Coding for the ℓ_∞-Limited Permutation Channel

Michael Langberg, *Senior Member, IEEE*, Moshe Schwartz, *Senior Member, IEEE*, and Eitan Yaakobi, *Senior Member, IEEE*

***Abstract*— We consider the communication of information in the presence of synchronization errors. Specifically, we consider *permutation channels* in which a transmitted codeword $x = (x_1, \ldots, x_n)$ is corrupted by a permutation $\pi \in S_n$ to yield the received word $y = (y_1, \ldots, y_n)$, where $y_i = x_{\pi(i)}$. We initiate the study of worst case (or zero-error) communication over permutation channels that distort the information by applying permutations π, which are limited to displacing any symbol by at most r locations, i.e., permutations π with weight at most r in the ℓ_∞-metric. We present direct and recursive constructions, as well as bounds on the rate of such channels for binary and general alphabets. Specific attention is given to the case of $r = 1$.**

***Index Terms*— Permutation channel, ℓ_∞-metric.**

I. Introduction

PERMUTATION channels have received some attention in recent years due to their relevance in different applications of networking technologies and various read channels. Under this setup, a vector of symbols is transmitted in some order, but due to synchronization errors, the symbols received are not necessarily in the order in which they were transmitted, e.g., [12], [13], [27] (permutation channels), [14], [23] (the bit-shift magnetic recording channel), and [19] (the trapdoor channel).

We can think of the channel as applying a permutation to the transmitted vector. However, not all permutations may be equally likely, or even feasible. In this work we focus on channels that can only displace symbols a limited amount of positions away from their origin. Such permutations are exactly those that have a limited weight in the ℓ_∞-metric over permutations.

When the transmitted vectors are in themselves permutations, this channel has been studied as the limited-magnitude rank-modulation channel. In particular, error-correcting codes were studied [11], [24], [25], as well as systematic codes [30], anticodes [22], covering codes [5], [28], and various other related combinatorial problems [10], [17], [21].

Unlike the rank-modulation case, this work considers the transmission of general vectors over the channel, and in particular, allows repeated symbols and small alphabets. More specifically, for a finite alphabet Σ, the transmitted codeword $x = (x_1, \ldots, x_n)$ may be any element in Σ^n. The codeword x is corrupted by a permutation $\pi \in S_n$ to yield $y = (y_1, \ldots, y_n)$ where $y_i = x_{\pi(i)}$. We consider the worst-case (or zero-error) communication model over permutations π for which $\forall i : |i - \pi(i)| \leqslant r$ for a pre-specified magnitude r, i.e., the weight of π is at most r in the ℓ_∞-metric. We refer to such channels as ℓ_∞-limited permutation channels, $\mathrm{LPC}_\infty(r)$.

In this work we initiate the study of $\mathrm{LPC}_\infty(r)$ for general alphabets Σ and magnitudes r under the worst-case setting. Although similar models have been studied in the literature, to the best of our knowledge, the study of zero error $\mathrm{LPC}_\infty(r)$ has not been explicitly addressed. Most closely related models include the permutation model of [14] and [23] in which $\Sigma = \{0, 1\}$ but the limitation $|i - \pi(i)| \leqslant r$ on permutations π holds only for i such that $x_i = 1$, [13] which has a model similar to ours but applies a random permutation instead of a worst-case one, [12] in which random synchronization errors of limited ℓ_∞-norm are applied to vectors of natural numbers, and [27] in which the channel is governed by a distribution over S_n.

We present direct and recursive code constructions, encoding and decoding algorithms, bounds on code parameters, and constructions for covering codes for $\mathrm{LPC}_\infty(r)$. Specifically, our model and preliminaries are given in Section II. In Section III we study the combinatorial properties of $\mathrm{LPC}_\infty(r)$ including the average and precise size of balls according to the ℓ_∞-metric. In Section IV we present codes for $\mathrm{LPC}_\infty(r)$. Finally, in Section V we present general upper bounds on the size of codes for $\mathrm{LPC}_\infty(r)$ via covering codes together with the comparison of our lower and upper bounds for some specific settings of parameters. Our main focus in several of the sections above is on general $|\Sigma| = q$ and $r = 1$, and only at times do we address larger values of r.

II. Preliminaries

Assume a finite alphabet Σ. The notation for a vector $x \in \Sigma^n$ will usually be given by a list of its components, separated by commas, and surrounded by parentheses, i.e., $x = (x_1, x_2, \ldots, x_n)$. We shall sometimes, however, call this vector a *string*, and denote its components without the commas and parentheses, i.e., $x_1\, x_2 \ldots x_n$.

Manuscript received May 26, 2016; revised March 3, 2017; accepted October 3, 2017. Date of publication October 13, 2017; date of current version November 20, 2017. M. Langberg was supported by BSF under Grant 2010075. M. Schwartz was supported by the Israel Science Foundation (ISF) under Grant 130/14. E. Yaakobi was supported by ISF under Grant 1624/14. The work was presented in part at the 2014 IEEE International Symposium on Information Theory [16].

M. Langberg is with the Department of Electrical Engineering, State University of New-York at Buffalo, Buffalo, NY 14260 USA (e-mail: mikel@buffalo.edu).

M. Schwartz is with the Department of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Beer Sheva 8410501, Israel (e-mail: schwartz@ee.bgu.ac.il).

E. Yaakobi is with the Department of Computer Science, Technion–Israel Institute of Technology, Haifa 32000, Israel (e-mail: yaakobi@cs.technion.ac.il).

Communicated by C.-C. Wang, Associate Editor for Coding Techniques.

Digital Object Identifier 10.1109/TIT.2017.2762676

Let us denote $[n] = \{1, 2, \ldots, n\}$, and let S_n denote the set of all permutations over $[n]$. A permutation $\pi \in S_n$ is written in vector notation $\pi = [\pi_1, \pi_2, \ldots, \pi_n]$, and may be considered a bijection $\pi : [n] \to [n]$ mapping $\pi(i) = \pi_i$. The identity permutation is denoted by $\mathrm{Id} = [1, 2, \ldots, n]$.

Given two permutations, $\pi, \pi' \in S_n$, the ℓ_∞-distance between the two is defined as

$$d_\infty(\pi, \pi') = \max_{i \in [n]} \left|\pi(i) - \pi'(i)\right|.$$

The ℓ_∞-distance defines a metric [4]. The *weight* of a permutation $\pi \in S_n$ is defined as

$$\mathrm{wt}_\infty(\pi) = d_\infty(\pi, \mathrm{Id}).$$

Thus, all the permutations of weight at most r form the ball of radius r centered at the identity permutation. Balls in the ℓ_∞-metric over permutations have been studied in the past [10], [17], [21].

We now formally introduce the ℓ_∞-limited permutation channel, $\mathrm{LPC}_\infty(r)$.

Definition 1: Let Σ be some finite alphabet. Assume a vector $x = (x_1, x_2, \ldots, x_n) \in \Sigma^n$ has been transmitted. The $\mathrm{LPC}_\infty(r)$ *channel distorts it by applying to it a permutation of weight at most r. Thus, the received vector $y = (y_1, y_2, \ldots, y_n) \in \Sigma^n$ satisfies $y = \pi x$, i.e.,*

$$y_i = x_{\pi(i)} \quad \text{for all } i \in [n],$$

for some permutation $\pi \in S_n$ with $\mathrm{wt}_\infty(\pi) \leqslant r$.

Definition 2: The ball *of radius r centered at $x \in \Sigma^n$ is*

$$B_r(x) = \{\pi x \mid \pi \in S_n, \mathrm{wt}_\infty(\pi) \leqslant r\}.$$

It follows that a vector $x \in \Sigma^n$ transmitted over $\mathrm{LPC}_\infty(r)$ may be received as any vector in $B_r(x)$. This gives rise to the following definition of an error-correcting code for $\mathrm{LPC}_\infty(r)$.

Definition 3: Let Σ be a finite alphabet of size q, and $C \subseteq \Sigma^n$. We say C is an $(n, M; r)_q$-LPC_∞ code if its size is $|C| = M$, and for all $c, c' \in C$, $c \neq c'$, we have

$$B_r(c) \cap B_r(c') = \emptyset.$$

In an analogous fashion we also define covering codes.

Definition 4: Let Σ be a finite alphabet and $C \subseteq \Sigma^n$. We say C is an $(n, M)_q R$-LPC_∞ covering code if its size is $|C| = M$, and

$$\bigcup_{c \in C} B_R(c) = \Sigma^n.$$

Definition 5: Let Σ be some finite alphabet, $|\Sigma| = q$, and $C \subseteq \Sigma^n$ a code (be it error-correcting or covering). The rate *of the code C is defined, as usual, as*

$$R(C) = \frac{1}{n} \log_q |C|.$$

The sizes of the largest code, and the smallest covering code, are now defined. We use a notation similar to [2] and [18].

Definition 6: Let $\Sigma = \mathbb{Z}_q$ be the alphabet. Given n and r, we denote by $A_q(n; r)$ the largest M such that there exists an $(n, M; r)_q$-LPC_∞ code over Σ. Similarly, given n and R, we denote by $K_q(n; R)$ the smallest M such that there exists an $(n, M)_q R$-LPC_∞ covering code over Σ.

