCC} = O(1)$ *for correcting radius* $\tau = \Omega(1)$ *(with respect to* $\mathrm{dist}(\cdot)$*) and soundness* $\epsilon_{in} = 1/2$.

The following notation will be used in the proof of Theorem 6.1.

Definition 6.2. *For* $x \in \Sigma^n$ *and* $j \in [n]$ *denote by define* $x^{(j)}$ *the concatenation of* x *with* n *repetitions of* x_j*. That is,* $x^{(j)} = x \circ x_j^n$.

For a language $L \subseteq \Sigma^*$ *and* $j \in \mathbb{N}$ *define* $L^{(j)} = \cup_{n=j}^{\infty} \{x^{(j)} : x \in L \cap \mathbb{F}^n\}$.

It is clear from the definition that if $L \in \mathcal{P}$, then $L^{(j)} \in \mathcal{P}$. The following observation is immediate from the definition.

Observation 6.3. *Fix a language* $L \subseteq \mathbb{F}^*$*. Then, for all* $w \in \mathbb{F}^n$ *and all* $j \in [n]$ *it holds that*

$$\mathrm{dist}_j(w, L) = \mathrm{dist}(w^{(j)}, L^{(j)}) \ .$$

Proof of Theorem 6.1. Let L be a language in $\mathcal{P}$, and let $j \in \mathbb{N}$. In order to construct the required cPCPP verifier we use the following two prior constructions.

- Theorem 3.8: a (non-correctable) strongly canonical PCP of proximity for L with polynomial length and query complexity $O(1/\rho)$ and soundness $1/2$ for proximity parameter $\rho/4$.
- Theorem 3.6: a systematic $(q_{RLCC}, \tau_{\mathrm{cor}})$-RLCC C_{RLCC} of message length k and block length $n = \mathrm{poly}(k)$ with query complexity $q_{RLCC} = O(1)$, correcting radius $\tau_{\mathrm{cor}} = \Omega(1)$, and soundness $\epsilon = 1/2$.

We combine these ingredients via a simple composition to obtain the desired PCPP for L.

Let $V^{(j)}$ be the PCPP verifier for $L^{(j)}$ as per Theorem 3.8. Define a PCPP verifier V_j for L that checks that a given string is close to L with respect to the distance dist_j as follows. Given an oracle access to $x \in \mathbb{F}^n$ the verifier V_j runs $V^{(j)}$ on $x^{(j)}$ by reading one symbol of x for each query that $V^{(j)}$ makes to $x^{(j)}$. By Theorem 3.8 the verifier V_j makes $O(1/\rho)$ queries, and has soundness $1/2$ for proximity parameter $\rho/4$ (with respect to the uniform distance). Therefore, by Observation 6.3 it follows the verifier V_j defines for each $x \in L$ a proof $\pi_{x,j}$ of length $|\pi_{x,j}| = \ell(2n) = \mathrm{poly}(|x|)$. The verifier V_j makes $O(1/\rho)$ queries to x, and has soundness $1/2$ for proximity parameter $\rho/4$ with respect to the distance measure dist_j.

Next, given V_j we define a canonical PCPP verifier V_j^{RLCC} such that the language of canonical proofs admits a local correcting algorithm with desired parameters. Given an input $x \in \Sigma^n$ and a canonical proof $\pi_j(x)$ for V_j claiming that x is (dist_j-close to some word) in L we define a canonical proof $\pi_j^{\mathrm{RLCC}}(x)$ for V_j^{RLCC} by encoding π using a relaxed locally correctable code C_{RLCC} as in Theorem 3.6 as follows. The canonical proof of $x \in L$ for V_j^{RLCC}, denoted by $\pi_j^{\mathrm{RLCC}}(x)$, is given by concatenating t identical copies of $\pi_j(x)$ with $C_{\mathrm{RLCC}}(x \circ \pi_j(x))$, where the number of t is such that $2|C_{\mathrm{RLCC}}(x \circ \pi_j(x))|/\rho < t \cdot |\pi_j(x)|$, and we treat C_{RLCC} as a mapping from $\Sigma^{n+\ell(2n)}$ to $\Sigma^{\mathrm{poly}(\ell(n))}$. The PCPP verifier V_j^{RLCC} on input $x \in \Sigma^n$ and $j \in [n]$ invokes the verifier V_j on a random copy of the systematic part of $\pi_j^{\mathrm{RLCC}}(x)$ obtained by sampling each of the symbols of $\pi_j(x)$ by sampling each symbol uniformly among the t corresponding symbols independently.

