

Shannon meets Gray: Noise-robust, Low-sensitivity Codes with Applications in Differential Privacy

David Rasmussen Lolck*

Rasmus Pagh*

Abstract

Integer data is typically made differentially private by adding noise from a Discrete Laplace (or Discrete Gaussian) distribution. We study the setting where differential privacy of a counting query is achieved using bit-wise randomized response, i.e., independent, random bit flips on the encoding of the query answer.

Binary *error-correcting codes* transmitted through noisy channels with independent bit flips are well-studied in information theory. However, such codes are unsuitable for differential privacy since they have (by design) high sensitivity, i.e., neighbouring integers have encodings with a large Hamming distance. *Gray codes* show that it is possible to create an efficient sensitivity 1 encoding, but are also not suitable for differential privacy due to lack of noise-robustness.

Our main result is that it is possible, with a constant rate code, to *simultaneously* achieve the sensitivity of Gray codes *and* the noise-robustness of error-correcting codes (down to the noise level required for differential privacy). An application of this new encoding of the integers is an asymptotically faster, space-optimal differentially private data structure for histograms.

1 Introduction

Random noise in computing can both be a blessing and a curse. In a nutshell, coding theory aims to *amplify* the difference between different data sets such that even after adding random noise it is possible to recreate an original data set with high probability. Conversely, differential privacy [6] deliberately adds noise to *obscure* the difference between different data sets, such that “neighboring” data sets that differ only in the data of one individual become hard to distinguish. In this paper we consider, for given integers m and d , noise-robust binary encodings $\mathcal{C}_{\text{enc}}(v) \in \{0, 1\}^d$ for $v \in \{0, \dots, m-1\}$. Our noise model is a *binary symmetric channel*, meaning that each bit of $\mathcal{C}_{\text{enc}}(v)$ is independently flipped with some probability p upper bounded by a (sufficiently small) absolute constant.

In the differential privacy literature, reporting a noisy random bit is known as *randomized response* [7, 17]. It is known that such noisy encodings satisfy ϵ -differential privacy, where ϵ depends on p and the *sensitivity* of the encoding. Unlike traditional uses of randomized response directly on the input data, we are interested in differential privacy in the context of counting problems where we want to estimate the number of data points that satisfy some predicate. Since two neighbouring datasets will have counts that differ by at most one, if we use an encoding \mathcal{C}_{enc} for the output of the counting problem, the sensitivity is the maximum Hamming distance between the encodings $\mathcal{C}_{\text{enc}}(v)$ and $\mathcal{C}_{\text{enc}}(v+1)$ for $v = 0, \dots, m-2$.

Arguably, the symmetric binary channel is the simplest way in which one could possibly add noise in order to achieve differential privacy. In comparison, adding (say) Laplace noise requires a relatively complex hardware/software system performing nontrivial arithmetic, making it considerably harder to verify and trust. Thus the question we address in this paper is:

Is it possible to achieve good efficiency, privacy and utility guarantees with a deterministic encoding of the integers passed through a binary symmetric channel?

A positive answer to this question requires an “error-correcting Gray code” code with the following properties:

- has short length (ideally close to $\log_2 m$),
- has low sensitivity (ideally 1), and
- is noise robust in the sense that from a noisy version of $\mathcal{C}_{\text{enc}}(v)$ we can compute an estimate whose distribution is tightly concentrated around v .

*Basic Algorithms Research Copenhagen, University of Copenhagen

Encoding	Length	Sensitivity	Noise Robust
Binary	$\log_2(m)$	$\log_2(m)$	No
Gray	$\log_2(m)$	1	No
Unary	m	1	Yes
ECC	$O(\log m)$	$\Omega(\log m)$	Yes
New	$O(\log m)$	1	Yes

Figure 1: Overview of properties of different encodings for integers in $\{0, \dots, m-1\}$. Noise robustness is relative to a noise model in which each bit is flipped with a fixed probability p , a sufficiently small positive constant. We require the encoded integer to be recovered with high probability, up to an additive noise term whose magnitude is bounded by a geometric distribution. All methods have encoding and decoding time that is polynomial in the length of the encoding.

Figure 1 shows properties of four well-known types of integer encodings, all having at most two of these properties. Our main result is that there exists an explicit and efficient code enjoying *all* three properties.

1.1 Background Error-correcting codes. The study of error correction and communication through noisy channels was first introduced by Shannon [13]. He showed that there exists capacity achieving codes, while never explicitly constructing them. Many explicit encoding schemes have since achieved a constant fraction of the capacity, including Reed-Solomon codes [11], Justesen codes [8] (the first such binary code), Expander codes [14, 15] (the first such linear time encodable/decodable code), Polar codes [3] (the first efficient code shown to achieve optimal capacity), and Reed-Muller codes (now known also to achieve optimal capacity [12]).

Low-sensitivity codes. The *reflected binary code* encodes the integers $\{0, \dots, m-1\}$ using $\lceil \log_2 m \rceil$ bits with the property that the encodings of consecutive integers differ only in one bit, i.e., the Hamming distance is 1. Though codes with this property have been known at least since the 19th century, the reflected binary code is commonly referred to as the *Gray code* after Frank Gray who described it in a 1947 patent application [9].

We are not aware of previous work that explicitly addresses combining error-correction capabilities with low sensitivity. *Locality properties* are important in several classes of error-correcting codes including locally decodable codes [19] and locally testable codes [5, 10], but this does not seem to translate into low sensitivity encodings of integers. Another type of (non-binary) codes based on the Chinese Remainder Theorem [16, 18] have nontrivial bounds on sensitivity but do not seem to imply good binary codes.

Integer encodings in differential privacy. Aumüller, Lebeda and Pagh [4] recently presented a differentially private mechanism, ALP, for representing a multiset S , where each data owner provides one of n elements from a ground set $\{1, \dots, u\}$. The output of the mechanism is a data structure that supports *frequency queries*: Given an element x , it returns an estimate of the number of occurrences of x in S . The main difficulty of this problem lies in representing small frequencies, below a threshold $\ell = O(\log u)$, with good precision while keeping space close to the information-theoretical limit. The idea of ALP is to represent frequencies up to ℓ in *unary*, using hashing to determine the location of each bit. To answer a frequency query for x we inspect the bits at positions $h((x, 1)), h((x, 2)), \dots, h((x, \ell))$, where h is a hash function. Without privacy (i.e., $\varepsilon = \infty$), if the frequency is f_x the bits at positions $h((x, 1)), h((x, 2)), \dots, h((x, f_x))$ are guaranteed to be 1. Some bits at positions $h((x, f_x + 1)), \dots, h((x, \ell))$ may be 1 due to hash collisions, but we can determine f_x from the data structure up to a small, geometrically distributed error. To achieve pure differential privacy ALP uses *randomized response* [17], where each bit is flipped with probability depending on ε . This works because the *sensitivity* of a unary code is 1: Adding or removing an element changes at most one bit in the (non-private) data structure. Answering a frequency query is done by taking the *most likely* frequency, in a maximum likelihood sense, namely the one that is closest in Hamming distance to the sequence of observed bits. This yields a tightly concentrated error distribution, comparable to using Laplace distributed noise to release each frequency f_x . However, the use of a unary representation means that the number of bits ℓ needed to retrieve a frequency estimate is $\Theta(\log u)$. We would prefer a more efficient encoding that can still tolerate the errors due to randomized response or hash collisions and has small sensitivity — which is exactly what we achieve in this work.

1.2 Our results Our main result shows how to transform an error-correcting code to a code that has sensitivity 1 and retains good error-correction properties.

THEOREM 1.1. (INFORMAL VERSION OF THEOREM 3.1) *Let \mathcal{C} be a code with block length d and message length $\lg m$. Let $p \in (0; 1/2)$ and let $b_p \sim \text{Bern}(p)^{d'}$. Then there exists an explicit code \mathcal{G} with block length $d' = O(d)$ and message length at least $\lg m$ consisting of the encoder \mathcal{G}_{enc} and decoder \mathcal{G}_{dec} , where \mathcal{G} has sensitivity 1. Furthermore, for every $v \in [m]$ and $t > 0$,*

$$\Pr[|v - \mathcal{G}_{\text{dec}}(\mathcal{G}_{\text{enc}}(v) \oplus b_p)| \geq t] \leq e^{-\Omega(t)} + e^{-\Omega(d)} + O(P_p(\mathcal{C})),$$

where $P_p(\mathcal{C})$ is the probability of \mathcal{C} being decoded incorrectly. If the encoding and decoding of \mathcal{C} runs in polynomial time, then so does the encoding and decoding of \mathcal{G} .

An application of our codes is to replace the unary encodings used in the ALP mechanism of [4]. For concreteness, we consider the case of representing histograms with ε -differential privacy (their Theorem 5.10 with $d = k = n$) where we can show:

THEOREM 1.2. *Given $\epsilon > 0$ and integers n, u , there exists a mechanism for releasing the histogram of a multiset of n elements from $\{1, \dots, u\}$, producing an ε -differentially private data structure with the following properties:*

- *The space usage is $O(n \log u)$ bits,*
- *expected access time to a multiplicity estimate is $O(\log \log u)$,*
- *estimation error is $O(1/\epsilon)$ in expectation and $O(\log(u)/\epsilon)$ with high probability.*

This matches the privacy, space and error properties of [4], which are optimal up to constant factors, while speeding up asymptotic access time exponentially. Competing methods either use space proportional to u , have estimation error that is logarithmic in u , or query time that is proportional to n , see [4] for details. The proof of Theorem 1.2 can be found in Appendix C.

Since our code combines natural properties we believe it may have additional applications. For example, Acharya et al. [1, 2] discuss how the lack of error robustness of the Gray code poses problems for some approaches to private, distributed mean estimation.

1.3 Technical overview We provide two different reductions that transform a classical error-correcting code into an error-correcting Gray code \mathcal{G} . The first reduction works for any error-correcting encoder-decoder pair $\mathcal{C}_{\text{enc}} : [m] \rightarrow \{0, 1\}^d$, $\mathcal{C}_{\text{dec}} : \{0, 1\}^d \rightarrow [m]$ and conceptually works in two steps. First, we construct a code $\mathcal{K} = (\mathcal{K}_{\text{enc}}, \mathcal{K}_{\text{dec}})$ with two key properties:

- Constant consecutive distance, meaning the Hamming distance between the encodings $\mathcal{K}_{\text{enc}}(v)$, $\mathcal{K}_{\text{enc}}(v+1)$ of consecutive integers is a fixed value g , independent of v , and
- decoding is possible even if, in addition to noise, the first or second half is replaced by bits from a different codeword.