Let Σ be some finite alphabet. We recall some useful notation commonly used in the theory of formal languages. An n-string $x = x_1 x_2 \ldots x_n \in \Sigma^n$ is a finite sequence of alphabet symbols, $x_i \in \Sigma$. We say n is the length of x and denote it by $|x| = n$. For two strings, $x \in \Sigma^n$ and $y \in \Sigma^m$, their concatenation is denoted by $xy \in \Sigma^{n+m}$. The set of all finite strings over the alphabet Σ is denoted by Σ^*. For $s \in \Sigma^*$ and a non-negative integer k, we use s^k to denote the sequence obtained by concatenating k copies of s.

III. Properties of the $\mathrm{LPC}_\infty(r)$ Space

In this section we study several properties of the $\mathrm{LPC}_\infty(r)$ space, including the size of balls, and the distance between vectors.

Definition 7: The LPC_∞-distance *between $x, y \in \Sigma^n$, denoted $d(x, y)$, is defined as the minimum non-negative integer w such that there exists a permutation $\pi \in S_n$, $\mathrm{wt}_\infty(\pi) = w$, and $y = \pi x$. If no such integer exists we say the distance is ∞.*

For any symbol $a \in \Sigma$, we denote by $n_a(x)$ the number of occurrences of a in x, i.e.,

$$n_a(x) = |\{i \in [n] \mid x_i = a\}|.$$

Additionally, the index of the jth occurrence of a in x is denoted as $L_a(j; x)$. More precisely, $L_a(j; x) = i$ if $x_i = a$ and a appears exactly $j - 1$ times in the string $x_1\, x_2 \ldots x_{i-1}$. We say two strings, $x, y \in \Sigma^n$, have the same *composition* if $n_a(x) = n_a(y)$ for all $a \in \Sigma$.

One can easily observe that the LPC_∞-distance between two strings is ∞ if and only if their composition differs.

Lemma 8: Let $X \subseteq \Sigma^n$ be a set of strings of equal composition. Then the LPC_∞*-distance function defines a metric over X.*

Proof: Let $x, y \in \Sigma^n$. It is easy to verify that $d(x, y) = 0$ if and only if $x = y$.

The ℓ_∞-metric over S_n is right invariant (see [4]), i.e., for all $\pi, \sigma, \tau \in S_n$,

$$d_\infty(\pi, \sigma) = d_\infty(\pi\tau, \sigma\tau),$$

where $\pi\tau$ is the permutation composition of π and τ, and similarly $\sigma\tau$. Thus,

$$\mathrm{wt}_\infty(\pi) = d_\infty(\pi, \mathrm{Id}) = d_\infty(\pi\pi^{-1}, \pi^{-1}) = \mathrm{wt}_\infty(\pi^{-1}).$$

It then follows that the LPC_∞ distance is symmetric.

Finally, the triangle inequality holds, $d(x, y) \leqslant d(x, z) + d(z, y)$, for all x, y, and z. To see this, assume the minimal-weight permutations that determine the distances are $\pi_1 x = z$, $\pi_2 z = y$, and $\pi x = y$. Then,

$$\begin{aligned} d(x, z) + d(z, y) &= \mathrm{wt}_\infty(\pi_1) + \mathrm{wt}_\infty(\pi_2) \\ &= \mathrm{wt}_\infty(\pi_1) + \mathrm{wt}_\infty(\pi_2^{-1}) \\ &\geqslant \mathrm{wt}_\infty(\pi_1\pi_2) \geqslant \mathrm{wt}_\infty(\pi) = d(x, y), \end{aligned}$$

where the first inequality is due to the triangle inequality in the ℓ_∞-metric over permutations, and the second is due to the

fact that $\pi_1\pi_2\, x = y$, but $\pi_1\pi_2$ is not necessarily the minimal-weight permutation changing x into y. Thus, when restricting ourselves to sets of vectors with the same composition, the distance function d defines a metric. ∎

The next theorem states how to find $d(x, y)$ and the corresponding permutation connecting x and y. Before proving it we require the following auxiliary lemma.

Lemma 9: Let $x_1, x_2, y_1, y_2 \in \mathbb{R}$ be real numbers such that $x_1 \leqslant x_2$ and $y_1 \leqslant y_2$. Then

$$\max\left(|x_1 - y_1|, |x_2 - y_2|\right) \leqslant \max\left(|x_1 - y_2|, |x_2 - y_1|\right).$$

Proof: A proof may be obtained by a tedious examination of the six possible configurations of x_1, x_2, y_1, y_2 when sorted in ascending order. ∎

Theorem 10: Let Σ be a finite alphabet, and $x, y \in \Sigma^n$ such that $d(x, y) < \infty$. Then

$$d(x, y) = \max_{\substack{a\in\Sigma \\ j\in[n_a(x)]}} |L_a(j; x) - L_a(j; y)|.$$

In addition, finding π such that $y = \pi x$ and $\mathrm{wt}_\infty(\pi) = d(x, y)$ can be done in $O(n\,|\Sigma|)$ time.

Proof: Let us define the permutation $\pi \in S_n$ mapping $L_a(j; y) \mapsto L_a(j; x)$ for all $a \in \Sigma$ and $j \in [n_a(x)]$. It is easily seen that

$$y = \pi x,$$

and that

$$\mathrm{wt}_\infty(\pi) = \max_{\substack{a\in\Sigma \\ j\in[n_a(x)]}} |L_a(j; x) - L_a(j; y)|.$$

We further contend that π has minimum weight of all permutations mapping x to y, and that will complete the proof for the claim.

Let $\pi' \in S_n$ be a permutation such that $y = \pi x$, and $\pi' \neq \pi$. Let $a \in \Sigma$ and $j \in [n_a(x)]$ be such that $\pi'(L_a(j; y)) \neq \pi(L_a(j; y))$, and assume j is the minimal integer with this property. By our construction of π we must have

$$t_1 = \pi'(L_a(j; y)) > \pi(L_a(j; y)). \tag{1}$$

Furthermore, we must therefore have some index $j' > j$ such that

$$t_2 = \pi'(L_a(j'; y)) = \pi(L_a(j; y)). \tag{2}$$

Thus, rearranging π' by setting

$$\pi'(L_a(j; y)) = t_2$$
$$\pi'(L_a(j'; y)) = t_1,$$

where t_1 and t_2 are defined in (1) and (2) respectively, and by using Lemma 9 we are not increasing the weight of π'. Repeating the process we get that π has minimal weight as claimed. We note that the proof above allows finding π in $O(n\,|\Sigma|)$ time. ∎

On several occasions in the following sections, we will focus specifically on $(n; 1)_q$-LPC$_\infty$ codes. We therefore study in more detail balls of radius 1. Assume the alphabet is $\Sigma = \mathbb{Z}_q$, and let $x = (x_1, x_2, \ldots, x_n) \in \mathbb{Z}_q^n$ be some vector. The number of permutations $\pi \in S_n$ such that $\mathrm{wt}_\infty(\pi) \leqslant 1$ is known to be the n-th Fibonacci number F_n (see [17], [21]), where

$$F_i = \begin{cases} F_i = F_{i-1} + F_{i-2} & i \geqslant 2 \\ F_i = 1 & i = 0, 1 \end{cases}$$

Thus, we immediately get that

$$|B_1(x)| \leqslant F_n.$$

However, it is also clear that applying distinct permutations to x does not always result in distinct vectors.

Definition 11: Let $x \in \mathbb{Z}_q^n$ be some string. Given two permutations $\pi, \pi' \in S_n$, with $\mathrm{wt}_\infty(\pi) = \mathrm{wt}_\infty(\pi') \leqslant 1$, we say they are x-equivalent, denoted $\pi \overset{x}{\sim} \pi'$, if $\pi x = \pi' x$.

It is obvious that $\overset{x}{\sim}$ is an equivalence relation, and that $|B_1(x)|$ is the number of equivalence classes of $\overset{x}{\sim}$.

We also note that every permutation $\pi \in S_n$ with $\mathrm{wt}_\infty(\pi) \leqslant 1$ can be written uniquely as a product of non-overlapping adjacent transpositions, and more precisely,

$$\pi = \prod_{i\in[k]} (j_i, j_i + 1), \tag{3}$$

with $j_i + 1 < j_{i+1}$ for all $i \in [k - 1]$. Here, (a, b), $a \neq b$, is the cycle notation for the permutation exchanging a and b while fixing all other elements. Additionally, $\prod$, as it appears in (3), when applied to permutations, denotes permutation composition.

We also introduce a new operator on permutations.