It is clear that V_j^{RLCC} inherits the completeness property of V_j. For the soundness suppose that either x is ρ-far from L or x is ρ-close to some $w \in L$ but the proof is ρ-far from $\pi_j^{\mathrm{RLCC}}(x)$. We consider each of the two cases below.

- x is ρ-far from L: In this case no proof will convince the verifier to accept with probability more than $1/2$.
- x is ρ-close to some $w \in L$, but the proof is ρ-far from the canonical proof $\pi_j^{\mathrm{RLCC}}(w)$. In this case, by the choice of t satisfying $2|C_{\mathrm{RLCC}}(x \circ \pi_j(x))|/\rho < t \cdot |\pi_j(x)|$ it follows that the t copies of $\pi_j(x)$ are at least $\rho/2$-far from t repetitions of $\pi_j(w)$. Therefore, by concentration inequalities for sums of bounded independent (but *not* identically distributed) random variables, such as Theorem 3.2, if we choose a random copy of $\pi_j(w)$ by sampling each symbol independently, then with probability at least $p_0 = \Omega(1)$ a random copy is $\rho/4$-far from $\pi_j(w)$. Therefore, the PCPP verifier will reject with probability at least $p_0 \cdot 1/2$. By repeating the verifier $O(1)$ times we obtain a verifier with soundness $\epsilon_{\mathrm{PCPP}} = 1/2$, as required.

It is left to prove that $\{w \circ \pi_j^{\mathrm{RLCC}}(w) : w \in L\}$ admits a local correction algorithm (where recall that j is fixed in Theorem 6.1). Indeed, for any $w \in L$ of length $|w| = n$ the length of $w \circ \pi_j(w)$ is $n + \ell(2n)$. By the assumption, for each $n \in \mathbb{N}$ we have $\{\pi_j^{\mathrm{RLCC}}(w) : w \in L \cap \Sigma^n\} \subseteq \{C_{\mathrm{RLCC}}(y) : y \in \{0,1\}^{n+\ell(2n)}\}$. Therefore, since $\{C_{\mathrm{RLCC}}(y) : y \in \{0,1\}^{n+\ell(2n)}\}$ is an RLCC it follows from Observation 3.5 that the local correction algorithm for C_{RLCC} is also a local correction algorithm for $\{w \circ \pi_j^{\mathrm{RLCC}}(w) : w \in L\}$ with the same soundness guarantees. □

Remark 6.4 (cPCPP with nearly-linear length and query complexity $O(1)$)**.** The proof of Theorem 6.1 shows how to compose a PCPP with a relaxed LCC to obtain a cPCP (note that this composition is in the *reverse direction* of our main composition theorem for deriving relaxed LCC from PCPPs). The length of the composed cPCPP is the length of the original PCPP, with a blowup that is determined by the block length of the relaxed LCC, and its query complexity is proportional to the maximum between the query complexity of the original PCPP and the relaxed LCC. Thus, by applying the argument in Theorem 6.1 to a nearly-linear length, $O(1)$-query PCPP (say, the construction in [BS08; Din07]) together with our main result (the $O(1)$-query relaxed LCC with nearly-linear block length in Theorem 1), we obtain the first construction of cPCPP with nearly-linear length and query complexity $O(1)$. Other constructions in the literature, which (implicitly) achieve relaxed-correctability in our setting, either have super-constant query complexity, or *exponential* length.