We show that such a code with block length $d' = O(d)$ can be constructed by concatenating 4 copies of $\mathcal{C}_{\text{enc}}(v)$, of which two are bit-wise negated if v is odd, plus some padding bits that allow us to determine the parity of v .

The first step gives us codewords for every v divisible by g in which case we let $\mathcal{G}_{\text{enc}}(v) = \mathcal{K}_{\text{enc}}(v/g)$. To obtain intermediate codewords we simply flip the g bits that differ, one by one, such that the Hamming distance between consecutive codewords is 1. In this way, the codeword for v is always a prefix of $\mathcal{K}_{\text{enc}}(\lfloor v/g \rfloor)$ followed by a suffix of $\mathcal{K}_{\text{enc}}(\lfloor v/g \rfloor + 1)$, meaning that we can recover either $\lfloor v/g \rfloor$ or $\lfloor v/g \rfloor + 1$ with high probability. Finally, given $\lfloor v/g \rfloor$ or $\lfloor v/g \rfloor + 1$ we can compute a maximum-likelihood estimate of $v \bmod g$ that is tightly concentrated around the true value.

Our second reduction starts with a linear error-correcting code over $GF(2)$ given by its encoding matrix. As the encoding of a linear code is more predictable than a black-box code, this allows us to avoid working with constant consecutive distance codes, and results in an error-correcting Gray code with better constant factors. Though our constructions can be instantiated with any code, we highlight the combination with expander codes and polar codes, respectively.

2 Definitions

For bitstrings s, t , we define the concatenation of s and t as st . We define the bitwise xor of s and t as $s \oplus t$, and the bitwise inverse of s as \bar{s} . For a natural number n , we write $[n]$ to denote the set $\{0, 1, \dots, n-1\}$. We denote $\text{Bern}(p)$ as the Bernoulli distribution. For a vector x , we let $\|x\|_1$ be the Manhattan norm, that is, the sum of all entries in the vector. In particular for binary vectors, this corresponds to the number of 1 entries in the vector.

We will only be exploring the setting of binary codes, that is, codes that uses a binary alphabet. This is especially meaningful for this work, since our motivation is to use the most simple type of noise to achieve differential privacy in the setting of counting.

DEFINITION 2.1. (CODE) A code $\mathcal{C} = (\mathcal{C}_{\text{enc}}, \mathcal{C}_{\text{dec}})$ is a tuple of an encoding mapping $\mathcal{C}_{\text{enc}} : [m] \rightarrow \{0, 1\}^d$ and a decoding mapping $\mathcal{C}_{\text{dec}} : \{0, 1\}^d \rightarrow [m]$ such that $\mathcal{C}_{\text{dec}}(\mathcal{C}_{\text{enc}}(v)) = v$ for all $v \in [m]$.

In this setting, we will refer to d as the *block length*. Often in literature, we also have the *message length*, which is the number of bits needed to represent the value encoded. We will often be writing this as $\lg m$, signifying that we are able to encode m different values, as we for our usage are more interested in m than the number of bits needed to represent m .

We refer to the value $\mathcal{C}_{\text{enc}}(v)$ as a codeword, signifying it is in the image of the code \mathcal{C} . When looking at error-correcting codes there is often a need for measuring the efficiency. We will focus on the failure probabilities of codes when considering their performance.

DEFINITION 2.2. (FAILURE PROBABILITY) We define the failure probability of a code $\mathcal{C} = (\mathcal{C}_{\text{enc}}, \mathcal{C}_{\text{dec}})$ with block length d and error $b_p \sim \text{Bern}(p)^d$ as the probability,

$$P_p(\mathcal{C}) = \max_{v \in [m]} \Pr[\mathcal{C}_{\text{dec}}(\mathcal{C}_{\text{enc}}(v) \oplus b_p) \neq v].$$

What the failure probability encapsulates is the probability of a code failing under an adversarial choice of v to be encoded. A second measure which can be used to reason about the effectiveness of codes is the distance of a code. This is informally a measure of how close the closest codewords of a code are to each other in terms of Hamming Distance.

DEFINITION 2.3. (HAMMING DISTANCE) The Hamming distance $H : \{0, 1\}^n \times \{0, 1\}^n \rightarrow \mathbb{N}$ between two binary strings is defined as the number of bits where the two strings differ:

$$H(a, b) = |\{i : a_i \neq b_i\}|.$$

DEFINITION 2.4. (DISTANCE) The distance $D(\mathcal{C})$ of a code $\mathcal{C} = (\mathcal{C}_{\text{enc}}, \mathcal{C}_{\text{dec}})$ with block length d is defined as the minimum distance between any two distinct codewords:

$$D(\mathcal{C}) := \min_{a, b \in [m], a \neq b} H(\mathcal{C}_{\text{enc}}(a), \mathcal{C}_{\text{enc}}(b))$$

Finally, we will formally define what we are going to mean by a Gray code.

DEFINITION 2.5. (GRAY CODE) A Gray code is a code $\mathcal{G} = (\mathcal{G}_{\text{enc}}, \mathcal{G}_{\text{dec}})$ with block length d and message length $\lg m$ such that for each $v \in [m-1]$, $H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(v+1)) = 1$.

3 Constructions

In this section, we will look at how to construct an error-correcting Gray code. The construction will be done through reductions from a black-box error-correcting code. The goal is to prove Theorem 1.1.

We will show two similar constructions that achieve the same bound, though with different constants. The first one will be constructed from a general code \mathcal{C} , where we make no assumptions about the structure of the code. This is shown in Section 3.1.

In the second construction, we will show how the assumption that the code \mathcal{C} is linear leads to using fewer repetitions, thereby lowering the constants associated with the code. This is shown in Appendix A.

Before we start with the actual codes, we are going to start by introducing unary codes. As mentioned in the introduction, these codes have most of the properties we are looking for, except they are very space-inefficient.

CONSTRUCTION 3.1. (UNARY CODE) An unary code $\mathcal{U} = (\mathcal{U}_{\text{enc}}, \mathcal{U}_{\text{dec}})$ with block length m and message length $\lg m$ is defined such that

$$\begin{aligned}\mathcal{U}_{\text{enc}}(v) &= 1^v 0^{m-v} \\ \mathcal{U}_{\text{dec}}(c) &= \arg \min_{v \in [m]} H(\mathcal{U}_{\text{enc}}(v), c).\end{aligned}$$

Note that there exist efficient decoding algorithms for the unary code, see e.g. [4].

3.1 Construction from a general code In this section we will show how to construct an error-correcting Gray code from a general code. We will start with introducing what we call complement codes. The idea is that we want to transform a code \mathcal{C} into a code \mathcal{L} where even values are encoded unmodified, while odd values have their bits negated. This will lead to us being able to achieve constant distance between consecutive codewords in Construction 3.3.

CONSTRUCTION 3.2. (COMPLEMENT CODE) Using a code \mathcal{C} with block length d and message length $\lg m$, we construct a code $\mathcal{L} = (\mathcal{L}_{\text{enc}}, \mathcal{L}_{\text{dec}})$ called a complement code with block length $d + D(\mathcal{C})$ and message length m . Define

$$\mathcal{L}_{\text{enc}}(v) = \begin{cases} \mathcal{C}_{\text{enc}}(v) 0^{D(\mathcal{C})} & \text{if } v \text{ is even} \\ \overline{\mathcal{C}_{\text{enc}}(v)} 1^{D(\mathcal{C})} & \text{otherwise} \end{cases}$$

and for $c \in \{0, 1\}^d, t \in \{0, 1\}^{D(\mathcal{C})}$,

$$\mathcal{L}_{\text{dec}}(ct) = \begin{cases} \mathcal{C}_{\text{dec}}(c) & \text{if } t \text{ contains more 0s than 1s} \\ \mathcal{C}_{\text{dec}}(\bar{c}) & \text{if } t \text{ contains more 1s than 0s} \end{cases}$$

As a technical detail, if there is equally many 0s and 1s, then one of the options is chosen uniformly at random. It is clear that \mathcal{L} is a code since for all $v \in [m]$, $\mathcal{L}_{\text{dec}}(\mathcal{L}_{\text{enc}}(v)) = v$ from the fact that \mathcal{C} is a code.

To start with, we are going to bound the error rate of a general code, which we can use to argue that the addition of the parity padding does not significantly worsen the quality of the code. The proof of this lemma can be found in Appendix B.

LEMMA 3.1. Let $\mathcal{C} = (\mathcal{C}_{\text{enc}}, \mathcal{C}_{\text{dec}})$ be a code with block length d and message length $\lg m$, let $p \in (0, 1/2)$ and let $c_p \sim \text{Bern}(p)^{D(\mathcal{C})}$. Then

$$P_p(\mathcal{C}) \geq \Pr \left[\|c_p\|_1 > \frac{D(\mathcal{C})}{2} \right] + \frac{1}{2} \Pr \left[\|c_p\|_1 = \frac{D(\mathcal{C})}{2} \right]$$

What this lemma should be read as is, that any code would perform poorly if put in the same situation as the situations where the parity padding performs poorly. In the setting we are exploring we are mostly concerned with the failure probability so the next lemma lets us directly relate the failure probability of this construction to the failure probability of \mathcal{C} . The lemma mostly argues and uses that any code must have an error probability that is lower bounded by that of decoding the wrong parity.

LEMMA 3.2. Let \mathcal{L} be a code constructed using Construction 3.2 on the code \mathcal{C} . Then $P_p(\mathcal{L}) \leq 2P_p(\mathcal{C})$.

Proof. Let m be the message length and let d be the block length of \mathcal{C} . The code \mathcal{L} can be decoded wrongly in two different ways. Either the parity bit-string is decoded incorrectly, or the inner code \mathcal{C} is decoded incorrectly.

For the first case, observe that this is the setting of Lemma 3.1, and so we can bound the probability of the parity code failing with probability $P_p(\mathcal{C})$. Taking a union bound with the probability of decoding the inner code incorrectly, we get $P_p(\mathcal{L}) \leq 2P_p(\mathcal{C})$. \square

We now show one of our main constructions. While we do insert a large amount of redundancy in the code through repetition, it is this repetition we will later use in the decoding process to rule out collisions of codewords.