Definition 12: Let $x \in \mathbb{Z}_q^n$ be some string, and $\pi \in S_n$, $\mathrm{wt}_\infty(\pi) \leqslant 1$, some permutation. Assume the notation of (3)*. The x-*reduced form *of π is defined by*

$$\mathrm{rdc}_x(\pi) = \prod_{\substack{i\in[k] \\ x_{j_i} \neq x_{j_i+1}}} (j_i, j_i + 1).$$

Intuitively, the x-reduced form of π keeps only those transpositions that switch the positions of distinct symbols in x. By definition we have the following simple observation: for every $x \in \mathbb{Z}_q^n$ and every $\pi \in S_n$, $\mathrm{wt}_\infty(\pi) \leqslant 1$, we have

$$\pi x = \mathrm{rdc}_x(\pi) x. \tag{4}$$

In addition, the operator $\mathrm{rdc}_x(\cdot)$ characterizes the equivalence relation $\overset{x}{\sim}$.

Lemma 13: For $x \in \mathbb{Z}_q^n$, $\pi, \pi' \in S_n$, $\mathrm{wt}_\infty(\pi), \mathrm{wt}_\infty(\pi') \leqslant 1$, we have $\pi \overset{x}{\sim} \pi'$ if and only if $\mathrm{rdc}_x(\pi) = \mathrm{rdc}_x(\pi')$.

Proof: In the first direction, let $\pi \overset{x}{\sim} \pi'$, and thus by (4), $\mathrm{rdc}_x(\pi)x = \mathrm{rdc}_x(\pi')x$, but assume to the contrary that $\mathrm{rdc}_x(\pi) \neq \mathrm{rdc}_x(\pi')$. Let $J = \{j_1, j_2, \ldots, j_k\}$, respectively $J' = \{j'_1, j'_2, \ldots, j'_{k'}\}$, be the relevant integers in a decomposition of $\mathrm{rdc}_x(\pi)$, respectively $\mathrm{rdc}_x(\pi')$, as in (3). Let $i = \min(J \triangle J')$, where $\triangle$ denotes the symmetric difference operator on sets. By definition we must have $x_i \neq x_{i+1}$, and therefore, x_i appears as the ith element of either $\mathrm{rdc}_x(\pi)x$ or $\mathrm{rdc}_x(\pi')x$, whereas x_{i+1} appears as the ith element of the other. It follows that $\mathrm{rdc}_x(\pi)x \neq \mathrm{rdc}_x(\pi')x$, a contradiction.

In the other direction we have $\mathrm{rdc}_x(\pi) = \mathrm{rdc}_x(\pi')$, and assume to the contrary that $\pi \overset{x}{\not\sim} \pi'$. But that means that

$\pi x \neq \pi' x$, which implies by (4) that $\mathrm{rdc}_x(\pi)x \neq \mathrm{rdc}_x(\pi')x$, which is a contradiction. ∎

It now follows that $|B_1(x)|$ is exactly the number of x-reduced permutations. As already observed, x-reduced permutations are uniquely defined by a product of non-overlapping adjacent transpositions, exchanging positions in x with distinct symbols.

Definition 14: Given a vector $x = (x_1, x_2, \ldots, x_n) \in \mathbb{Z}_q^n$, an antirun *of length $\ell + 1$ is a subsequence $(x_j, x_{j+1}, \ldots, x_{j+\ell})$ such that $x_{j+i} \neq x_{j+i-1}$ for all $i \in [\ell]$. A* maximum antirun *is an antirun that cannot be extended in either direction.*

Any sequence of $x \in \mathbb{Z}_q^n$ can be partitioned uniquely into a sequence of maximal antiruns. We call the sequence of the lengths of the maximal antiruns in such a partition, the *antirun profile of x*, and denote it as $\mathcal{P}(x)$.

Example 15: Let $\Sigma = \mathbb{Z}_3$, and take

$$x = (1, 1, 2, 0, 1, 0, 2, 2, 2, 2, 0, 0, 1, 2, 0).$$

We note that $(x_3, x_4, x_5) = (2, 0, 1)$ is an antirun, however it is not a maximal antirun since it may be extended. The partition of x into maximal antiruns produces

$$(1), (1, 2, 0, 1, 0, 2), (2), (2), (2, 0), (0, 1, 2, 0).$$

Thus, the antirun profile of x is

$$\mathcal{P}(x) = (1, 6, 1, 1, 2, 4).$$

□

Theorem 16: Let $x \in \mathbb{Z}_q^n$ be a vector with an antirun profile $\mathcal{P}(x) = (\ell_1, \ell_2, \ldots, \ell_k)$. Then $|B_1(x)| = \prod_{i\in[k]} F_{\ell_i}$.

Proof: By the previous discussion, $|B_1(x)|$ is given by the number of x-reduced permutations. These permutations decompose uniquely into a product of non-overlapping adjacent transpositions, where each such adjacent transposition is allowed in coordinates containing two distinct symbols from the alphabet. Thus, within a maximal antirun of length ℓ_i, the choice of such a product reduces to a choice of permutation of weight at most 1 in S_{ℓ_i}, and there are F_{ℓ_i} ways of doing so. Since choosing this permutation is independent of the choice of permutations for the other maximal antiruns, the claim follows immediately. ∎

We are interested in finding the extreme cases of the size of radius-1 balls. Since for every $x \in \mathbb{Z}_q^n$, the sum of the entries in $\mathcal{P}(x)$ is also n, to find the maximum size of $|B_1(x)|$, we are interested in finding an integer partition of n, say $(\ell_1, \ell_2, \ldots, \ell_k)$, $\ell_i \geqslant 1$, $\sum_{i\in[k]} \ell_k = n$, such that $\prod_{i\in[k]} F_{\ell_i}$ is maximized. The following identity on Fibonacci numbers is well known,

$$F_{a+b} = F_a F_{b+1} + F_{a-1} F_b.$$

A simple rearrangement of this equation, also using the basic recursion, gives

$$F_{a+b} = 2F_a F_b + F_a F_{b-1} - F_{a-2} F_b > F_a F_b,$$

for all $a, b \geqslant 1$. The next corollary follows.

Corollary 17: The maximum size of a radius-1 ball is obtained when $x \in \mathbb{Z}_q^n$ is made of a single maximal antirun, and then

$$\max_{x\in\mathbb{Z}_q^n} |B_1(x)| = F_n.$$

There are exactly $q(q-1)^{n-1}$ such vectors $x \in \mathbb{Z}_q^n$.

Conversely, the minimum size of a ball is obtained when $x \in \mathbb{Z}_q^n$ is comprised of a single repeating symbol from $\mathbb{Z}_q$, so

$$\min_{x\in\mathbb{Z}_q^n} |B_1(x)| = 1,$$

and there are q such vectors $x \in \mathbb{Z}_q^n$.

The average size of balls is also of interest in code design and bounds. We define the average ball size as

$$\overline{B}_{r,q,n} = \frac{1}{q^n} \sum_{x\in\mathbb{Z}_q^n} |B_r(x)|.$$

The case of radius 1 is analyzed in the following theorem.

Theorem 18: For any $q, n \geqslant 2$,

$$\overline{B}_{1,q,n} = \overline{B}_{1,q,n-1} + \frac{q-1}{q}\overline{B}_{1,q,n-2},$$

with base cases $\overline{B}_{1,q,0} = \overline{B}_{1,q,1} = 1$. Explicitly,

$$\overline{B}_{1,q,n} = \left(\frac{\ell + \sqrt{\ell}}{2\ell}\right)\left(\frac{1+\sqrt{\ell}}{2}\right)^n + \left(\frac{\ell - \sqrt{\ell}}{2\ell}\right)\left(\frac{1-\sqrt{\ell}}{2}\right)^n,$$

where $\ell = 1 + \frac{4(q-1)}{q} = 5 - \frac{4}{q}$.

Proof: We recursively construct the collection of (unordered) pairs (x, y), with $x, y \in \Sigma^n$, such that $x \neq y$ and $y = \pi x$ for a permutation π with $\mathrm{wt}_\infty(\pi) \leqslant 1$. Denote this set by P_n.

First consider pairs $(x, y) \in P_n$ starting with the same symbol. Such pairs are obtained by concatenating any starting symbol $x_1 = y_1$ with a pair $(x^{[n-1]}, y^{[n-1]}) \in P_{n-1}$. The number of such (un-ordered) pairs is exactly $q\,|P_{n-1}|$.

Now consider pairs $(x, y) \in P_n$ starting with different symbols, $x_1 \neq y_1$. As $(x, y) \in P_n$, the second symbols of x and y must satisfy $x_2 = y_1$ and $y_2 = x_1$. The remaining symbols of (x, y) are obtained by $(x^{[n-2]}, y^{[n-2]}) \in P_{n-2}$ or by $x^{[n-2]} = y^{[n-2]}$. The number of (un-ordered) pairs $(x, y) \in P_n$ for which $(x^{[n-2]}, y^{[n-2]}) \in P_{n-2}$ is $q(q-1)\,|P_{n-2}|$: q options for x_1, $q-1$ options for $y_1 \neq x_1$ and for each x_1, y_1 there are $|P_{n-2}|$ possible values for $(x^{[n-2]}, y^{[n-2]})$. The number of (un-ordered) pairs $(x, y) \in P_n$ for which $x^{[n-2]} = y^{[n-2]} \in P_{n-2}$ is $q(q-1)q^{n-2}/2$: q options for x_1, $q-1$ options for $y_1 \neq x_1$ and for each x_1, y_1 there are q^{n-2} possible values for $x^{[n-2]} = y^{[n-2]}$, however, as we are counting un-ordered pairs so need to divide by 2.