7 Proof of main theorem

Below restate our main result (Theorem 1) formally, and prove this theorem using the various components that we constructed.

Theorem 7.1. *Let $k, q \in \mathbb{N}$ be parameters, and let $\mathbb{F}$ be a finite field of size $|\mathbb{F}| \geq k$. Then, there exists a $O(q)$-query relaxed LCC $C \colon \mathbb{F}^K \to \mathbb{F}^N$ with relative distance $\delta(C) \geq 1/4$, decoding radius $\tau_{\mathrm{cor}} = \Omega(1)$, and block length*

$$N = q^{O(\sqrt{q})} \cdot K^{1+O(1/\sqrt{q})} \ .$$

In particular, there exists a RLCC of message length K and block length $N \leq K \cdot 2^{\tilde{O}\left(\sqrt{\log(K)}\right)}$ with constant relative distance that admits a local correction algorithm with query complexity $q = \log(K)/\log\log(K)$ and correction radius $\tau_{\mathrm{cor}} = \Omega(1)$.

Proof. We prove the theorem by invoking the composition theorem (Theorem 4.3) with respect to the components we constructed. Specifically, by Corollary 5.2 the code $\mathsf{RS}^{\otimes m}$ has message length d^m and block length $(2md)^m$. The tensor code $\mathsf{RS}^{\otimes m}$ admits an (n, m)-CTRW with appropriate parameters. Therefore, by applying Theorem 4.3 to the components given in Corollary 5.2 and Theorem 6.1 with $\rho = 1/4m$

we obtain a code $C_{comp} \subseteq \mathbb{F}^N$ of message length $K = d^m$ and block length $N = 2(2md)^m \cdot m \cdot (2dm)^{O(1)} = m^{O(m)} \cdot K^{1+O(1/m)}$. The code C_{comp} is a (τ, ϵ_{RLCC})-RLCC with query complexity $m \cdot q_{\mathrm{PCPP}} = O(m^2)$, where the decoding radius of C_{comp} is $\tau_{\mathrm{cor}} = \tau/4 = \Omega(1)$ and the soundness is $\epsilon_{RLCC} = \min(\frac{\epsilon_{RW} \cdot (1-\epsilon_{\mathrm{PCPP}}) \cdot \delta_C}{2}, \frac{\epsilon_{in}}{2}) = \Omega(1)$. □

References

[Aro+98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. "Proof verification and the hardness of approximation problems". In: *Journal of the ACM (JACM)* 45.3 (1998), pp. 501–555.

[AS98] Sanjeev Arora and Shmuel Safra. "Probabilistic Checking of Proofs: A New Characterization of NP". In: *J. ACM* 45.1 (1998), pp. 70–122.

[Bab+91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. "Checking computations in polylogarithmic time". In: *Proceedings of the 23rd ACM Symposium on Theory of Computing*. STOC '91. 1991, pp. 21–32.

[BDG17] Jop Briët, Zeev Dvir, and Sivakanth Gopi. "Outlaw Distributions and Locally Decodable Codes". In: *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9-11, 2017, Berkeley, CA, USA*. 2017, 20:1–20:19.

[Ben+06] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. "Robust PCPs of Proximity, Shorter PCPs, and Applications to Coding". In: *SIAM Journal on Computing* 36.4 (2006), pp. 889–974.

[BFL91] László Babai, Lance Fortnow, and Carsten Lund. "Non-Deterministic Exponential Time has Two-Prover Interactive Protocols". In: *Computational Complexity* 1 (1991). Preliminary version appeared in FOCS '90., pp. 3–40.

[BGT17] Arnab Bhattacharyya, Sivakanth Gopi, and Avishay Tal. "Lower Bounds for 2-Query LCCs over Large Alphabet". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*. 2017, 30:1–30:20.