CONSTRUCTION 3.3. (CONSTANT CONSECUTIVE DISTANCE CODE) Using the code \mathcal{C} with block length d and message length $\lg m$ and the code \mathcal{L} that is obtained from Construction 3.2 on \mathcal{C} , we construct a code $\mathcal{K} = (\mathcal{K}_{\text{enc}}, \mathcal{K}_{\text{dec}})$ with block length $4d + 2D(\mathcal{C})$ and message length $\lg m$. We define

$$\mathcal{K}_{\text{enc}}(v) := \mathcal{C}_{\text{enc}}(v)\mathcal{L}_{\text{enc}}(v)\mathcal{C}_{\text{enc}}(v)\mathcal{L}_{\text{enc}}(v).$$

Decoding is defined as follows: Let $c_1, c_2 \in \{0, 1\}^d, l_1, l_2 \in \{0, 1\}^{d+D(\mathcal{C})}$. Define the output of $\mathcal{K}_{\text{dec}}(c_1 l_1 c_2 l_2)$ as computing the four values $\mathcal{C}_{\text{dec}}(c_1), \mathcal{C}_{\text{dec}}(c_2), \mathcal{L}_{\text{dec}}(l_1)$ and $\mathcal{L}_{\text{dec}}(l_2)$ and outputting the most frequent one, breaking ties arbitrarily.

Since $\mathcal{C}_{\text{dec}}(\mathcal{C}_{\text{enc}}(v)) = \mathcal{L}_{\text{dec}}(\mathcal{L}_{\text{enc}}(v)) = v$, decoding will also result in v , and so it holds that $\mathcal{K}_{\text{dec}}(\mathcal{K}_{\text{enc}}(v)) = v$ for all v , implying that \mathcal{K} is a code.

The following lemma gives one of the primary reasons to use this construction, namely that successive codewords differs by a fixed number of bits independent of v . This will make decoding feasible later. It should be mentioned that the number of bits still depends on $D(\mathcal{C})$.

LEMMA 3.3. Let $\mathcal{K} = (\mathcal{K}_{\text{enc}}, \mathcal{K}_{\text{dec}})$ be a code obtained from Construction 3.3 on the code $\mathcal{C} = (\mathcal{C}_{\text{enc}}, \mathcal{C}_{\text{dec}})$ with block length d and message length $\lg m$. Then for any $v \in [m - 1]$,

$$H(\mathcal{K}_{\text{enc}}(v), \mathcal{K}_{\text{enc}}(v + 1)) = 2(d + D(\mathcal{C}))$$

Proof. From the construction of \mathcal{K} , we have

$$(3.1) \quad H(\mathcal{K}_{\text{enc}}(v), \mathcal{K}_{\text{enc}}(v + 1)) = 2H(\mathcal{C}_{\text{enc}}(v), \mathcal{C}_{\text{enc}}(v + 1)) + 2H(\mathcal{L}_{\text{enc}}(v), \mathcal{L}_{\text{enc}}(v + 1)),$$

where $\mathcal{C} = (\mathcal{C}_{\text{enc}}, \mathcal{C}_{\text{dec}})$ and $\mathcal{L} = (\mathcal{L}_{\text{enc}}, \mathcal{L}_{\text{dec}})$ is a code obtained from Construction 3.2. Since v and $v + 1$ have different parity, we have that

$$H(\mathcal{L}_{\text{enc}}(v), \mathcal{L}_{\text{enc}}(v + 1)) = H(\mathcal{C}_{\text{enc}}(v), \overline{\mathcal{C}_{\text{enc}}(v + 1)}) + D(\mathcal{C}).$$

Observe that $H(\mathcal{C}_{\text{enc}}(v), \overline{\mathcal{C}_{\text{enc}}(v + 1)}) = d - H(\mathcal{C}_{\text{enc}}(v), \mathcal{C}_{\text{enc}}(v + 1))$, since $\mathcal{C}_{\text{enc}}(v)$ differs from $\mathcal{C}_{\text{enc}}(v + 1)$ in exactly the bit positions where $\mathcal{C}_{\text{enc}}(v)$ is equal to $\overline{\mathcal{C}_{\text{enc}}(v + 1)}$. This gives

$$H(\mathcal{L}_{\text{enc}}(v), \mathcal{L}_{\text{enc}}(v + 1)) = d - H(\mathcal{C}_{\text{enc}}(v), \mathcal{C}_{\text{enc}}(v + 1)) + D(\mathcal{C}).$$

Substituting this into (3.1) completes the proof. \square

The second important property of Construction 3.3 is that because it is composed of 4 codes, we can allow any one of the codes to be modified to a degree where we are unable to decode it correctly as we can discover the encoded value from the other three codes. As we might be unable to directly decide which codeword is the one we are unable to decode, the final decoding is decided by a majority vote.

For the next construction, we define the functions $\text{pref}_i(s)$ on the binary string s to be the prefix of s containing i characters, and similarly we define $\text{suf}_i(s)$ to be the suffix of s containing i characters.

CONSTRUCTION 3.4. (ERROR CORRECTING GRAY CODE) Let \mathcal{C} be a code with block length d and message length $\lg m$ and let $\mathcal{K} = (\mathcal{K}_{\text{enc}}, \mathcal{K}_{\text{dec}})$ be a code obtained from Construction 3.3 on \mathcal{C} . Let $g = 2(d + D(\mathcal{C}))$. We define the Error Correcting Gray Code $\mathcal{G} = (\mathcal{G}_{\text{enc}}, \mathcal{G}_{\text{dec}})$ with block length $4d + 2D(\mathcal{C})$ and message length $\lg mg$.

Let $b_1^{(v)}, \dots, b_g^{(v)}$ be the indices of the bits where $\mathcal{K}_{\text{enc}}(v)$ and $\mathcal{K}_{\text{enc}}(v + 1)$ are different, in sorted order and define $b_0^{(v)} = 0$. Observe that by Lemma 3.3, $\mathcal{K}_{\text{enc}}(v)$ and $\mathcal{K}_{\text{enc}}(v + 1)$ are different in exactly g bits.

Let $v = qg + r$ where $0 \leq r < g$. Then

$$(3.2) \quad \mathcal{G}_{\text{enc}}(v) = \text{pref}_{b_r^{(q)}}(\mathcal{K}_{\text{enc}}(q + 1)) \text{suf}_{g - b_r^{(q)}}(\mathcal{K}_{\text{enc}}(q)).$$

Define the decoding $\mathcal{G}_{\text{dec}}(c)$ as follows: Let $h_v \in \{0, 1\}^g$ such that

$$(h_v)_i = \begin{cases} 0 & \text{if } c_{b_i^{(v)}} = \mathcal{K}_{\text{enc}}(v)_{b_i^{(v)}} \\ 1 & \text{if } c_{b_i^{(v)}} = \mathcal{K}_{\text{enc}}(v + 1)_{b_i^{(v)}} \end{cases}.$$

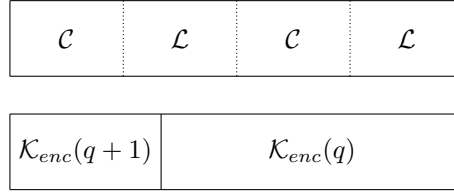


Figure 2: Sketch of the two different perspectives the code can be viewed. The upper one shows the four different component codes, while the lower one shows how it consists of a prefix and a suffix

From this construction, h_v is an unary code. Let \mathcal{U}_{dec} be a decoder for unary codes and let $t = \mathcal{K}_{\text{dec}}(c)$. We then end up with two candidate decodings, which we name

$$v_0 = g(t-1) + \mathcal{U}_{\text{dec}}(h_{t-1}) \text{ and } v_1 = gt + \mathcal{U}_{\text{dec}}(h_t)$$

Then we define

$$(3.3) \quad \mathcal{G}_{\text{dec}}(c) = \arg \min_{v \in \{v_0, v_1\}} H(c, \mathcal{G}_{\text{enc}}(v))$$

We refer to Lemma 3.6 to show that this indeed is a Gray code.

REMARK 3.1. Observe that when constructing \mathcal{G} from \mathcal{C} using Construction 3.4, the time complexity of the encoding and decoding of \mathcal{G} is the same as that of \mathcal{C} , up to constant factors.

From the previous construction, we observe that we end up needing 4 copies of the original code \mathcal{C} . We have to use at least 3 to be able to uniquely decode the outer code, while we needed an even number to be able to easily decode the code from having constant distance.

In the next three lemmas, we are going to show that our construction indeed is a Gray code. Recall that a code has the property of $\mathcal{G}_{\text{dec}}(\mathcal{G}_{\text{enc}}(v)) = v$ for all v and that a Gray code furthermore has sensitivity of 1. We show that \mathcal{G} has a sensitivity of 1 in Lemma 3.4. In Lemma 3.5 we show that $\mathcal{G}_{\text{enc}}(v)$ is injective. These two lemmas are combined in Lemma 3.6 to show that \mathcal{G} indeed is a Gray code.

LEMMA 3.4. Let \mathcal{C} be a code with message length $\lg m$ and let $\mathcal{G} = (\mathcal{G}_{\text{enc}}, \mathcal{G}_{\text{dec}})$ be obtained from Construction 3.4 on \mathcal{C} . Then for any $v \in [m-1]$,

$$H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(v+1)) = 1.$$

Proof. Let \mathcal{C} have block length d and let $g = 2(d + D(\mathcal{C}))$. Let $v = qg + r$ where $0 \leq r < g$. From Construction 3.4, observe that if $r < g-1$, then $\mathcal{G}_{\text{enc}}(v)$ and $\mathcal{G}_{\text{enc}}(v+1)$ are different only in bit $b_{r+1}^{(q)}$. Otherwise, if $r = g-1$ then

$$\begin{aligned} \mathcal{G}_{\text{enc}}(v+1) &= \text{pref}_{b_0^{(q+1)}}(\mathcal{K}_{\text{enc}}(q+2)) \text{suf}_{g-b_0^{(q+1)}}(\mathcal{K}_{\text{enc}}(q+1)) \\ &= \text{pref}_0(\mathcal{K}_{\text{enc}}(q+2)) \text{suf}_g(\mathcal{K}_{\text{enc}}(q+1)) \\ &= \mathcal{K}_{\text{enc}}(q+1) \\ &= \text{pref}_{b_g^{(q)}}(\mathcal{K}_{\text{enc}}(q+1)) \text{suf}_{g-b_g^{(q)}}(\mathcal{K}_{\text{enc}}(q)) \end{aligned}$$

Since $\mathcal{G}_{\text{enc}}(v) = \text{pref}_{b_{g-1}^{(q)}}(\mathcal{K}_{\text{enc}}(q+1)) \text{suf}_{g-b_{g-1}^{(q)}}(\mathcal{K}_{\text{enc}}(q))$, this means that $\mathcal{G}_{\text{enc}}(v)$ and $\mathcal{G}_{\text{enc}}(v+1)$ only differs in the bit $b_g^{(q)}$, proving the statement. \square

LEMMA 3.5. Let \mathcal{C} be a code and let $\mathcal{G} = (\mathcal{G}_{\text{enc}}, \mathcal{G}_{\text{dec}})$ be obtained using Construction 3.4 on \mathcal{C} . Then \mathcal{G}_{enc} is injective.