All in all,

$$|P_n| = q\,|P_{n-1}| + q(q-1)\left(|P_{n-2}| + \frac{q^{n-2}}{2}\right).$$

By definition we have:

$$\overline{B}_{1,q,n} = \frac{1}{q^n} \sum_{x\in\Sigma^n} |B_r(x)| = \frac{2|P_n|}{q^n} + 1.$$

Implying that

$$|P_n| = \frac{q^n}{2}(\overline{B}_{1,q,n} - 1).$$

We can thus manipulate the recursive expression for P_n to obtain our assertion. ∎

It is interesting to note that for $|\Sigma| = q$, using Turán's Theorem [26], there exists a subset $\overline{C}$ of Σ^n of size $q^n/(\overline{B}_{1,q,n} + 1)$ such that for each $c, c' \in \overline{C}$, $c \neq c'$, it holds that $c' \notin B_r(c)$. In general, the size of $\overline{C}$ does not act as an upper bound on the maximum $(n, M; 1)_q$-LPC$_\infty$ code, as $\overline{C}$ is not necessarily a covering code. Nevertheless, in [6] it was observed that for many natural distance measures, the size of a corresponding $\overline{C}$ indeed happens to be larger than the size of any potential code for that distance measure. The question whether there is a natural distance measure for which $|\overline{C}|$ does not act as an upper bound for code construction was left open.

If we take as an example LPC$_\infty(1)$ over $\Sigma = \mathbb{Z}_2$, the asymptotic rate of $\overline{C}$ is

$$\lim_{n\to\infty} \frac{\log_2 |\overline{C}|}{n} = 1 - \log_2 \frac{1+\sqrt{3}}{2} \approx 0.55,$$

by Theorem 18. This is much smaller than the asymptotic rate of Example 24 (appearing in the upcoming section). Thus, it is interesting to note that LPC$_\infty(1)$ over $\Sigma = \mathbb{Z}_2$ is a natural example for which the size of $\overline{C}$ clearly does not act as an upper bound on the size of error-correcting codes

IV. Code Constructions

In this section we present two constructions of different flavor. The first is a direct construction, inspired by constrained-coding theory. In contrast, the second construction is recursive and requires seed codes.

A. Direct Construction

The direct construction we present focuses on the binary case. As we shall later see, the rate of any binary $(n, M; 1)$-LPC$_\infty$ is *asymptotically* upper bounded by $2/3$. Thus, we are interested in finding codes with rate as close as possible to this upper bound. Our approach to constructing such codes is motivated by a similar problem which was studied by Shamai and Zehavi [23] and later by Krachkovsky [14]. The problem studied in these works is an asymmetric version of the binary channel studied here. While in the binary model of the LPC$_\infty(r)$ channel, every bit can change its location by at most r positions, in the model studied in [14] and [23] this constraint is applied only to the bits having value 1. For example, for the word $x = 000111$, the ball of radius 1 under the LPC$_\infty(1)$ channel is $\{000111, 001011\}$, whereas in the asymmetric version of this channel, it is the set $\{000111, 001011, 001101, 001110\}$.

Let us fix for now $\Sigma = \mathbb{Z}_2 = \{0, 1\}$. The construction in [14] and [23] consists of the following idea. Given a set of blocks $\mathcal{B} \subseteq \Sigma^*$ (these blocks can be of any length), the code $C_n(\mathcal{B})$ is defined to be

$$C_n(\mathcal{B}) = \left\{ b_1 \dots b_m \;\middle|\; b_1, \dots, b_m \in \mathcal{B}, \sum_{i=1}^{m} |b_i| = n \right\}. \quad (5)$$

Under this construction it is possible to derive that the asymptotic rate $\log_2 \lambda$ of this code family, i.e.,

$$\limsup_{n\to\infty} \frac{\log_2 |C_n(\mathcal{B})|}{n} = \log_2 \lambda,$$

where λ is the largest (real) solution to the equation

$$\sum_{b\in\mathcal{B}} x^{-|b|} = 1.$$

The main goal of the works [14], [23] was to study the asymmetric version of the LPC$_\infty(r)$ channel for codes that additionally satisfy the *run-length limited (RLL)* constraint [8]. However, as a special case, one can eliminate the RLL constraint, whereupon the construction of [14] and [23] provides a set of blocks $\mathcal{B} = \left\{0^{3i}1 \mid i \geqslant 0\right\}$ that generates a code family with asymptotic rate ≈ 0.551. This family is optimal for the asymmetric version of the LPC$_\infty(r)$ channel without RLL constraints, and is also related to constrained-coding schemes studied in [20] and [29]. Since the error balls in the (symmetric) LPC$_\infty(r)$ channel are subsets of the error balls in the asymmetric LPC$_\infty(r)$ channel, we could also take the same set $\mathcal{B}$ as a solution for the (symmetric) LPC$_\infty(r)$ channel, and thus achieve at least the same rate. Next, we will show how to improve upon this construction and get an asymptotic rate of ≈ 0.5875.

Construction A: Define the block set $\mathcal{B} = \mathcal{B}_1 \cup \mathcal{B}_2 \cup \mathcal{B}_3 \cup \mathcal{B}_4$, where

$$\mathcal{B}_1 = \left\{ 0^{2+3i}1 \;\middle|\; i \geqslant 0 \right\}, \quad \mathcal{B}_2 = \left\{ 0^{3+3i}1^4 \;\middle|\; i \geqslant 0 \right\},$$
$$\mathcal{B}_3 = \left\{ 1^{2+3i}0 \;\middle|\; i \geqslant 0 \right\}, \quad \mathcal{B}_4 = \left\{ 1^{3+3i}0^4 \;\middle|\; i \geqslant 0 \right\}.$$

The constructed code is $C_n(\mathcal{B})$ as defined in (5). □

Theorem 19: For all $n \geqslant 3$, the code $C_n(\mathcal{B})$ from Construction A is an $(n, M; 1)_2$-LPC$_\infty$ code, and allows decoding in time $\Theta(n)$.

Proof: In order to show that $C_n(\mathcal{B})$ is an $(n, M; 1)_2$-LPC$_\infty$ code, we will explain how to uniquely decode it. Let $x \in C_n(\mathcal{B})$ be a transmitted word, and $y \in B_1(x)$ be the distorted received word.

We will scan the word y from left to right and show how to decode the first block. Once we know the identity of the first block, we also know whether the last symbol of the block exchanged places with the first symbol of the following block, and may correct it. We then proceed to remove the first block, and repeat the process.

First note that in every block from $\mathcal{B}$ the first bit y_1 cannot be in error. Thus we can already distinguish whether the first block belongs to either $\mathcal{B}_1 \cup \mathcal{B}_2$ or $\mathcal{B}_3 \cup \mathcal{B}_4$. Assume without loss of generality that $y_1 = 0$, and let us denote by ℓ the run length of 0's starting from y_1, so $\ell \geqslant 1$. We consider the following cases:

1) $\ell \equiv 1 \pmod 3$: This block can belong only to the set $\mathcal{B}_1$, so it is decoded as $0^{\ell+1}1$.
2) $\ell \equiv 2 \pmod 3$: This block can either belong to $\mathcal{B}_1$ or $\mathcal{B}_2$. If it belongs to the set $\mathcal{B}_2$ then the block is of the form $0^{\ell+1}1^4$ and an error has occurred in this block so we receive $0^\ell 1011$ as the first $\ell + 4$ bits.[1] If it

[1] Note that we do not examine the last bit of this block since it may be erroneous due to an interaction with the following block.

TABLE I
ASYMPTOTIC RATE FOR DIFFERENT VALUES OF r

r	λ_r	$\log_2(\lambda_r)$
2	1.2742	0.3496
3	1.1888	0.2495
4	1.1437	0.1937
5	1.1160	0.1583
6	1.0973	0.1340
7	1.0837	0.1160
8	1.0735	0.1023
9	1.0655	0.0915
10	1.0591	0.0828

belongs to $\mathcal{B}_1$, it is of the form $0^\ell 1$ and there is no error in this block.

If the next block starts with a 1, or strictly more than two 0's, we will receive that the following three bits are neither 011 nor 010, and we will be able to distinguish between $\mathcal{B}_1$ and $\mathcal{B}_2$.

Thus, the only remaining case we need to consider is the case in which the next block is 001. Then, the first $\ell+4$ bits can be one of the three options: $0^\ell 1001, 0^\ell 1010, 0^\ell 1000$, and in each case this sequence is different than $0^\ell 1011$, as required.