[Blo+18] Jeremiah Blocki, Venkata Gandikota, Elena Grigorescu, and Samson Zhou. "Relaxed Locally Correctable Codes in Computationally Bounded Channels". In: *45th International Colloquium on Automata, Languages, and Programming (ICALP 2018)*. 2018.

[BS08] Eli Ben-Sasson and Madhu Sudan. "Short PCPs with Polylog Query Complexity". In: *SIAM J. Comput.* 38.2 (2008), pp. 551–607.

[CG18] Clément L. Canonne and Tom Gur. "An adaptivity hierarchy theorem for property testing". In: *Computational Complexity* 27.4 (2018), pp. 671–716.

[CGW09] Victor Chen, Elena Grigorescu, and Ronald de Wolf. "Efficient and error-correcting data structures for membership and polynomial evaluation". In: *arXiv preprint arXiv:0909.3696* (2009).

[DGG19] Irit Dinur, Oded Goldreich, and Tom Gur. "Every Set in P Is Strongly Testable Under a Suitable Encoding". In: *10th Innovations in Theoretical Computer Science Conference, ITCS*. 2019.

[DH13] Irit Dinur and Prahladh Harsha. "Composition of Low-Error 2-Query PCPs Using Decodable PCPs". In: *SIAM J. Comput.* 42.6 (2013), pp. 2452–2486.

[Din07] Irit Dinur. "The PCP theorem by gap amplification". In: *J. ACM* 54.3 (2007), p. 12.

[Efr12] Klim Efremenko. "3-Query Locally Decodable Codes of Subexponential Length". In: *SIAM J. Comput.* 41.6 (2012), pp. 1694–1703.

[GG16] Oded Goldreich and Tom Gur. "Universal Locally Verifiable Codes and 3-Round Interactive Proofs of Proximity for CSP". In: *Electronic Colloquium on Computational Complexity (ECCC)* 23 (2016), p. 192.

[GG18] Oded Goldreich and Tom Gur. "Universal Locally Testable Codes". In: *Chicago J. Theor. Comput. Sci.* (2018).

[GGK15] Oded Goldreich, Tom Gur, and Ilan Komargodski. "Strong Locally Testable Codes with Relaxed Local Decoders". In: *Proceedings of the 30th Conference on Computational Complexity*. CCC 2015. 2015, pp. 1–41.

[GL20] Tom Gur and Oded Lachish. "A Lower Bound for Relaxed Locally Decodable Codes". In: *31st ACM-SIAM Symposium on Discrete Algorithms (to appear)* (2020).

[GR17] Tom Gur and Ron D. Rothblum. "A Hierarchy Theorem for Interactive Proofs of Proximity". In: *Proceedings of the 8th Innovations in Theoretical Computer Science Conference*. ITCS 2017. 2017.

[GR18] Tom Gur and Ron D. Rothblum. "Non-interactive proofs of proximity". In: *Computational Complexity* 27.1 (2018), pp. 99–207.

[GRR18] Tom Gur, Govind Ramnarayan, and Ron D. Rothblum. "Relaxed Locally Correctable Codes". In: *9th Innovations in Theoretical Computer Science Conference*. ITCS '18. 2018, 27:1–27:11.

[GS06] Oded Goldreich and Madhu Sudan. "Locally testable codes and PCPs of almost-linear length". In: *J. ACM* 53.4 (2006), pp. 558–655.

[Hua+12] Cheng Huang et al. "Erasure coding in windows azure storage". In: *Presented as part of the 2012 USENIX Annual Technical Conference*. 2012, pp. 15–26.

[Jus72] Jørn Justesen. "Class of constructive asymptotically good algebraic codes". In: *IEEE Transactions on Information Theory* (1972).

[KS17] Swastik Kopparty and Shubhangi Saraf. "Local Testing and Decoding of High-Rate Error-Correcting Codes". In: *Electronic Colloquium on Computational Complexity (ECCC)* 24 (2017), p. 126.