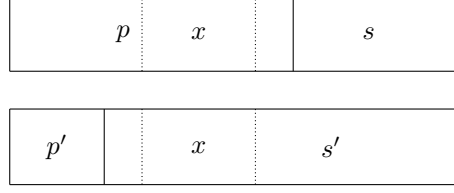


Figure 3: Sketch of the how \mathcal{G} would have to look if the code was not injective.

Proof. Let \mathcal{G} have block length d and message length $\lg m$. Let $v, v' \in [m]$ such that $\mathcal{G}_{\text{enc}}(v) = \mathcal{G}_{\text{enc}}(v') = w$. Let $g = 2(d + D(\mathcal{C}))$. Assume without loss of generality that $v \leq v'$. From the definition of \mathcal{G} , we can split w into 4 codewords $w = c_1 l_1 c_2 l_2$, such that $c_1, c_2 \in \{0, 1\}^d$ and $l_1, l_2 \in \{0, 1\}^{d+D(\mathcal{C})}$. From (3.2), observe that $w = ps = p's'$ can be seen as composed of a prefix p of $\mathcal{K}_{\text{enc}}(\lfloor v/g \rfloor + 1)$ and a suffix s of $\mathcal{K}_{\text{enc}}(\lfloor v/g \rfloor)$. It can also be seen as a prefix p' of $\mathcal{K}_{\text{enc}}(\lfloor v'/g \rfloor + 1)$ and a suffix s' of $\mathcal{K}_{\text{enc}}(\lfloor v'/g \rfloor)$. This means that some codewords of c_1, l_1, c_2, l_2 are to the left of the split, at most one of them is split by the prefix and the suffix, and some of them are to the right of the split. There are therefore at least one codeword x among c_1, l_1, c_2, l_2 , for which x is not split by neither p and s nor p' and s' . Let $\mathcal{X}_{\text{enc}} \in \{\mathcal{C}_{\text{enc}}, \mathcal{L}_{\text{enc}}\}$ be the encoder used to encode x . It then holds that $x \in \{\mathcal{X}_{\text{enc}}(\lfloor v/g \rfloor), \mathcal{X}_{\text{enc}}(\lfloor v/g \rfloor + 1)\}$ and $x \in \{\mathcal{X}_{\text{enc}}(\lfloor v'/g \rfloor), \mathcal{X}_{\text{enc}}(\lfloor v'/g \rfloor + 1)\}$. Since both \mathcal{C} and \mathcal{L} are codes, and \mathcal{C}_{enc} and \mathcal{L}_{enc} therefore injective, we can conclude that $|\lfloor v/g \rfloor - \lfloor v'/g \rfloor| \leq 1$.

Now, assume for the purpose of contradiction that $\lfloor v/g \rfloor + 1 = \lfloor v'/g \rfloor$. This means the codeword they share which is not split, x , must be fully contained in p and in s' , implying that s is fully contained in s' . See Fig. 3 for a sketch of this. We now consider any codeword y fully contained in s . As s is contained in s' so is y . Since y is fully contained in s' then y must be the result of encoding $\lfloor v'/g \rfloor$. This would however imply that y is not in s by injectivity of \mathcal{C}_{enc} . Therefore such a y cannot exist. Furthermore, no codeword z can be split by p and s . To see this, observe that z would also be fully contained in s' , a contradiction since this implies z is fully contained in p . We can therefore conclude that s is empty. This is a contradiction since by construction, s is non-empty. We conclude that $\lfloor v/g \rfloor = \lfloor v'/g \rfloor$.

Next, for any $q \in [m/g - 1]$, $H(\mathcal{G}_{\text{enc}}(gq), \mathcal{G}_{\text{enc}}(g(q+1))) = g$ by Lemma 3.3, and at the same time for any $u \in [m-1]$, $H(\mathcal{G}_{\text{enc}}(u), \mathcal{G}_{\text{enc}}(v+1)) = 1$. This means that for every $r \in [g]$, $H(\mathcal{G}_{\text{enc}}(gq), \mathcal{G}_{\text{enc}}(gq+r)) = r$ and so all encodings must be different. We conclude that $v = v'$, showing that \mathcal{G}_{enc} is injective. \square

LEMMA 3.6. *Let \mathcal{C} be a code and let \mathcal{G} be obtained using Construction 3.4 on \mathcal{C} . Then \mathcal{G} is a Gray code.*

Proof. Let \mathcal{G} have block length d and message length $\lg m$ and let $g = 2(d + D(\mathcal{C}))$. To show that \mathcal{G} indeed is a Gray code, we need to show that Construction 3.4 is a code, or in other words that we decode it correctly. Let $v \in [m]$ and let $\mathcal{G}_{\text{enc}}(v) = c_1 l_1 c_2 l_2$ such that $c_1, c_2 \in \{0, 1\}^d$ and $l_1, l_2 \in \{0, 1\}^{d+D(\mathcal{C})}$. Let $g = 2(d + D(\mathcal{C}))$. By construction, we have $s \notin \{\lfloor v/g \rfloor, \lfloor v/g \rfloor + 1\}$ for at most 1 codeword $s \in \{\mathcal{C}_{\text{dec}}(c_1), \mathcal{L}_{\text{dec}}(l_1), \mathcal{C}_{\text{dec}}(c_2), \mathcal{L}_{\text{dec}}(l_2)\}$. By the pigeonhole principle, at least one of these options must therefore occur twice. This means that in the decoding, the most frequent element t of the multiset $\{\mathcal{C}_{\text{dec}}(c_1), \mathcal{L}_{\text{dec}}(l_1), \mathcal{C}_{\text{dec}}(c_2), \mathcal{L}_{\text{dec}}(l_2)\}$ has the property that $t \in \{\lfloor v/g \rfloor + 1, \lfloor v/g \rfloor\}$. From the construction of the decoder, both these cases are considered.

Next, we observe that using the prefix of $\mathcal{K}_{\text{enc}}(\lfloor v/g \rfloor + 1)$ and the suffix of $\mathcal{K}_{\text{enc}}(\lfloor v/g \rfloor)$ exactly corresponds to a unary encoding on the changing bits in (3.2), implying that decoding $h_{\lfloor v/g \rfloor}$ with \mathcal{U}_{dec} determines the number of bits belonging to the prefix and the number belonging to the suffix, where h_t is defined as in Construction 3.4. This means that $v \in \{v_0, v_1\}$ for $v_0 = g(t-1) + \mathcal{U}_{\text{dec}}(h_{t-1})$ and $v_1 = gt + \mathcal{U}_{\text{dec}}(h_t)$. Since \mathcal{G}_{enc} is injective by Lemma 3.5, $\mathcal{G}_{\text{enc}}(v) \in \{\mathcal{G}_{\text{enc}}(v_0), \mathcal{G}_{\text{enc}}(v_1)\}$ and $\mathcal{G}_{\text{enc}}(v_0) \neq \mathcal{G}_{\text{enc}}(v_1)$. This means exactly one of $H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(v_0))$ and $H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(v_1))$ is equal to zero. As the decoder minimises the Hamming distance, this implies that $\mathcal{G}_{\text{dec}}(\mathcal{G}_{\text{enc}}(v)) = v$, showing that \mathcal{G} is a code. It then follows from Lemma 3.4 that \mathcal{G} is a Gray code. \square

As we now have established that Construction 3.4 is indeed a code and that it has a sensitivity of 1 we can now start looking at the error handling properties of the code. Since our code essentially is an unary code built on top of a black box error correction code we will start by looking at the probability that adding noise results in another decoding being obtained. We are able to bound this based on the Hamming distance between the two bitstrings.

LEMMA 3.7. Let $c_1, c_2 \in \{0, 1\}^d$ be bitstrings, and let $p \in [0, 1/2)$ and let $b_p \sim \text{Bern}(p)^d$. Then

$$\Pr[H(c_1 \oplus b_p, c_2) \leq H(c_1 \oplus b_p, c_1)] \leq \exp\left(-\frac{(1-2p)^2}{4p+2}H(c_1, c_2)\right).$$

Proof. Let $k = H(c_1, c_2)$. Note that if $H(c_1 \oplus b_p, c_1) \geq H(c_1 \oplus b_p, c_2)$, then at least $k/2$ of the k bits where c_1 and c_2 are different must have been flipped. Letting Y be the random variable denoting the number of the k bits that have been flipped, we get

$$\Pr[H(c_1 \oplus b_p, c_2) \leq H(c_1 \oplus b_p, c_1)] \leq \Pr[Y \geq k/2].$$

Due to independence, we can use a Chernoff bound. With $E[Y] = pk$ we get:

$$\Pr[H(c_1 \oplus b_p, c_2) \leq H(c_1 \oplus b_p, c_1)] \leq \Pr[Y \geq k/2] = \Pr\left[Y \geq \frac{1}{2p}pk\right] \leq \exp\left(-\frac{(1-2p)^2}{4p+2}k\right),$$

completing the proof. \square

The relation between any two values $v, v' \in [m]$ encoded with Construction 3.4 depends in large part on $|v - v'|$. Recall that if $|v - v'|$ is large, then the component codes are going to be different. However if $|v - v'|$ is small then the decoding will be more like a unary code decoding. We show this formally with the next two lemmas.

LEMMA 3.8. Let $\mathcal{C} = (\mathcal{C}_{\text{dec}}, \mathcal{C}_{\text{enc}})$ be a code with block length d and message length $\lg m$, and let $\mathcal{G} = (\mathcal{G}_{\text{dec}}, \mathcal{G}_{\text{enc}})$ be obtained using Construction 3.4 on \mathcal{C} . Then for all $v, v' \in [m]$, if $|v - v'| < D(\mathcal{C})$ then $H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(v')) = |v - v'|$.