3) $\ell \equiv 3 \pmod 3$: This block can belong to either $\mathcal{B}_1$ or $\mathcal{B}_2$. If it belongs to $\mathcal{B}_1$ then it is of the form $0^{\ell-1}1$ and there was an error in this block so its following block starts with a 0, and thus the first $\ell+3$ bits can be $0^\ell 100, 0^\ell 101$, or $0^\ell 110$. If it belongs to $\mathcal{B}_2$ then the block is $0^\ell 1111$ and it has no error (in its first ℓ bits). Thus, the first $\ell+3$ bits are $0^\ell 111$, which are different than all other options in case the block belongs to $\mathcal{B}_1$.

Lastly, this proof provides a decoder with complexity $\Theta(n)$. ∎

The proof of the next corollary follows standard techniques (see [20]).

Corollary 20: The asymptotic rate of the code family $C_n(\mathcal{B})$ from Construction A is $\log_2 \lambda \approx 0.5875$, where λ is the largest solution of the equation $x^7 - 3x^4 - 2 = 0$.

Construction A can be generalized for arbitrary $(n, M; r)_2$-LPC$_\infty$ codes. This generalization uses the following block set $\mathcal{B}^r = \mathcal{B}_1^r \cup \mathcal{B}_2^r \cup \mathcal{B}_3^r \cup \mathcal{B}_4^r$, where

$$\mathcal{B}_1^r = \left\{ 0^{r+1+(2r+1)i} 1^r \,\middle|\, i \geqslant 0 \right\},$$
$$\mathcal{B}_2^r = \left\{ 0^{2r+1+(2r+1)i} 1^{3r+1} \,\middle|\, i \geqslant 0 \right\},$$
$$\mathcal{B}_3^r = \left\{ 1^{r+1+(2r+1)i} 0^r \,\middle|\, i \geqslant 0 \right\},$$
$$\mathcal{B}_4^r = \left\{ 1^{2r+1+(2r+1)i} 0^{3r+1} \,\middle|\, i \geqslant 0 \right\}.$$

The proof that the code $C_n(\mathcal{B}^r)$ is an $(n, M; r)_2$-LPC$_\infty$ code appears in Appendix VI. Similarly to Corollary 20, we also conclude that the asymptotic rate of the code family $C_n(\mathcal{B}^r)$ is $\log_2 \lambda_r$, where λ_r is the largest solution of the equation $x^{5r+2} - 3x^{3r+1} - 2 = 0$. The numerical values of these asymptotic rates are listed in Table I.

We also mention that in the other extreme case in which $q = |\Sigma|$ is large (w.r.t. r) we have codes with rate approaching 1. Such codes are reminiscent of network protocols that add meta-data to packets in order to correct packets that arrive out of order.

Construction B: Let $\ell = q/(2r+1)$ be an integer. Fix the alphabet $\Sigma = \mathbb{Z}_\ell \times \mathbb{Z}_{2r+1}$. The code we construct is

$$C_n = \left\{ x \in \Sigma^n \mid \forall i : x_i = (w_i, i \bmod (2r+1)), w_i \in \mathbb{Z}_\ell \right\}.$$

□

Intuitively, each symbol contains an "information" part, w_i, and a "meta-data" part, $i \bmod (2r+1)$. We show that this meta-data part allows us to correct permutations of weight at most r acting on the codeword.

Theorem 21: For $(2r+1)|q$, there exist $(n, M; r)_q$-LPC$_\infty$ codes with $M = \left(\frac{q}{2r+1}\right)^n$ (and thus rate $1 - \log_q(2r+1)$).

Proof: We use the codes of Construction B. Assume $x \in C_n$ was transmitted, but we received $y = \pi x$, with $\mathrm{wt}_\infty(\pi) \leqslant r$. Denote the elements of y as $y_i = (a_i, b_i) = (w_{\pi(i)}, \pi(i) \bmod (2r+1))$. Decoding $\hat{x}$ from y is done using the location information b_i. Specifically, to find $\hat{x}_j$ the decoder identifies the value of i closest to j for which $b_i = (j \bmod (2r+1))$, and sets $\hat{x}_j = y_i$.

For correctness, we contend $\hat{x} = x$. We show for i as defined above, that $j = \pi^{-1}(i)$. The fact that $b_i = (j \bmod (2r+1))$ implies that $\pi^{-1}(i) \equiv j \pmod{2r+1}$. Assume in contradiction that j is not $\pi^{-1}(i)$ but rather $\pi^{-1}(i')$. This implies that $|j - \pi^{-1}(i)| > 2r$, and in turn that $|j - i| > r$. On the other hand, as $j = \pi^{-1}(i')$ we have that $|j - i'| \leqslant r$. The discussion above contradicts the fact that i was chosen to be the closest to j for which $b_i = (j \bmod (2r+1))$. ∎

B. Recursive Construction

We present two recursive constructions that may be combined with seed codes either from a direct construction or from a computer search. The first construction is for a general alphabet $\Sigma = \mathbb{Z}_q$ and radius $r = 1$, whereas the second construction is for the binary alphabet $\Sigma = \mathbb{Z}_2$ and a general radius $r \geqslant 1$.

Before presenting the first construction we give the following definition. The *q-weight* function, $\mathrm{wt_q} : \mathbb{Z}_q^n \to \mathbb{Z}_q$ maps any vector $v = (v_1, v_2, \ldots, v_n)$ to $\mathrm{wt_q}(v) = \sum_{i=1}^n v_i$, where the summation is done in $\mathbb{Z}_q$.

Construction C: Assume that for each $a \in \mathbb{Z}_q$ we have a code $C_a \in \mathbb{Z}_q^n$ which is an $(n, M_a; 1)_q$-LPC$_\infty$ code, with the additional restriction that for each $c \in C_a$ we have $\mathrm{wt_q}(c) = a$. We call the C_a's the inner codes.

Additionally, let $C' \subseteq \mathbb{Z}_q^\ell$ be a set of vectors with distinct q-weight. We call C' the outer code.

The constructed code is

$$C = \bigcup_{(a_1, \ldots, a_\ell) \in C'} C_{a_1} \times \cdots \times C_{a_\ell}.$$

□

Theorem 22: Let $C_0, \ldots, C_{q-1}$, and C', be as in Construction C. Then the code C from Construction C is an $(n\ell, M; 1)_q$-LPC$_\infty$ code, with

$$M = \sum_{(a_1, \ldots, a_\ell) \in C'} \prod_{i=1}^{\ell} |C_{a_i}|.$$

Proof: The length and size of the code C are obvious. It remains to show its error-correction capabilities. Assume we receive a vector $y = (y_1, y_2, \ldots, y_{n\ell}) \in \mathbb{Z}_q^{n\ell}$, where $y \in B_1(c)$ for some $c \in C$. We will show a unique way of decoding y to prove the claim.

We first compute $\mathrm{wt_q}(y)$, which equals $\mathrm{wt_q}(c)$, since the channel does not change it. Since the words in the outer code have distinct q-weight, we obtain the unique words in C' with that q-weight, which we denote $(a_1, \ldots, a_\ell) \in C'$.

The next step in the decoding process comprises $\ell - 1$ iterations. Before we start we set $y^0 = y$. For the first iteration, we compute the q-weight of the first n-block of y^0, i.e.,

$$a'_1 = \sum_{i=1}^{n} y_i^0.$$

If $a'_1 \neq a_1$ then it must follow that the channel exchanged c_n with c_{n+1}, i.e., $y_n^0 = c_{n+1}$ and $y_{n+1}^0 = c_n$. Thus, if $a'_1 \neq a_1$ we exchange y_n^0 and y_{n+1}^0 to create a new vector we denote y^1. More precisely, if $a'_1 \neq a_1$ we set,

$$y_i^1 = \begin{cases} y_i^0 & i < n \text{ or } i > n+1, \\ y_{n+1}^0 & i = n, \\ y_n^0 & i = n+1, \end{cases}$$

and if $a'_1 = a_1$ we set $y^1 = y^0$.

Generally, in the jth iteration, $j \in [\ell - 1]$, we compute the q-weight of the jth block,

$$a'_j = \sum_{i=1}^{n} y_{(j-1)n+i}.$$

If $a'_j \neq a_j$ then we exchange the symbols y_{jn} with y_{jn+1}, i.e., we set

$$y_i^j = \begin{cases} y_i^{j-1} & i < jn \text{ or } i > jn+1, \\ y_{jn+1}^{j-1} & i = jn, \\ y_{jn}^{j-1} & i = jn+1, \end{cases}$$

and if $a'_j = a_j$ we set $y^j = y^{j-1}$.