[MR10] Dana Moshkovitz and Ran Raz. "Two-query PCP with subconstant error". In: *J. ACM* 57.5 (2010), 29:1–29:29.

[Mul54] David E Muller. "Application of Boolean algebra to switching circuit design and to error detection". In: *Transactions of the IRE professional group on electronic computers* (1954).

[Par20] Orr Paradise. "Smooth and Strong PCPs". In: *Proceedings of the 11th Innovations in Theoretical Computer Science Conference*. ITCS 2020. 2020.

[RR19] Noga Ron-Zewi and Ron Rothblum. "Local Proofs Approaching the Witness Length". In: *Electronic Colloquium on Computational Complexity (ECCC)* 26 (2019), p. 127.

[Tre04] Luca Trevisan. "Some Applications of Coding Theory in Computational Complexity". In: *Electronic Colloquium on Computational Complexity (ECCC)* (2004).

[Yek08] Sergey Yekhanin. "Towards 3-query locally decodable codes of subexponential length". In: *J. ACM* 55.1 (2008), 1:1–1:16.

[Yek12] Sergey Yekhanin. "Locally Decodable Codes". In: *Foundations and Trends in Theoretical Computer Science* 6.3 (2012), pp. 139–255. DOI: 10.1561/0400000030. URL: http://dx.doi.org/10.1561/0400000030.

A Proof of Theorem 1 for binary codes

Below we explain how to prove Theorem 1 for the binary alphabet. The key idea is to use concatenated codes.

Definition A.1. *Let $C_{out}\colon \mathbb{F}^k \to \mathbb{F}^n$ be an error correcting code over the alphabet $\mathbb{F} = \mathsf{GF}(2^{\ell'})$, and let $C_{bin}\colon \{0,1\}^{\ell'} \to \{0,1\}^{\ell}$ be a binary error correcting code. Code concatenation of C_{out} and C_{bin}, denoted by $C_{in} \circ C_{out}$, is the code $C_{concat}\colon \{0,1\}^{k\ell'} \to \{0,1\}^{\ell n}$ defined as follows: A message $x \in \{0,1\}^{k\ell'}$ is treated as a string in $\mathbb{F}^k$. It is first encoded using C_{out}, and then each symbol of the resulting codeword in $\mathbb{F}^n$ is encoded with C_{in}. It will be convenient to treat a codeword $c \in \{0,1\}^{\ell n}$ of C_{concat} as $c \in (\{0,1\}^{\ell})^n$ with the natural interpretation that each block in $\{0,1\}^{\ell}$ is a codeword of C_{in} encoding of one of the symbols in $\mathbb{F}$.*

For a coordinate $i \in [n]$ of the outer code denote by $B[i] \subseteq [\ell n]$ the coordinates corresponding to the i'th block. More generally, for $Q \subseteq [n]$ define $B[Q] = \cup_{i \in Q} B[i]$ to be all coordinates of C_{concat} in all blocks that belong to Q.

Below we define a *block consistency test using random walk (*bCTRW*)* for the concatenated code $C_{in} \circ C_{out}$. This notion extends the notion of CTRW in Definition 4.1. Informally, block consistency test for concatenated codes is defined using a CTRW on the outer code, C_{out}. and the notion of distance in the soundness condition is adapted to the concatenated code.

Before formally defining bCTRW we need one more definition, generalizing the notion of dist_k in Definition 3.1. For a set $Q \subseteq [n]$ define the distance dist_Q between two strings $x, y \in \Sigma^n$ as

$$\mathrm{dist}_Q(x,y) = \frac{\{i \in Q : x_i \neq y_i\}}{2|Q|} + \frac{\{i \in [n] : x_i \neq y_i\}}{2n} \ ,$$

and define dist_Q between a string $x \in \Sigma^n$ and a set $S \subseteq \Sigma^n$ as $\mathrm{dist}_Q(x,S) = \min_{y \in S} \mathrm{dist}_Q(x,y)$.