Proof. Let $g = 2(d + D(\mathcal{C}))$ and fix v . Assume without loss of generality that $v < v'$. Let $b_i^{(v)}$ be defined as in Construction 3.4. To show the statement it suffices to show that all indices $b_0^{(v)}, \dots, b_g^{(v)}$ are unique, and that all indices $b_{g-D(\mathcal{C})+1}^{(v)}, \dots, b_g^{(v)}, b_0^{(v+1)}, \dots, b_{D(\mathcal{C})}^{(v+1)}$ are unique, since one of these is a superset of the bit indices that changes one by one when transforming $\mathcal{G}_{\text{enc}}(v)$ to $\mathcal{G}_{\text{enc}}(w)$ through the series $\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(v+1), \dots, \mathcal{G}_{\text{enc}}(w)$ by the assumption $|v - v'| < D(\mathcal{C})$.

Observe that, $b_0^{(v)}, \dots, b_g^{(v)}$ are all unique by definition. Furthermore, by Lemma 3.3 the first $g/2$ bit changes are found in the first two component codes of \mathcal{G} , while the last $g/2$ are found in the last two component codes. In other words, $b_{g-D(\mathcal{C})+1}^{(v)}, \dots, b_g^{(v)} > 2d + D(\mathcal{C})$, while $b_0^{(v+1)}, \dots, b_{D(\mathcal{C})-1}^{(v+1)} \leq 2d + D(\mathcal{C})$. This means that there can be no duplicates between $b_{g-D(\mathcal{C})+1}^{(v)}, \dots, b_g^{(v)}$ and $b_0^{(v+1)}, \dots, b_{D(\mathcal{C})-1}^{(v+1)}$. \square

LEMMA 3.9. Let $\mathcal{C} = (\mathcal{C}_{\text{dec}}, \mathcal{C}_{\text{enc}})$ be a code with block length d and message length $\lg m$, and let $\mathcal{G} = (\mathcal{G}_{\text{dec}}, \mathcal{G}_{\text{enc}})$ be obtained using Construction 3.4 on \mathcal{C} . Then for all $v, v' \in [m]$, if $|v - v'| \geq D(\mathcal{C})$ then $H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(v')) \geq D(\mathcal{C})$.

Proof. Assume for the purpose of contradiction that there exist v and v' such that $|v - v'| \geq D(\mathcal{C})$ but $H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(v')) < D(\mathcal{C})$. Let $\mathcal{G}_{\text{enc}}(v) = c_1 l_1 c_2 l_2$ and let $\mathcal{G}_{\text{enc}}(v') = c'_1 l'_1 c'_2 l'_2$ where $c_1, c_2, c'_1, c'_2 \in \{0, 1\}^d$ and $l_1, l_2, l'_1, l'_2 \in \{0, 1\}^{d+D(\mathcal{C})}$. Then $H(c_i, c'_i) < D(\mathcal{C})$ and $H(l_i, l'_i) < D(\mathcal{C})$ for $i \in \{1, 2\}$.

Observe that by construction at most one of c_1, l_1, c_2, l_2 is not a codeword of their respective codes, and at most one of c'_1, l'_1, c'_2, l'_2 is not a codeword of their respective codes. This means that there exist a pair: $(x, y) \in \{(c_1, c'_1), (l_1, l'_1), (c_2, c'_2), (l_2, l'_2)\}$ such that both x and y are codewords of the same code. By the definition of $D(\mathcal{C})$, this implies that $x = y$, and so that $\lfloor v/g \rfloor - \lfloor v'/g \rfloor \leq 1$ by injectivity of \mathcal{C}_{enc} .

Next, let f_i be the index of the bit where $\mathcal{G}_{\text{enc}}(v + i)$ and $\mathcal{G}_{\text{enc}}(v + i + 1)$ differ. Observe that since $x = y$, no value in $f_0, \dots, f_{|v-v'|}$ can lie in the bit interval of $\mathcal{G}_{\text{enc}}(v)$ and $\mathcal{G}_{\text{enc}}(v')$ where x and y are placed respectively. This however directly implies that all $f_0, \dots, f_{|v-v'|}$ are unique, since for any duplicates to exist, there would have to be at least one bit-flip in the interval covered by x and y in $\mathcal{G}_{\text{enc}}(v)$ and $\mathcal{G}_{\text{enc}}(v')$. This implies that $H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(v')) = |v - v'|$, a contradiction. \square

Finally, we will look at how well our the result of encoding and decoding is concentrated around the encoded value when adding noise after encoding.

THEOREM 3.1. *Let $\mathcal{C} = (\mathcal{C}_{\text{dec}}, \mathcal{C}_{\text{enc}})$ be a code with block length d and message length $\lg m$, and let $\mathcal{G} = (\mathcal{G}_{\text{dec}}, \mathcal{G}_{\text{enc}})$ be obtained using Construction 3.4 on \mathcal{C} . Let $p \in (0, 1)$ and let $b_p \sim \text{Bern}(p)^{4d+2D(\mathcal{C})}$. Let $c = (1 - 2p)^2 / (4p + 2)$. Then for all $t \geq 0$,*

$$\Pr[|v - \mathcal{G}_{\text{dec}}(\mathcal{G}_{\text{enc}}(v) \oplus b_p)| \geq t] \leq \frac{2}{1 - \exp(-c)} \exp(-ct) + 12d \exp(-cD(\mathcal{C})) + 5P_p(\mathcal{C})$$

Proof. Let $v' = \mathcal{G}_{\text{dec}}(\mathcal{G}_{\text{enc}}(v) \oplus b_p)$. Observe that v' is a random variable. To show the statement we are going to split all possible decoding events for v' into two sets, S_1 and S_2 . S_1 contains all events such that $t \leq |v - v'| < D(\mathcal{C})$. S_2 contains all events such that $|v - v'| \geq D(\mathcal{C})$. Observe that for any v' such that $|v' - v| \geq t$, $v' \in S_1 \cup S_2$. Finally, let F be the event that at least one of the 3 codewords of $\mathcal{G}_{\text{enc}}(v) = c_1 l_1 c_2 l_2$ that are not a concatenation of two different codes, is decoded incorrectly.

We can now rewrite the probability:

$$\begin{aligned} \Pr[|v - v'| \geq t] &= \Pr[(|v - v'| \geq t) \cap F^c] + \Pr[|v - v'| \geq t | F] \Pr[F] \\ &\leq \Pr[(S_1 \cup S_2) \cap F^c] + \Pr[F] \\ (3.4) \quad &\leq \Pr[S_1 \cap F^c] + \Pr[S_2 \cap F^c] + \Pr[F]. \end{aligned}$$

We will bound each of these terms.

We start by bounding $\Pr[S_1 \cap F^c]$. By the definition of F^c , v must have been considered in the second phase of the decoding. This means that v' was chosen over v implying that

$$H(\mathcal{G}_{\text{enc}}(v) \oplus b_p, \mathcal{G}_{\text{enc}}(v')) \leq H(\mathcal{G}_{\text{enc}}(v) \oplus b, \mathcal{G}_{\text{enc}}(v))$$

From this, we can bound the probability $\Pr[S_1 \cap F^c]$ by the sum of probabilities of each possible value of v' that is decodable in the event $S_1 \cap F^c$ being chosen over v . That is for all $w \in [m]$ such that $t \leq |v - w| < D(\mathcal{C})$,

$$\begin{aligned} \Pr[(v' = w) \cap (S_1 \cap F^c)] &\leq \Pr[H(\mathcal{G}_{\text{enc}}(v) \oplus b_p, \mathcal{G}_{\text{enc}}(w)) \leq H(\mathcal{G}_{\text{enc}}(v) \oplus b_p, \mathcal{G}_{\text{enc}}(v))] \\ &\leq \exp(-cH(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(w))) \\ (3.5) \quad &= \exp(-c|v - w|) \end{aligned}$$

using Lemmas 3.7 and 3.8. Observe that for each value $l = |v - v'|$ there exists at most two possible values of v' . Summing over the different values of l , using Eq. (3.5), and that it is a geometric progression we get

$$\begin{aligned} \Pr[S_1 \cap F^c] &= \sum_{l=t}^{D(\mathcal{C})-1} \Pr[(l = |v' - v|) \cap (S_1 \cap F^c)] \\ &\leq \sum_{l=t}^{D(\mathcal{C})-1} 2 \exp(-c|v - w|) \\ &= 2 \frac{1 - \exp(-cD(\mathcal{C}))}{1 - \exp(-c)} - 2 \frac{1 - \exp(-ct)}{1 - \exp(-c)} \\ (3.6) \quad &\leq \frac{2}{1 - \exp(-c)} \exp(-ct) \end{aligned}$$

For the event $S_2 \cap F^c$, observe only if $||v/g| - |v'/g|| \leq 1$ can both v and v' be considered during second decoding step. This means that at most $3g \leq 12d$ different v' values can be compared to v , the ones in the interval $[v - 6d; v + 6d]$. Of these, only the ones in the intervals $[v - 6d; v - D(\mathcal{C})]$ and $[v + D(\mathcal{C}); v + 6d]$ are actually part of the event $S_2 \cap F^c$ per definition. All other values of v' have a probability of 0 to be decoded by the fact that we are looking at events that are a subset of F^c . By Lemma 3.9 we have $H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(v')) \geq D(\mathcal{C})$. From

Lemma 3.7, we can therefore calculate the probabilities of these v' values as:

$$\begin{aligned}
\Pr[S_2 \cap F^c] &\leq \sum_{w \in [m]} \Pr[(v' = w) \cap (S_2 \cap F^c)] \\
&\leq \sum_{w=v-6d}^{v-D(\mathcal{C})} \Pr[(v' = w) \cap S_2 \cap F^c] + \sum_{w=v+D(\mathcal{C})}^{v+6d} \Pr[(v' = w) \cap S_2 \cap F^c] \\
&\leq \sum_{w=v-6d}^{v-D(\mathcal{C})} \exp\left(-\frac{(1-2p)^2}{4p+2} H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(w))\right) + \sum_{w=v+D(\mathcal{C})}^{v+6d} \exp\left(-\frac{(1-2p)^2}{4p+2} H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(w))\right) \\
(3.7) \quad &\leq 12d \exp(-cD(\mathcal{C})).
\end{aligned}$$

Finally, we can determine the probability of F happening as the union bound of the probability of any of the 3 component codes being decoded incorrectly, which means

$$(3.8) \quad \Pr[F] \leq P_p(D(\mathcal{C})) + 2P_p(D(\mathcal{L})) \leq 5P_p(D(\mathcal{C})),$$

where \mathcal{L} is a code obtained using Construction 3.2 on \mathcal{C} and the error probability is obtained from Lemma 3.2. Substituting Eqs. (3.6) to (3.8) into Eq. (3.4), we get

$$\Pr[|v - v'| \geq t] \leq \frac{2}{1 - \exp(-c)} \exp(-ct) + 12d \exp(-cD(\mathcal{C})) + 5P_p(\mathcal{C}),$$

as desired. \square

4 Concrete Error-Correcting Gray Codes

In this section, we will be looking at instantiating the presented codes, and what kinds of guarantees these give us. The idea is that we will instantiate Construction 3.4 with polar codes as well as expander codes to show the properties we are able to achieve with these codes.