To complete the decoding process, it remains to determine the order of elements within each block. Since C_{a_j} is an $(n; 1)_q$-LPC_∞ code as well, by recursion there is a unique way of decoding each of the ℓ blocks of $y^{\ell-1}$ to obtain the desired c. ∎

Example 23: Consider the binary $(3, 4; 1)_2$-LPC_∞ *code* $\{000, 100, 110, 111\}$. *We can partition it by q-weights to obtain* $C_0 = \{000, 110\}$ *and* $C_1 = \{100, 111\}$. *Using Construction C we can create a family of* LPC_∞ *codes with parameters* $(3m, 2 \cdot 2^m; 1)_2$ *for all* $m \geqslant 2$. *This code family has an asymptotic rate of* $\frac{1}{3}$. □

Example 24: We can construct codes using a greedy computer search in the following manner. Fix an alphabet, in this case, $\Sigma = \mathbb{Z}_2$. *Set a length n, and write a lexicographic list of all the length-n vectors over* Σ. *Start with an empty set* C^0. *At the ith iteration,* $i = 1, 2, \ldots$, *find the lexicographically-least vector* $c \in \Sigma^n \setminus C^{i-1}$ *such that* $C^{i-1} \cup \{c, \overline{c}\}$ *is still an* $(n; 1)_2$-LPC_∞ *code, where* $\overline{c}$ *denotes the bit-wise complement of c.*

Using such a procedure, for length $n = 24$ *a computer search resulted in an* LPC_∞*-code C with parameters* $(24, 50220; 1)_2$. *This code has rate* ≈ 0.650667. *The code C has* 25122 *codewords of even weight, and* 25098 *codewords of odd weight. Using Construction C we can create a family of* LPC_∞*-codes with parameters* $(24m, 50220 \cdot 25122^{m-1}; 1)_2$ *for all* $m \geqslant 2$. *This code family has an asymptotic rate of* ≈ 0.609028 *(which is the highest asymptotic rate of a family of binary codes presented in this work).* □

We now turn to describe the second construction, which addresses the binary case $\Sigma = \mathbb{Z}_2$ and arbitrary radius $r \geqslant 1$.

Construction D: Fix $\Sigma = \mathbb{Z}_2$ *and some* $r \geqslant 1$. *Let* C' *be an* $(n, M; r)_2$-LPC_∞ *code with* $n \geqslant r$. *We also require the extra property that there exists* $a \in \mathbb{Z}_{r+1}$ *such that for every* $c' \in C'$, $\mathrm{wt}(c') \equiv a \pmod{r+1}$. *Here* $\mathrm{wt}(c')$ *denotes the regular weight function, that counts the number of non-zero entries in the vector* c'.

The constructed code is

$$C = \underbrace{C' \times C' \times \cdots \times C'}_{\ell \text{ times}}.$$

□

Theorem 25: Let C' *be as in Construction D. Then the code C from Construction D is an* $(n\ell, M^\ell; r)_q$-LPC_∞.

Proof: The length and size of the code are obvious from the construction. It remains to show that it can correct distortions by permutations of weight at most r. This is done in a similar fashion to the proof of Theorem 22, though with slightly different arguments.

Denote the received word as $y = (y_1, y_2, \ldots, y_{n\ell}) \in \mathbb{Z}_2^{n\ell}$. We focus on the first two blocks of length n, i.e., $y^1 = (y_1, \ldots, y_n)$ and $y^2 = (y_{n+1}, \ldots, y_{2n})$. We similarly denote the transmitted word by $c = (c_1, c_2, \ldots, c_{n\ell})$ and its first two blocks by $c^1 = (c_1, \ldots, c_n)$ and $c^2 = (c_{n+1}, \ldots, c_{2n})$.

All the bits of y^1 originate from those of c^1 except perhaps bits appearing in the last r positions of y^1, which originate from the first r positions in c^2. Let us denote $w = \mathrm{wt}(y^1) \bmod (r+1)$. If $w \neq a$, where a is the weight (modulo $r+1$) of all the codewords in C', then there is a unique number $t \in [r]$, and a bit $b \in \mathbb{Z}_2$, such that we can exchange the last t occurrences of b from the last r positions of y^1 with the first t occurrences of $\overline{b}$ from the first r positions of y^2, and the resulting first block, called z^1, now has $\mathrm{wt}(z^1) \equiv a \pmod{r+1}$. Here $\overline{b}$ denotes the binary complement of b.

It is easy to verify the new first block z^1, has the exact same weight as c^1. Furthermore, while the bits returned from the second block, y^2, may not be in their correct position, they are certainly now no more than r positions away from their original position in c^1. Thus, the first block, z^1, may be corrected using the inner code C'. The process now continues with the next block iteratively. ∎

It follows that Construction D allows us to extend a seed code to an infinite family of codes with the same error-correction capability and the exact same rate.

Example 26: Using a computer search, a lexicographic search procedure was performed. Starting with a list of the entire space of vectors, at each iteration the

lexicographically-least word in the list was added to the code, and the ball of radius r around it was removed from the list.

Thus, for length $n = 18$, radius $r = 2$, and parameter $a = 1$, a computer search found a code C' with 172 codewords, all of whose weights leave a residue of $a = 1$ modulo $r + 1 = 3$. By Construction D, for all $m \geqslant 1$ there exists an $(18m, 172^m; 2)_2$-LPC$_\infty$ code, giving a family with rate ≈ 0.41257.

The same procedure, for length $n = 17$, radius $r = 3$, and parameter $a = 3$, found a code C' with 43 codewords, all of whose weights leave a residue of $a = 3$ modulo $r + 1 = 4$. By Construction D, for all $m \geqslant 1$ there exists a $(17m, 43^m; 3)_2$-LPC$_\infty$ code, giving a family with rate ≈ 0.31919. Note that these rates are higher than the ones achieved by the direct construction in Section IV-A and are summarized in Table I. □

V. Bounds on Code Parameters

We now present a number of upper bounds on code size. This first theorem shows a connection between $A_q(n; r)$ and $K_q(n; r)$. While this connection is well-known in other settings, the usual techniques of proving it do not work here since the size of balls depends on their center. Nevertheless, the proof is elementary.

Theorem 27: For all n and r,

$$A_q(n; r) \leqslant K_q(n; r).$$

Proof: Assume to the contrary there exist n and r such that $A_q(n; r) > K_q(n; r)$. Let C_1 be such an $(n, A_q(n; r); r)_q$-LPC$_\infty$ code, and let C_2 be such an $(n, K_q(n; r))_q r$-LPC$_\infty$ covering code, both over $\Sigma = \mathbb{Z}_q$. We know that the $K_q(n; r)$ balls of radius r centered around the codewords of C_2, cover the entire space Σ^n. Since $A_q(n; r) > K_q(n; r)$, the average number of codewords of C_1 per ball around a codeword of C_2 is strictly greater than 1. Thus, there exist two codewords $c_1, c_1' \in C_1$ and a codeword $c_2 \in C_2$ such that $c_1, c_1' \in B_r(c_2)$. But that means $c_2 \in B_r(c_1) \cap B_r(c_1')$, contradicting the fact that C_1 is an $(n; r)_q$-LPC$_\infty$ code. ■

This simple argument implies the following general bound.

Theorem 28: Let $\Sigma = \mathbb{Z}_q$ be the alphabet, $q \geqslant 2$. Then for all $n \geqslant r \geqslant 1$,

$$A_q(n; r) \leqslant K_q(n; r) \leqslant \binom{r+q}{q-1}^{\lceil n/(r+1)\rceil}.$$

Proof: Construct the following code $C \subseteq \mathbb{Z}_q^n$: a word $c = (c_1, c_2, \ldots, c_n) \in \mathbb{Z}_q^n$ is in C if and only if, for every $0 \leqslant j < \lceil n/(r+1)\rceil$ we have

$$c_{(r+1)j+1} \leqslant c_{(r+1)j+2} \leqslant \cdots \leqslant c_{(r+1)j+r+1}.$$

To avoid problems due to divisibility, we assume that for all $i > n$, $c_i = \infty$.

A simple counting argument shows that

$$|C| \leqslant \binom{r+q}{q-1}^{\lceil n/(r+1)\rceil},$$

with equality if $(r + 1) \mid n$.

Finally, we contend C is an $(n, |C|)_q r$-LPC$_\infty$ covering code. Indeed, for any vector $v \in \mathbb{Z}_q^n$ we can sort, in non-descending order, positions $(r + 1)j + 1$ through $(r + 1)j + r + 1$ for all $0 \leqslant j < \lceil n/(r + 1)\rceil$, to obtain a codeword in C. We note that the permutation induced by this sorting operation does not move an element more than r positions from its original position, and so, the permutation has weight at most r. ■

For $r = 1$ we may obtain improved upper-bounds (which are tight for $n = 3$).

Theorem 29: Let $\Sigma = \mathbb{Z}_q$ be the alphabet, $q \geqslant 2$, and $r = 1$. Then for all $3|n$,

$$A_q(n; r) \leqslant K_q(n; r) \leqslant \left[q + 2\binom{q}{2} + 2\binom{q}{3}\right]^{n/3}.$$

Here, for $q = 2$ we have $\binom{q}{3} = 0$.

Proof: To prove our assertion, we present a $(3, M)_q$ 1-LPC$_\infty$ covering code C of size $M = q + 2\binom{q}{2} + 2\binom{q}{3}$ for block length $n = 3$. The theorem follows from considering the covering code $C^{n/3}$. The code C is the union of the following three sets of codewords, $C_1 \cup C_2 \cup C_3$. The set C_1 includes, for any character $c \in \Sigma$, the codeword (c, c, c). The set C_2 includes for any pair $c_1 \neq c_2$ in Σ, the codeword (c_1, c_2, c_1). The set C_3 includes for any three distinct elements of Σ, $c_1 \leqslant c_2 \leqslant c_3$, the codewords (c_1, c_2, c_3) and (c_3, c_2, c_1).