Definition A.2 (Block consistency test for concatenated codes)**.** *Let $C_{out}\colon \mathbb{F}^k \to \mathbb{F}^n$ be an error correcting code over the alphabet $\mathbb{F} = \mathsf{GF}(2^{\ell'})$, and suppose that C_{out} admits a (q,t)-CTRW. Let $C_{in}\colon \{0,1\}^{\ell'} \to \{0,1\}^{\ell}$ be a binary error correcting code.*

*A $(\ell q, t+1)$-block consistency test using random walk (*bCTRW*) for the concatenated code $C_{in} \circ C_{out}$ is an algorithm that gets as input a string $w \in \{0,1\}^{\ell n}$ and a coordinate $i \in [\ell n]$, and works as follows.*

Algorithm 6 Block consistency test using random walks for the concatenated code $C_{in} \circ C_{out}$

Input: $w \in \{0,1\}^{\ell n}, i \in [\ell n]$

1: Define $\bar{w} \in \mathbb{F}^n$ by letting $\bar{w}_j = \arg\min_{\sigma \in \mathbb{F}} \{\mathrm{dist}(C_{in}(\sigma), w_{|B[j]})\}$. The ties are broken arbitrarily.
2: Treat w as a string in $(\{0,1\}^{\ell})^n$ and let $k_1 \in [n]$ be the index of the block containing the coordinate i.
3: Let $\mathcal{P}_0$ be the predicate checking that $w_{|B[k_1]} \in C_{in}$.
4: Run the CTRW for C_{base} on input $(\bar{w}, k_1)$, and let $\bar{\mathcal{P}}_1, \ldots, \bar{\mathcal{P}}_t$ be the obtained predicates with domains $\mathrm{Dom}_1, \ldots, \mathrm{Dom}_t \subseteq [n]$
5: For each $r \in [t]$ define $\mathcal{P}_r$ be the predicate on $\{0,1\}^{B[\mathrm{Dom}_r]}$ checking that $\bar{\mathcal{P}}_r(\bar{w}_{|\mathrm{Dom}_r}) = 1$ and $w_{B[j]} \in C_{in}$ for all $j \in \mathrm{Dom}_r$.
6: **if** $\mathcal{P}_0(w_{|B[k_1]}) = 1$ and $\mathcal{P}_r(w_{|B[\mathrm{Dom}_r]|}) = 1\ \forall r \in [t]$ **then**
7: **return** ACCEPT
8: **else**
9: **return** REJECT
10: **end if**

That is, the test creates $t+1$ predicates, $\mathcal{P}_0, \mathcal{P}_1, \ldots, P_t$. The predicate $\mathcal{P}_0$ checks that the block containing i contains a codeword of C_{in}, and for $r \geq 1$ the predicates $\mathcal{P}_r$ behave similarly to CTRW *on C_{out}, by checking that each block separately is in C_{in}, and the symbols they encode satisfy the corresponding constraints of C_{out}. In particular, $\mathcal{P}_0$ depends on at most ℓ coordinates of w, and for $r \geq 1$ the predicates $\mathcal{P}_r$ depends on at most ℓq coordinates.*

We say that bCTRW *has perfect completeness and (τ, ρ, ϵ)-robust soundness if it satisfies the following guarantees.*

Perfect completeness: *If $w \in C$, then $\Pr[\mathsf{bCTRW}^w(i) = ACCEPT] = 1$ for al $i \in [\ell n]$.*

(τ, ρ, ϵ)-robust soundness: *If w is τ-close to some $c^* \in C$, but $w_i \neq c^*_i$, then*

$$\Pr[\mathrm{dist}_i(w_{|B[k_1]}, \mathsf{sat}(\mathcal{P}_0)) \geq \rho \text{ or } \exists r \in [t] \text{ such that } \mathrm{dist}_{B[k_r]}(w_{|B[\mathrm{Dom}_r]}, \mathsf{sat}(\mathcal{P}_r)) \geq \rho] \geq \epsilon \ .$$

Below we prove that if C_{out} admits a CTRW and C_{in} is an arbitrary code with good distance, then their composition admits a bCTRW with related parameters.