First, we look at expander codes [14, 15]. These codes are linear codes that cannot quite reach the information theoretical limit, but on the other hand, they are robust when the noise is lower than their decoding limit. Selecting \mathcal{C} to be an expander code constructed to be able to handle a ratio of $\alpha < 1/4$ bit flips we will look at how our code performs with an error of $p = \alpha/2$. We furthermore know that expander codes can be encoded and decoded in $O(d)$ time [15].

LEMMA 4.1. *Let $\mathcal{C} = (\mathcal{C}_{\text{enc}}, \mathcal{C}_{\text{dec}})$ be an expander code with block length d that can correct all errors of at most αd bit flips, then*

$$P_{\alpha/2}(\mathcal{C}) \leq e^{-\alpha d/6}$$

Proof. Let Y be the number of errors. Since we are guaranteed to be able to handle αd errors and the expected number of errors is $E[Y] = \alpha d/2$, we have we have

$$P_{\alpha/2}(\mathcal{C}) \leq \Pr[Y \geq \alpha d] \leq \exp(-\alpha d/6)$$

using the Chernoff bound $\Pr[Y \geq 2E[Y]] \leq \exp(-E[Y]/3)$. \square

From this, we present the instantiation of our codes using expander codes,

COROLLARY 4.1. *Let \mathcal{C} be an expander code with block length $d = O_\alpha(\lg m)$ and message length $\lg m$ that can correct all errors of at most αd bit flips, where $\alpha < \frac{1}{4}$ and let $\mathcal{G} = (\mathcal{G}_{\text{enc}}, \mathcal{G}_{\text{dec}})$ be a code constructed using Construction 3.4 on \mathcal{C} . Then \mathcal{G} is a Gray code with message length of at least $\lg m$ and block length $d' \leq 6d$ such that for all $t, v \in [m]$.*

$$\Pr[|v - \mathcal{G}_{\text{dec}}(\mathcal{G}_{\text{enc}}(v) \oplus b_{\alpha/2})| \geq t] \leq \exp(-\Omega(t)) + \exp(-\Omega(d))$$

where $b_{\alpha/2} \sim \text{Bern}(\alpha/2)^{d'}$ with running time $O(d')$ for both encoding and decoding. In addition, we have $\mathcal{G}_{\text{enc}}(0) = 0^{d'}$

Proof. Let $c = (1 - 2p)^2/(4p + 2) = (1 - \alpha)^2/(2\alpha + 2) > \frac{9}{40}$. By Theorem 3.1, we have

$$\Pr[|v - v'| \geq t] \leq \frac{2}{1 - \exp(-c)} \exp(-ct) + 12de^{-cD(\mathcal{C})} + 5P_p(\mathcal{C}).$$

From evaluation we find $\frac{2}{1 - \exp(-c)} < 10$. Substituting the bound on c and using Lemma 4.1 yields:

$$\Pr[|v - \mathcal{G}_{\text{dec}}(\mathcal{G}_{\text{enc}}(v) \oplus b_{\alpha/2})| \geq t] < 10e^{-\frac{9}{40}t} + 12de^{-\frac{9}{40}D(\mathcal{C})} + 5e^{-\alpha d/6}$$

which can be simplified to the desired result since $D(\mathcal{C}) \geq 2\alpha d$.

To see that $\mathcal{G}_{\text{enc}}(0) = 0^{d'}$, by construction we have $\mathcal{G}_{\text{enc}}(0) = \mathcal{C}_{\text{enc}}(0)\mathcal{C}_{\text{enc}}(0)0^{D(\mathcal{C})}\mathcal{C}_{\text{enc}}(0)\mathcal{C}_{\text{enc}}(0)0^{D(\mathcal{C})}$. Then the fact that expander codes are linear implies $\mathcal{C}_{\text{enc}}(0) = 0^{d'}$. \square

Another family of codes that is worth considering for instantiation are polar codes [3]. These codes are of interest since they achieve the capacity of the information in the channel. For a polar code \mathcal{C} with message length $\lg m$ and block length d , the probability of error is $P_p(\mathcal{C}) = O(d^{-1/4})$, with a running time of the decoder and encoder of $O(d \lg d)$, see [3]. This leads to the corollary:

COROLLARY 4.2. *Let \mathcal{C} be a polar code with block length d and message length $\lg m$ and let $\mathcal{G} = (\mathcal{G}_{\text{enc}}, \mathcal{G}_{\text{dec}})$ be a code constructed using Construction 3.4 on \mathcal{C} . Then \mathcal{G} is a Gray code with message length of at least $\lg m$ and block length $d' \leq 6d$ such that for all $t, v \in [m]$*

$$\Pr[|v - \mathcal{G}_{\text{dec}}(\mathcal{G}_{\text{enc}}(v) \oplus b_p)| \geq t] \leq e^{-\Omega(t)} + O(d^{-1/4})$$

where $b_p \sim \text{Bern}(p)^{d'}$ with running time $O(d \lg d)$ for both encoding and decoding.

5 Acknowledgement

We thank the reviewers for constructive and detailed feedback. The authors are affiliated with Basic Algorithms Research Copenhagen (BARC), supported by the VILLUM Foundation grant 16582. Rasmus Pagh is supported by Providentia, a Data Science Distinguished Investigator grant from Novo Nordisk Fonden.

References

- [1] Jayadev Acharya, Clement Canonne, Yuhang Liu, Ziteng Sun, and Himanshu Tyagi. Distributed estimation with multiple samples per user: Sharp rates and phase transition. *Advances in Neural Information Processing Systems*, 34:18920–18931, 2021.
- [2] Jayadev Acharya, Yuhang Liu, and Ziteng Sun. Discrete distribution estimation under user-level local differential privacy. In *International Conference on Artificial Intelligence and Statistics*, pages 8561–8585. PMLR, 2023.
- [3] Erdal Arıkan. Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Transactions on Information Theory*, 55(7):3051–3073, July 2009. doi:10.1109/tit.2009.2021379.
- [4] Martin Aumüller, Christian Janos Lebeda, and Rasmus Pagh. Representing sparse vectors with differential privacy, low error, optimal space, and fast access. *Journal of Privacy and Confidentiality*, 12(2), 2022. doi:10.29012/jpc.809.
- [5] Irit Dinur, Shai Evra, Ron Livne, Alexander Lubotzky, and Shahar Mozes. Locally testable codes with constant rate, distance, and locality. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 357–374, 2022. doi:10.1145/3519935.3520024.
- [6] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of Theory of Cryptography Conference (TCC)*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006. doi:10.1007/11681878_14.
- [7] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3–4):211–407, 2014. doi:10.1561/04000000042.
- [8] Jørn Justesen. Class of constructive asymptotically good algebraic codes. *IEEE Transactions on Information Theory*, 18(5):652–656, 1972. doi:10.1109/TIT.1972.1054893.
- [9] Donald E. Knuth. *The Art of Computer Programming: Combinatorial Algorithms, Part 1*. Addison-Wesley Professional, 2011.

- [10] Pavel Panteleev and Gleb Kalachev. Asymptotically good quantum and locally testable classical LDPC codes. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 375–388, 2022. doi:10.1145/3519935.3520017.
- [11] Irving S. Reed and Gustave Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2):300–304, 1960. doi:10.1137/0108018.
- [12] Galen Reeves and Henry D. Pfister. Reed-Muller codes achieve capacity on BMS channels, 2021. arXiv:2110.14631.
- [13] Claude E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. doi:10.1002/j.1538-7305.1948.tb01338.x.
- [14] Michael Sipser and Daniel A. Spielman. Expander codes. *IEEE Transactions on Information Theory*, 42(6):1710–1722, 1996. doi:10.1109/18.556667.
- [15] Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Transactions on Information Theory*, 42(6):1723–1731, 1996. doi:10.1109/18.556668.
- [16] Wenjie Wang and Xiang-Gen Xia. A closed-form robust chinese remainder theorem and its performance analysis. *IEEE Trans. Signal Process.*, 58(11):5655–5666, 2010. doi:10.1109/TSP.2010.2066974.
- [17] Stanley L. Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965. doi:10.1080/01621459.1965.10480775.
- [18] Li Xiao, Xiang-Gen Xia, and Yu-Ping Wang. Exact and robust reconstructions of integer vectors based on multidimensional chinese remainder theorem (MD-CRT). *IEEE Transactions on Signal Processing*, 68:5349–5364, 2020. doi:10.1109/TSP.2020.3023584.
- [19] Sergey Yekhanin. Locally decodable codes: A brief survey. In *Proceedings of Coding and Cryptology*, volume 6639 of *Lecture Notes in Computer Science*, pages 273–282. Springer, 2011. doi:10.1007/978-3-642-20901-7_18.

Appendices

A Construction from a linear code

In this section, we show how to exploit the linear structure of codes to achieve better constants for the length of the code. The general idea is that we avoid having to use the constant consecutive distance code of Construction 3.3.

To make notation simpler, we will be using the canonical binary code for integers. We will be writing it as $\mathcal{B} = (\mathcal{B}_{\text{enc}}, \mathcal{B}_{\text{dec}})$. A property that is often achieved in the construction of codes is linearity. This property can be used to make some small optimisations to our construction.

DEFINITION A.1. A code $\mathcal{C} = (\mathcal{C}_{\text{enc}}, \mathcal{C}_{\text{dec}})$ where $\mathcal{C}_{\text{enc}} : [2^n] \rightarrow GF(2)^d$ and a decoding mapping $\mathcal{C}_{\text{dec}} : GF(2)^d \rightarrow [2^n]$ is called linear if there exists a $n \times d$ generator matrix $G \in GF(2)^{n \times d}$ such that $\mathcal{C}_{\text{enc}}(v) = \mathcal{B}_{\text{enc}}(v)^T G$ for all $v \in [2^n]$.