The size of C is exactly $M = q + 2\binom{q}{2} + 2\binom{q}{3}$. We show that C is indeed a covering code. Consider any $(c_1, c_2, c_3) \in \Sigma^3$. If $c_1 = c_2 = c_3$ then $(c_1, c_2, c_3) \in C_1$. If there are two distinct elements in the set $\{c_1, c_2, c_3\}$ then it is not hard to verify that (c_1, c_2, c_3) is covered by an element of C_2. If there are three distinct elements in the set $\{c_1, c_2, c_3\}$ then it is not hard to verify that (c_1, c_2, c_3) is covered by an element of C_3. We note that the set C is also a $(3, M; 1)_q$-LPC$_\infty$ code. ■

In particular, for the binary case the last theorem provides an *asymptotic* upper bound of $2/3$ on the rate of $(n; 1)$-LPC$_\infty$ codes.

Thus far, we focused in this section and the previous one on codes with a constant error-correction capability. This is motivated by the next corollary that shows that all other cases have asymptotic rate 0.

Corollary 30: Let $\Sigma = \mathbb{Z}_q$ be the alphabet, $q \geqslant 2$ a constant. Let $\{C_i\}_{i \geqslant 1}$ be a sequence of codes, C_i being an $(n_i, M_i; r_i)_q$-LPC$_\infty$ code, and $n_{i+1} > n_i$ for all $i \in \mathbb{N}$. If $r_i = \omega(1)$, i.e., $\limsup_{i\to\infty} r_i = \infty$, then the asymptotic rate of the family is

$$\limsup_{i\to\infty} \frac{\log_q M_i}{n_i} = 0.$$

Proof: By Theorem 28,

$$\begin{aligned}\frac{1}{n_i}\log_q M_i &\leqslant \frac{1}{n_i}\log_q \binom{r_i+q}{q-1}^{\lceil n_i/(r_i+1)\rceil} \\ &\leqslant \frac{n_i + r_i}{n_i r_i}\log_q (r_i + q)^{q-1} \\ &\leqslant \frac{2(q-1)}{r_i}\log_q (r_i + q) \\ &= O\left(\frac{\log r_i}{r_i}\right),\end{aligned}$$

which completes the proof. ■

TABLE II
LOWER AND UPPER BOUNDS ON $A_2(n; 1)$

n	Lower Bound	Upper Bound
3	4	4
4	8	8
5	12	12
6	16	16
7	28	30
8	42	46
9	64	64
10	104	116
11	157	178
12	246	256
13	388	450
14	594	696
15	930	1024
16	1454	1750

We conclude this section by explaining how to apply the methodology from [15] in order to get an upper bound on $A_2(n; 1)$. Kulkarni and Kiyavash proposed [15] a method to apply a modified version of the ball-packing bound for the deletion channel. Since their method can be applied not only for the deletion channel but also for other irregular channels, several follow-up works have studied other channels such as grain errors [7], [9] and multipermutations with Kendall's τ-metric [1]. More general results were derived also in [3] and in [6] with more specific cases of irregular channels such as asymmetric errors and projective spaces.

The main idea in deriving such a bound is to construct a hypergraph whose set of vertices consists of all possible received words. Its hyperedges are given by the error balls of all possible transmitted words. Then, an error-correcting code in such a channel corresponds to a set of disjoint hyperedges, i.e., a *matching*. Thus, an upper bound on the code size can be given by an upper bound on the size of the largest matching in the hypergraph. A *transversal* in a hypergraph is a set of vertices which intersects every hyperedge in the hypergraph, and a fractional transversal is an assignment of weights to the vertices such that the sum of weights of the vertices in every hyperedge is at least 1. The main idea in [15] was to use fractional transversals as an upper bound on the size of the largest matching in the hypergraph, and thus, on the size of a code.

We use this method in order to derive such an upper bound. The composition of a vector is the ordered list of the number of times each symbol from the alphabet appears. Since vectors of different composition have corresponding hyperedges that do not intersect, we may study different compositions separately.

In particular, we focus on radius $r = 1$ and the binary case in which the composition is specified by a single integer, i.e., the weight of the vector. For each weight we solve a linear program that calculates the optimal fractional transversal with minimum sum weights in order to get an upper bound on each code. Table II summarizes the upper bound we calculated on $A_2(n; 1)$ for $3 \leqslant n \leqslant 16$, together with the lower bound implied by the best codes we could find by computer search.

VI. CONCLUSION

In this work we initiated the study of ℓ_∞-limited permutation channels $\mathrm{LPC}_\infty(r)$ for worst-case errors and general alphabets Σ. We presented code constructions and upper bounds on code size. The majority of our results are for the case of $r = 1$. Despite significant efforts, our upper and lower bounds on code size are not tight and should be viewed as initial steps in a full understanding of $\mathrm{LPC}_\infty(r)$. For the case of binary codes with $r = 1$ we conjecture that the optimal asymptotic rate is $2/3$. This agrees with our upper bounds and our simulations up to block length $n = 24$. The optimal rate of codes for $\mathrm{LPC}_\infty(r)$ it left open and subject to future research.

APPENDIX

In this appendix we prove that the code $C_n(\mathcal{B}^r)$ is an $(n, M; r)_2$-LPC_∞ code, where $\mathcal{B}^r$ is the union of the four sets:

$$\mathcal{B}_1^r = \left\{ 0^{r+1+(2r+1)i} 1^r \,\middle|\, i \geqslant 0 \right\},$$
$$\mathcal{B}_2^r = \left\{ 0^{2r+1+(2r+1)i} 1^{3r+1} \,\middle|\, i \geqslant 0 \right\},$$
$$\mathcal{B}_3^r = \left\{ 1^{r+1+(2r+1)i} 0^r \,\middle|\, i \geqslant 0 \right\},$$
$$\mathcal{B}_4^r = \left\{ 1^{2r+1+(2r+1)i} 0^{3r+1} \,\middle|\, i \geqslant 0 \right\}.$$

Theorem 31: For all $n \geqslant 2r + 1$, the code $C_n(\mathcal{B}^r)$ is an $(n, M; r)_2$-LPC_∞ code, and allows decoding in time $\Theta(n)$.

Proof: We follow the proof of Theorem 19. Assume that $x \in C_n(\mathcal{B}^r)$ was transmitted, and the word $y \in B_r(x)$ was received. We only show how to decode the first block in y as the consecutive blocks are decoded in the same way.

The first bit of the first block cannot change its value and thus we can easily determine whether the block belongs to $\mathcal{B}_1^r \cup \mathcal{B}_2^r$ or $\mathcal{B}_3^r \cup \mathcal{B}_4^r$. Assume without loss of generality that it belongs to $\mathcal{B}_1^r \cup \mathcal{B}_2^r$ so $y_1 = 0$. Let $\ell_x, \ell_y \geqslant 1$ be the length of the first run of zeroes in x, y, respectively.

Note that if $x \in \mathcal{B}_2^r$ then $\ell_x \pmod{2r+1} = 2r + 1$ and $r + 1 \leqslant \ell_y \pmod{2r+1} \leqslant 2r + 1$, since the first sequence of zeros is followed by $3r+1$ ones.[2] On the other hand, if $x \in \mathcal{B}_1^r$ then $\ell_x \pmod{2r+1} = r + 1$, and $1 \leqslant \ell_y \pmod{2r+1} \leqslant 2r + 1$.

We consider the following cases:

1) $\ell_y \pmod{2r+1} \in \{1, 2, \ldots, r\}$: The first decoded block belongs to the set $\mathcal{B}_1^r$, so it is decoded as $0^{r+1+(2r+1)i} 1^r$, where $i = \lfloor \frac{\ell_y}{2r+1} \rfloor$.
2) $\ell_y \pmod{2r+1} = r + 1$: Assume that for some $i \geqslant 0$, $\ell_y = r+1+(2r+1)i$. If the first block in x belongs to $\mathcal{B}_2^r$ then it is the sequence $0^{2r+1+(2r+1)i} 1^{3r+1}$ and the last r zeros among the first ℓ_x zeros change their position with some of the following one bits, so we get after this transposition the sequence $0^{r+1+(2r+1)i} 1^b 0 u$, where $1 \leqslant b \leqslant r$, and u is a sequence including the remainder of the bits of this block. We can also claim that among the first $2r+1+(2r+1)i+2r+1 = (2r+1)i+4r+2$ bits of y there are exactly $(2r+1)i+2r+1$ zeros. Note that we did not consider the last r bits of the this block as they can change their value due to an interaction with the following block.

[2]Here, we take the residues modulo $2r + 1$ to be the set $\{1, \ldots, 2r + 1\}$.