Theorem A.3. *Let $C_{out} \colon \mathbb{F}^k \to \mathbb{F}^n$ be a linear error correcting code over the alphabet $\mathbb{F} = \mathsf{GF}(2^{\ell'})$, and let $C_{bin} \colon \{0,1\}^{\ell'} \to \{0,1\}^{\ell}$ be a linear binary error correcting code. Suppose that they satisfy the following properties*

1. *C_{out} admits a (q,t)-*CTRW *with perfect completeness and (τ, ρ, ϵ)-robust soundness for $\rho < 1/2$.*
2. *C_{in} has minimal distance δ.*

*Then $C_{concat} = C_{in} \circ C_{out}$ admits a $(\ell q, t+1)$-*bCTRW *with perfect completeness and $(\frac{\delta\tau}{2}, \frac{\delta\rho}{2}, \epsilon)$-robust soundness.*

Proof. It is clear from Algorithm 6 that if $w \in C_{comp}$, then the bCTRW always accepts. For the soundness analysis, let $w \in \{0,1\}^{\ell n}$ be a word that is $\frac{\delta\tau}{2}$-close to some codeword $c^* \in C_{concat}$, and let $i \in [\ell n]$.

Observe that by linearity we may assume that the closest codeword to w is the all-zero codeword, and that $w_i = 1$. Indeed, if c^* is not the all-zeros codeword, then we can consider the word $w' = w - c^*$. It is easy to verify that w' is $\frac{\delta\tau}{2}$-close to the all-zero codeword, and the behaviors of the consistency test on w and w' are the same.

Define $\bar{w} \in \mathbb{F}^n$ by letting $\bar{w}_j = \arg\min_{\sigma \in \mathbb{F}}\{\mathrm{dist}(C_{in}(\sigma), w_{|B[j]})\}$, breaking ties arbitrarily. That is $\bar{w}$ is obtained from w by taking in the j'th block the symbol $\sigma \in \mathbb{F}$ such that $C_{in}(\sigma)$ is the closest to $w_{|B[j]}$.

Let $k_1 \in [n]$ be the index of the block containing the i'th coordinate.

Note first that since $w_i = 1$ and the minimal distance of C_{in} is at least δ, it follows that if $w_{B[k_1]}$ contains at most $(\delta - \delta\rho)\ell$ ones, then $\mathrm{dist}_i(w_{|B[k_1]}, \mathsf{sat}(\mathcal{P}_0)) \geq \frac{\delta\rho}{2}$, which implies the robust soundness condition.

Next, observe that if $\mathrm{dist}(w_{B[k_1]}, C_{in}) \geq \delta\rho$, i.e., $w_{B[k_1]}$ differs from all $c \in C_{in}$ in at least $\delta\rho\ell$ coordinates, then $\mathrm{dist}_i(w_{|B[k_1]}, \mathsf{sat}(\mathcal{P}_0)) \geq \frac{\delta\rho}{2}$, which also implies the robust soundness condition.

Therefore, we will assume from now on that $\mathrm{dist}(w_{B[k_1]}, C_{in}) < \delta\rho$, and $w_{B[k_1]}$ contains more than $(\delta - \delta\rho)\ell$ ones. Since $\rho < 1/2$, this implies that $\bar{w}_{k_1} \neq 0$. Also, observe that if w is $\frac{\delta\tau}{2}$-close to the all-zero codeword in C_{concat}, then $\bar{w}$ is τ-close to the all-zero codeword in C_{out}. Therefore, if we run CTRW for C_{out} on the input $(\bar{w}, k_1)$, then

$$\Pr[\exists r \in [t] \text{ such that } \mathrm{dist}_{k_r}(\bar{w}, \mathsf{sat}(\bar{\mathcal{P}}_r)) \geq \rho] \geq \epsilon \ .$$