Previously, we used the constant consecutive distance code to encode and decode efficiently. However, it also meant that we ended up needing an even number of repetitions of the code \mathcal{C} . It is however not enough to only use 2 for a black-box error correcting code, so we required 4 repetitions of the underlying \mathcal{C} . In addition to this, we also had to use some additional padding to communicate the parity of the encoded integer. In this section, we show how we can achieve the same properties, using only 3 repetitions of \mathcal{C} and no additional padding, while only getting an

The following lemma is central to our ability to accomplish this.

Algorithm 1 An algorithm for computing $\sum_{i=1}^m H(\mathcal{C}_{\text{enc}}(i-1), \mathcal{C}_{\text{enc}}(i))$

function COUNTCODEWORDS($t \in \mathbb{N}$, $G : n \times d$ matrix such that $\mathcal{C}_{\text{enc}}(v) = \mathcal{B}(v)^T G$)

$v_1 \leftarrow \mathbf{0}$

$s \leftarrow 0$

for $i = 1 \dots n$ **do**

$v_i \leftarrow v_{i-1} \oplus G_i.$

$s \leftarrow s + \|v_i\|_1 \cdot \lfloor \frac{t}{2^i} + \frac{1}{2} \rfloor$

return s

$\triangleright G_i.$ denotes the i th row of G

LEMMA A.1. Let \mathcal{C} be a linear code with block length d and message length n and $n \times d$ generator matrix G . Then

Algorithm 1 computes

$$\text{COUNTCODEWORDS}(t, G) = \sum_{i=1}^t H(\mathcal{C}_{\text{enc}}(i-1), \mathcal{C}_{\text{enc}}(i))$$

in $O(nd)$ time.

Proof. The time complexity is clear from the fact that v_i takes $O(d)$ time to compute and that this is done for $i = 1, \dots, n$.

For correctness, observe that for all i , $\mathcal{B}_{\text{enc}}(i-1) \oplus \mathcal{B}_{\text{enc}}(i) = 0^{n-k}1^k$ for some k . Since \mathcal{C} is linear, by definition $\mathcal{C}_{\text{enc}}(v) = \mathcal{B}(v)^\top G$ for some $n \times d$ matrix G . Letting $G_{i\cdot}$ denote the i th row of G and $\mathcal{B}(v)_i$ the i th entry of $\mathcal{B}(v)$, we observe that

$$\mathcal{C}_{\text{enc}}(v) = \bigoplus_{i=1}^n (\mathcal{B}(v)_i \cdot G_{i\cdot})$$

Since $\mathcal{B}_{\text{enc}}(i-1)$ and $\mathcal{B}_{\text{enc}}(i)$ only differs on the last k bits, this implies that

$$(1.9) \quad H(\mathcal{C}_{\text{enc}}(i-1), \mathcal{C}_{\text{enc}}(i)) = \left\| \bigoplus_{i=1}^k G_{i\cdot} \right\|_1 = \|v_k\|_1$$

It now simply remains for each k value to count the number of i 's such that $\mathcal{B}_{\text{enc}}(i-1)$ and $\mathcal{B}_{\text{enc}}(i)$ differs exactly on the last k bits. As the i th bit flips every 2^{i-1} increases by 1, we observe that it flips to 1 every 2^i increases. As it starts at 0, we end up having the number of i 's that differ on exactly the k last bits be $\lfloor \frac{m}{2^i} + \frac{1}{2} \rfloor$. Combining this with Eq. (1.9) we get

$$\sum_{i=1}^m H(\mathcal{C}_{\text{enc}}(i-1), \mathcal{C}_{\text{enc}}(i)) = \sum_{k=1}^n \|v_k\|_1 \cdot \left\lfloor \frac{m}{2^i} + \frac{1}{2} \right\rfloor$$

which is exactly the value computed by Algorithm 1. \square

CONSTRUCTION A.1. Let \mathcal{C} be a code with block length d and message length m . We construct the code $\mathcal{W} = (\mathcal{W}_{\text{enc}}, \mathcal{W}_{\text{dec}})$ with block length $3d$ and message length of $\lg m$. Define

$$\mathcal{W}_{\text{enc}}(v) = \mathcal{C}_{\text{enc}}(v)\mathcal{C}_{\text{enc}}(v)\mathcal{C}_{\text{enc}}(v)$$

$$\mathcal{W}_{\text{dec}}(c_1c_2c_3) = \text{Median of } \{\mathcal{C}_{\text{dec}}(c_1), \mathcal{C}_{\text{dec}}(c_2), \mathcal{C}_{\text{dec}}(c_3)\}$$

Notice that we specifically use the median instead of a majority vote for decoding. The reason is we want to be able to decode the code, even if one of the codewords has been modified to the point where it can not be decoded. However, from our later construction, we also cannot guarantee that the two non-broken codewords encode the same value, just that the values are numerically adjacent. By selecting the median, we guarantee that one of the two non-broken codewords is the one returned.

With the constructions we now have, we can construct the linear code-based error-correcting Gray code. Note that though this code is based upon a linear code, we do not make any claims that the code is linear and in general it will not be linear.

CONSTRUCTION A.2. (LINEAR CODE BASED ERROR CORRECTING GRAY CODE) Let $\mathcal{C} = (\mathcal{C}_{\text{enc}}, \mathcal{C}_{\text{dec}})$ be a linear code with block length d and message length $\lg m$ and let \mathcal{W} be a code constructed using Construction A.1 on \mathcal{C} . We will construct the code $\mathcal{G} = (\mathcal{G}_{\text{enc}}, \mathcal{G}_{\text{dec}})$ with block length $3d$ and message length at least $\lg m$.

Let $s_v = H(\mathcal{W}_{\text{enc}}(v), \mathcal{W}_{\text{enc}}(v+1))$. Let $b_1^{(v)}, \dots, b_{s_v}^{(v)}$ be the bit indices where $\mathcal{W}_{\text{enc}}(v)$ and $\mathcal{W}_{\text{enc}}(v+1)$ are different in sorted order and define $b_0^{(v)} = 0$. Let $v = q + r$ such that

$$q = \sum_{i=1}^l H(\mathcal{C}_{\text{enc}}(i-1), \mathcal{C}_{\text{enc}}(i)) \leq v < \sum_{i=1}^{l+1} H(\mathcal{C}_{\text{enc}}(i-1), \mathcal{C}_{\text{enc}}(i))$$

for some l and $0 \leq r < H(\mathcal{C}_{\text{enc}}(l), \mathcal{C}_{\text{enc}}(l+1))$. Then

$$\mathcal{G}_{\text{enc}}(v) = \text{pref}_{b_r^{(l)}}(\mathcal{W}_{\text{enc}}(l+1)) \text{ suf}_{s_l - b_r^{(l)}}(\mathcal{W}_{\text{enc}}(l)).$$

We define the decoding function in the following way. For input c , let $t = \mathcal{W}_{\text{dec}}(c)$. Then define the bitstring $h_v \in \{0, 1\}^{s_v}$ such that

$$(h_v)_i = \begin{cases} 0 & \text{if } c_{b_i^{(v)}} = \mathcal{K}_{\text{enc}}(v)_i \\ 1 & \text{if } c_{b_i^{(v)}} = \mathcal{K}_{\text{enc}}(v+1)_i \end{cases}.$$

From this construction, h_v becomes a unary code for all v . Letting \mathcal{U}_{dec} be the function for decoding unary functions. We then end up with two alternative decodings, which we name

$$v_0 = \mathcal{U}_{\text{dec}}(h_{t-1}) + \sum_{i=1}^{t-1} H(\mathcal{C}_{\text{enc}}(i-1), \mathcal{C}_{\text{enc}}(i)),$$

$$v_1 = \mathcal{U}_{\text{dec}}(h_t) + \sum_{i=1}^t H(\mathcal{C}_{\text{enc}}(i-1), \mathcal{C}_{\text{enc}}(i)).$$

We define $\mathcal{G}_{\text{dec}}(c) = \arg \min_{v \in \{v_0, v_1\}} H(c, \mathcal{G}_{\text{enc}}(v))$. We refer to Lemma A.4 to show that this actually is a Gray code.

It is straightforward to see that q can be efficiently computed in time $O(n^2d)$ through a binary search on Algorithm 1.

LEMMA A.2. Let \mathcal{C} be a code with block length d and message length $\lg m$ and let $\mathcal{G} = (\mathcal{G}_{\text{enc}}, \mathcal{G}_{\text{dec}})$ be obtained using Construction A.2 on \mathcal{C} . Then for any $v \in [m-1]$

$$H(\mathcal{G}_{\text{enc}}(v), \mathcal{G}_{\text{enc}}(v+1)) = 1$$

Proof. (Sketch) Use the same approach as in Lemma 3.4, but use the definitions of q , r and $\mathcal{G}_{\text{enc}}(v)$ as they are in Construction A.2. \square

LEMMA A.3. Let \mathcal{C} be a code and let $\mathcal{G} = (\mathcal{G}_{\text{enc}}, \mathcal{G}_{\text{dec}})$ be constructed using Construction A.2 on \mathcal{C} . Then \mathcal{G}_{enc} is injective.

Proof. (Sketch) The approach is exactly the same as in Lemma 3.5, except that there are only three codewords instead of four. Instead of using $\lfloor v/g \rfloor$ to determine q , use the same method as in Construction A.2. \square

LEMMA A.4. Let \mathcal{C} be a code with block length d and message length $\lg m$ and let \mathcal{G} be constructed using Construction A.2 on \mathcal{C} . Then \mathcal{G} is a Gray code.