Note that if $\mathrm{dist}_{k_r}(\bar{w}, \mathsf{sat}(\bar{\mathcal{P}}_r)) \geq \rho$, then $\mathrm{dist}_{B[k_r]}(w_{B[\mathrm{Dom}_r]}, \mathsf{sat}(\mathcal{P}_r)) \geq \frac{\delta\rho}{2}$. Therefore, if $w_{B[k_1]}$ contains more than $(\delta - \delta\rho)\ell$ ones, then

$$\Pr[\exists r \in [t] \text{ such that } \mathrm{dist}_{B[k_r]}(w_{B[\mathrm{Dom}_r]}, \mathsf{sat}(\mathcal{P}_r)) \geq \frac{\delta\rho}{2}] \geq \epsilon \ ,$$

as required. This completes the proof of Theorem A.3. □

By applying the composition theorem on Theorem A.3 we obtain Theorem 1 for binary alphabet.

Theorem A.4. *Let $K, q \in \mathbb{N}$ be parameters. Then, there exists a $O(q)$-query relaxed LCC $C\colon \{0,1\}^K \to \{0,1\}^N$ with relative distance $\delta(C) = \Omega(1)$, decoding radius $\tau_{\mathrm{cor}} = \Omega(1)$, and block length*

$$N = q^{O(\sqrt{q})} \cdot K^{1+O(1/\sqrt{q})} \ .$$

Proof Sketch. The proof of Theorem A.4 is analogous to the proof of Theorem 1 in Section 7. Specifically, let $\mathsf{RS} = \mathsf{RS}(n,d)$ be the Reed-Solomon code over a field $\mathbb{F}$ of $\mathrm{char}(\mathbb{F}) = 2$, and $\mathbb{F} = \mathsf{GF}(2^{\ell'})$ such that $n \leq 2^{\ell'} \leq 2n$, and let C_{out} be $\mathsf{RS}^{\otimes m}$. By Corollary 5.2 C_{out} admits a (n,m)-CTRW with perfect completeness and $(\tau = 0.1, \rho = 1/4m, \epsilon_{RW} = 0.15)$-robust soundness. Let $C_{in}\colon \{0,1\}^{\ell'} \to \{0,1\}^{\ell}$ be an arbitrary error correcting code of constant rate and constant distance. By applying Theorem A.3 we conclude that their concatenation $C_{in} \circ C_{out}$ admits an $(n \cdot \ell, m+1)$-bCTRW with $(\tau = \Omega(1), \rho = \Omega(m), \epsilon_{RW} = \Omega(1))$-robust soundness. Finally by composing this code with the PCPP from Theorem 6.1 using Theorem 4.3 we conclude Theorem A.4.

The only subtle issue is in the PCPP parts of the code. First, for every coordinate $i \in \ell n$, and the block containing i we need to define a PCPP claiming that the block contains a string that is close to C_{in} with respect to the distance dist_i. In addition for every coordinate k_r of C_{out} and a line containing it we need to construct a PCPP that is sound with respect to the distance $\mathrm{dist}_{B[k_r]}(\cdot, \mathsf{sat}(\mathcal{P}_r))$, where $\mathcal{P}_r$ is the predicated checking that each block on the line is in C_{in}, and the corresponding encoded symbols constitute an evaluation of a low degree polynomial. The construction of such PCPP is similar to that in Theorem 6.1. The only difference is that we need to adapt Definition 6.2 to the notion of $\mathrm{dist}_{B[k_r]}$: given a language $L = \mathsf{sat}(\mathcal{P}_r)$, we define $L^{(B[k_r])}$ where each string is obtained from a string $x \in L$ by concatenating to it $|\mathrm{Dom}_r|$ copies of $x_{|B[k_r]}$, so that these repetitions constitute half of the new string.

We omit the straightforward details. □