Proof. (Sketch) The structure of the proof is essentially the same as for Lemma 3.6. To show that Construction A.2 is a Gray code, we start by showing that it is a code. Let $v \in [m]$ and let l be the unique integer such that

$$\sum_{i=1}^l H(\mathcal{C}_{\text{enc}}(i-1), \mathcal{C}_{\text{enc}}(i)) \leq v < \sum_{i=1}^{l+1} H(\mathcal{C}_{\text{enc}}(i-1), \mathcal{C}_{\text{enc}}(i))$$

and let $\mathcal{G}_{\text{enc}}(v) = c_1 c_2 c_3$ such that $c_1, c_2, c_3 \in \{0, 1\}^d$. From construction, it holds that for at most 1 codeword s of $\{\mathcal{C}_{\text{dec}}(c_1), \mathcal{C}_{\text{dec}}(c_2), \mathcal{C}_{\text{dec}}(c_3)\}$ that $s \notin \{l, l+1\}$. Regardless of the value of s , the median of $\mathcal{C}_{\text{dec}}(c_1)$, $\mathcal{C}_{\text{dec}}(c_2)$ and $\mathcal{C}_{\text{dec}}(c_3)$ is either l or $l+1$. From here, the approach is the same as for Lemma 3.6. \square

THEOREM A.1. Let $\mathcal{C} = (\mathcal{C}_{\text{dec}}, \mathcal{C}_{\text{enc}})$ be a code with block length d and message length m , and let $\mathcal{G} = (\mathcal{G}_{\text{dec}}, \mathcal{G}_{\text{enc}})$ be obtained using Construction A.2 on \mathcal{C} . Let $p \in (0, 1)$ and let $b_p \sim \text{Bern}(p)^{3d}$. Let $c = (1-2p)^2 / (4p+2)$. Then for all $t \geq 0$,

$$\Pr[|v - v'| \geq t] \leq \frac{2}{1 - \exp(-c)} \exp(-ct) + 9d \exp(-cD(\mathcal{C})) + 2P_p(\mathcal{C})$$

Proof. (Sketch) The proof of this is mostly equivalent to the proof of Theorem 3.1, except that Construction A.2 only consists of 3 copies of \mathcal{C} and there are at most $9d$ values of v' which can be considered together with v . \square

B Proof of Lemma 3.1

In this section we prove Lemma 3.1. The general idea is to show that for two chosen codewords, at least one of them must be decoded incorrectly with some bounded probability.

Proof. Pick $v, w \in [m]$ such that $H(\mathcal{C}_{enc}(v), \mathcal{C}_{enc}(w)) = D(\mathcal{C})$ and let $b_p \sim \text{Bern}(p)^d$. Observe that since $p < \frac{1}{2}$,

$$(2.10) \quad \Pr \left[\|c_p\|_1 > \frac{D(\mathcal{C})}{2} \right] \leq \Pr \left[\|c_p\|_1 < \frac{D(\mathcal{C})}{2} \right]$$

Let $c_p \in \{0, 1\}^{D(\mathcal{C})}$ be the random bitstring where each bit in c_p corresponds to a unique bit in b_p where $\mathcal{C}_{enc}(v)$ and $\mathcal{C}_{enc}(w)$ are different. Let $(c_p)_i = 1$ if and only if the corresponding bit in b_p is 1. Observe that $c_p \sim \text{Bern}(p)^{D(\mathcal{C})}$. This means that $\|c_p\|_1$ becomes a count over how many of the bits where $\mathcal{C}_{enc}(v)$ and $\mathcal{C}_{enc}(w)$ are different have been flipped.

For the sake of notation, let $I(x)$ be the event $\mathcal{C}_{dec}(\mathcal{C}_{enc}(x) \oplus b_p) \neq x$ i.e., the event that the decoding fails. Next, observe that if $x = \mathcal{C}_{enc}(v) \oplus b_p$, and x differs from $\mathcal{C}_{enc}(v)$ in r of the positions where $\mathcal{C}_{enc}(v)$ and $\mathcal{C}_{enc}(w)$ differ, then it must hold that x differs from $\mathcal{C}_{enc}(w)$ in exactly $D(\mathcal{C}) - r$ of these positions. This means that we can bound the probability as

$$(2.11) \quad \Pr \left[I(v) \mid \|c_p\|_1 < \frac{D(\mathcal{C})}{2} \right] + \Pr \left[I(w) \mid \|c_p\|_1 > \frac{D(\mathcal{C})}{2} \right] \geq 1,$$

and the same symmetrically

$$(2.12) \quad \Pr \left[I(v) \mid \|c_p\|_1 > \frac{D(\mathcal{C})}{2} \right] + \Pr \left[I(w) \mid \|c_p\|_1 < \frac{D(\mathcal{C})}{2} \right] \geq 1,$$

and finally

$$(2.13) \quad \Pr \left[I(v) \mid \|c_p\|_1 = \frac{D(\mathcal{C})}{2} \right] + \Pr \left[I(w) \mid \|c_p\|_1 = \frac{D(\mathcal{C})}{2} \right] \geq 1,$$

since the conditional restrict to the same set of events and the decoding means the complements are disjoint.

Now let S the event that $\|c_p\| < \frac{D(\mathcal{C})}{2}$, T the event that $\|c_p\| = \frac{D(\mathcal{C})}{2}$, and U the event that $\|c_p\| > \frac{D(\mathcal{C})}{2}$. This means we can write

$$\Pr[I(v)] = \Pr[I(v)|S] \Pr[S] + \Pr[I(v)|T] \Pr[T] + \Pr[I(v)|U] \Pr[U]$$

$$\Pr[I(w)] = \Pr[I(w)|S] \Pr[S] + \Pr[I(w)|T] \Pr[T] + \Pr[I(w)|U] \Pr[U]$$

Using Eqs. (2.10) to (2.13), we get

$$\begin{aligned} \Pr[I(v)] + \Pr[I(w)] &\geq (\Pr[I(v)|S] + \Pr[I(w)|U]) \Pr[U] + (\Pr[I(v)|T] + \Pr[I(w)|T]) \Pr[T] \\ &\quad + (\Pr[I(v)|U] + \Pr[I(w)|S]) \Pr[S] \\ &\geq \Pr[T] + 2 \Pr[U]. \end{aligned}$$

Since at least one of $\Pr[I(v)]$ and $\Pr[I(w)]$ must be greater than the average, this proves the lemma. \square

C Proof of Theorem 1.2

We follow the approach of Aumüller, Lebeda and Pagh [4] for the case of pure differential privacy. This section is intended to be read alongside parts of their paper that we refer to and modify.¹ For a multiset S we denote the frequency of element $i \in S$ by x_i and let $x_i = 0$ for $i \in [u] \setminus S$, such that S is encoded by $x \in \{0, 1\}^u$ which is n -sparse. We modify the ALP1-Projection algorithm (Algorithm 2 in [4]) by replacing the choice of $z_{a,b}$ in

¹We note that the variables m and α used in the present paper to denote parameters of codes are used for other quantities in [4]. Conversely, in this section, we use b consistently with [4] to denote an index in the data structure, while b is used in previous sections to denote an error vector.

step (2). ALP1-Projection defines values y_i that are downscaled and rounded versions of x_i , and implicitly uses a unary encoding of y_i . We replace the unary encoding by a sensitivity 1 error-correcting code $\mathcal{G} = (\mathcal{G}_{dec}, \mathcal{G}_{enc})$ given by our Corollary 4.1 with decoding parameter $\alpha = 1/5$. Specifically,

$$z_{a,b} = \begin{cases} 1, & \exists i \in S : \mathcal{G}_{enc}(y_i)_b = 1 \text{ and } h_b(i) = a \\ 0, & \text{otherwise} \end{cases}.$$

Note that this requires the number of hash functions (denoted by m in [4]) to be equal to the block size d' of the code \mathcal{G} , discussed below. Since \mathcal{G}_{enc} has sensitivity 1 and since $\mathcal{G}_{enc}(0) = 0^2$ we see that adding an element to S changes at most one value $z_{a,b}$, and the same holds for removing an element. This means that the privacy of the modified ALP1-Projection algorithm, which applies randomized response to each bit $z_{a,b}$, follows exactly as in [4].

The ALP1-Estimator algorithm (Algorithm 3 in [4]), tailored to decoding the noisy unary encoding, must be replaced by running \mathcal{G}_{dec} on the relevant bits of the data structure. For $i \in [u]$ if we let $v^{(i)} \in \{0, 1\}^{d'}$ be the vector given by $v_b^{(i)} = \tilde{z}_{h_b(i), b}$, on input i the new ALP1-Estimator returns $\mathcal{G}_{dec}(v^{(i)})$. We can write $v^{(i)} = \mathcal{G}_{enc}(v) \oplus \eta^{(i)}$ for a vector $\eta^{(i)}$ where $\eta_b^{(i)}$ represents errors due to a hash collision $h_b(i) = h_b(i')$ for some $i' \in S \setminus \{i\}$ or due to a bit flip introduced by randomized response on the variable $z_{h_b(i), b}$. Since the hash functions and noise bits are independent, the bits of $\eta^{(i)}$ are independent. Corollary 4.1 is stated for noise distribution $\text{Bern}(\alpha/2)^{d'}$, but since expander codes enjoy a worst-case guarantee on decoding radius it is easy to see that the result holds as long as there is an *upper bound* of $\alpha/2$ on the probability of flipping each bit. Thus, it suffices to argue that for each b , $\Pr[\eta_b^{(i)} = 1] \leq 1/10$. This can be achieved by a union bound on two events: 1) bit b is flipped by randomized response, and 2) there is a hash collision $h_b(i) = h_b(i')$. Choosing the hash table size s and the randomized response parameter such that the probability for each of these events is bounded by $1/20$ changes only constant factors in the error bound and space usage. Here we make use of the fact that privacy for ALP1-Projection relies on a combination of scaling and randomized response so that we can choose the parameter of randomized response to be any constant without affecting privacy.

The ALP-Projection/Estimation algorithms (Algorithms 4 and 5 in [4]) that scale the inputs to achieve sensitivity ε work unchanged, except for relying on the changed ALP1-Projection/Estimation algorithms. To finalize the argument, the Threshold ALP-Projection/Estimation algorithms (Algorithms 7 and 8 in [4]) are changed to use the new versions of the ALP1 Projection/Estimator algorithms. Since it suffices to encode numbers up to $\ell = O(\log u)$, we can choose the code with message length $\log_2 \ell = O(\log \log u)$, and hence also with block length $d' = O(\log \log u)$. Since decoding takes expected $O(d')$ time, estimating an item frequency can be done in expected time $O(\log \log u)$.

Finally, to bound the estimation error we consider the two cases in Threshold ALP-Estimation. If the output is generated by the threshold Laplace mechanism the expected and high-probability bounds follow from the standard analysis of the Laplace mechanism. Otherwise, if the output is generated by ALP-Estimation we note that the final error is the error $|\mathcal{G}_{dec}(v^{(i)}) - y_i|$ of the error-correcting code multiplied by $O(1/\varepsilon)$. By Corollary 4.1, using the bound on $\eta^{(i)}$ above, this error is $O(1)$ in expectation and is bounded by $\ell = O(\log(u))$ with probability 1 since we always decode to $[\ell]$. This finishes the error analysis.

²The property that $\mathcal{G}_{enc}(0) = 0$ holds for the particular code in Corollary 4.1 but can be achieved in general by permuting the codewords using $x \mapsto x \oplus \mathcal{G}_{enc}(0)$, changing no other properties.