If the first block in x belongs to $\mathcal{B}_1^r$, then it is the sequence $0^{r+1+(2r+1)i}1^r$ and the first ℓ_x zeros do not change their position. If the next block starts with one, it has to be at least $r+1$ ones, and then after the sequence of $r+1+(2r+1)i$ zeros there are at least $r+1$ consecutive ones in y. As the previously defined b is at most r, we will be able to distinguish between this case and the case in which x belongs to $\mathcal{B}_2^r$.

If the next block starts with zero, then it has to be at least $r+1$ zeros and then among the first $r+1+(2r+1)i+r+r+1+r=(2r+1)i+4r+2$ bits of y there are at least $(2r+1)i+2r+2$ zeros, so again we can distinguish between this case and the case in which x belongs to $\mathcal{B}_2^r$.

3) $\ell_y \pmod{2r+1} = r+1+a$ for $1 \leqslant a \leqslant r$: Assume that for some $i \geqslant 0$, $\ell_y = r+1+a+(2r+1)i$. If the first block in x belongs to $\mathcal{B}_2^r$ then it is the sequence $0^{2r+1+(2r+1)i}1^{3r+1}$, where the last $r-a$ zeros among the first ℓ_x zeros change their position with the following one bits. We can again claim that among the first $2r+1+(2r+1)i+2r+1=(2r+1)i+4r+2$ bits of y there are exactly $(2r+1)i+2r+1$ zeros.

If the first block in x belongs to $\mathcal{B}_1^r$, then it is the sequence $0^{r+1+(2r+1)i}1^r$. In order to start with $r+1+a+(2r+1)i$ zeros, the next block has to start with a zero and thus it starts with at least $r+1$ zeros. However, then among the first $r+1+(2r+1)i+r+r+1+r=(2r+1)i+4r+2$ bits of y there are at least $(2r+1)i+2r+2$ zeros, so we can again distinguish between the two cases.

We conclude that in all cases it is possible to distinguish which set the first block belongs to and thus it can be decoded successfully. Furthermore, this decoder will have complexity $\Theta(n)$, when r is fixed. ■

REFERENCES

[1] S. Buzaglo, E. Yaakobi, T. Etzion, and J. Bruck, "Error-correcting codes for multipermutations," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Istanbul, Turkey, Jul. 2013, pp. 724–728.

[2] G. Cohen, I. Honkala, S. Litsyn, and A. Lobstein, *Covering Codes*. Amsterdam, The Netherlands: North Holland, 1997.

[3] D. Cullina and N. Kiyavash, "Generalized sphere-packing and sphere-covering bounds on the size of codes for combinatorial channels," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Honolulu, HI, USA, Jun. 2014, pp. 1266–1270.

[4] M. Deza and T. Huang, "Metrics on permutations, a survey," *J. Combinat., Inf. Syst. Sci.*, vol. 23, pp. 173–185, 1998.

[5] F. F. Hassanzadeh, M. Schwartz, and J. Bruck, "Bounds for permutation rate-distortion," *IEEE Trans. Inf. Theory*, vol. 62, no. 2, pp. 703–712, Feb. 2016.

[6] A. Fazeli, A. Vardy, and E. Yaakobi, "Generalized sphere packing bound," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2313–2334, May 2015.

[7] R. Gabrys, E. Yaakobi, and L. Dolecek, "Correcting grain-errors in magnetic media," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2256–2272, May 2015.

[8] K. A. S. Immink, *Coding Techniques for Digital Recorders*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1991.

[9] N. Kashyap and G. Zémor, "Upper bounds on the size of grain-correcting codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 8, pp. 4699–4709, Aug. 2014.

[10] T. Kløve, "Lower bounds on the size of spheres of permutations under the Chebychev distance," *Des., Codes Cryptogr.*, vol. 59, nos. 1–3, pp. 183–191, 2011.

[11] T. Kløve, T.-T. Lin, S.-C. Tsai, and W.-G. Tzeng, "Permutation arrays under the Chebyshev distance," *IEEE Trans. Inf. Theory*, vol. 56, no. 6, pp. 2611–2617, Jun. 2010.

[12] M. Kovačević and P. Popovski, "Zero-error capacity of a class of timing channels," *IEEE Trans. Inf. Theory*, vol. 60, no. 11, pp. 6796–6800, Nov. 2014.

[13] M. Kovačević and D. Vukobratović. (Jan. 2013). "Multiset code for permutation channels." [Online]. Available: https://arxiv.org/abs/1301.7564

[14] V. Y. Krachkovsky, "Bounds on the zero-error capacity of the input-constrained bit-shift channel," *IEEE Trans. Inf. Theory*, vol. 40, no. 4, pp. 1240–1244, Jul. 1994.

[15] A. A. Kulkarni and N. Kiyavash, "Nonasymptotic upper bounds for deletion correcting codes," *IEEE Trans. Inf. Theory*, vol. 59, no. 8, pp. 5115–5130, Aug. 2013.

[16] M. Langberg, M. Schwartz, and E. Yaakobi, "Coding for the ℓ_∞-limited permutation channel," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Hong Kong, Jun. 2015, pp. 1936–1940.

[17] D. H. Lehmer, "Permutations with strongly restricted displacements," in *Combinatorial Theory and its Applications II*, P. Erdös, A. Rényi, and V. T. Sós, Eds. Amsterdam, The Netherlands: North Holland, 1970, pp. 273–291.

[18] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. Amsterdam, The Netherlands: North Holland, 1978.

[19] H. H. Permuter, P. Cuff, B. Van Roy, and T. Weissman, "Capacity of the trapdoor channel with feedback," *IEEE Trans. Inf. Theory*, vol. 54, no. 7, pp. 3150–3165, Jul. 2008.

[20] M. Schwartz and J. Bruck, "On the capacity of the precision-resolution system," *IEEE Trans. Inf. Theory*, vol. 56, no. 3, pp. 1028–1037, Mar. 2010.

[21] M. Schwartz, "Efficiently computing the permanent and Hafnian of some banded Toeplitz matrices," *Linear Algebra Appl.*, vol. 430, no. 4, pp. 1364–1374, Feb. 2009.

[22] M. Schwartz and I. Tamo, "Optimal permutation anticodes with the infinity norm via permanents of (0, 1)-matrices," *J. Combinat. Theory A*, vol. 118, no. 6, pp. 1761–1774, Aug. 2011.

[23] S. Shamai (Shitz) and E. Zehavi, "Bounds on the capacity of the bit-shift magnetic recording channel," *IEEE Trans. Inf. Theory*, vol. 37, no. 3, pp. 863–872, May 1991.

[24] I. Tamo and M. Schwartz, "Correcting limited-magnitude errors in the rank-modulation scheme," *IEEE Trans. Inf. Theory*, vol. 56, no. 6, pp. 2551–2560, Jun. 2010.

[25] I. Tamo and M. Schwartz, "On the labeling problem of permutation group codes under the infinity metric," *IEEE Trans. Inf. Theory*, vol. 58, no. 10, pp. 6595–6604, Oct. 2012.

[26] P. Turán, "On the theory of graphs," *Colloq. Math*, vol. 3, no. 1, pp. 19–30, 1954.

[27] J. M. Walsh and S. Weber, "Capacity region of the permutation channel," in *Proc. 46th Annu. Allerton Conf. Commun., Control, Comput.*, Monticello, IL, USA, Sep. 2008, pp. 646–652.

[28] D. Wang, A. Mazumdar, and G. W. Wornell, "Compression in the space of permutations," *IEEE Trans. Inf. Theory*, vol. 61, no. 12, pp. 6417–6431, Dec. 2015.

[29] R. W. Yeung, N. Cai, S. W. Ho, and A. B. Wagner, "Reliable communication in the absence of a common clock," *IEEE Trans. Inf. Theory*, vol. 55, no. 2, pp. 700–712, Feb. 2009.

[30] H. Zhou, M. Schwartz, A. A. Jiang, and J. Bruck, "Systematic error-correcting codes for rank modulation," *IEEE Trans. Inf. Theory*, vol. 61, no. 1, pp. 17–32, Jan. 2015.

Michael Langberg (M'07–SM'15) received his B.Sc. in mathematics and computer science from Tel-Aviv University in 1996, and his M.Sc. and Ph.D. in computer science from the Weizmann Institute of Science in 1998 and 2003 respectively. Between 2003 and 2006, he was a postdoctoral scholar in the Electrical Engineering and Computer Science departments at the California Institute of Technology, and between 2007 and 2012 he was in the Department of Mathematics and Computer Science at The Open University of Israel. Prof. Langberg is currently an associate professor in the Department of Electrical Engineering at the State University of New York at Buffalo.

Prof. Langberg's research addresses the algorithmic and combinatorial aspects of information in communication, management, and storage; focusing on the study of information theory, coding theory, network communication and network coding, big data in the form of succinct data representation, and probabilistic methods in combinatorics. Prof. Langberg was an Associate Editor for the IEEE TRANSACTIONS ON INFORMATION THEORY during the years 2012-2015 and is currently the Editor of the IEEE INFORMATION THEORY SOCIETY NEWSLETTER.

Moshe Schwartz (M'03–SM'10) is an associate professor at the Department of Electrical and Computer Engineering, Ben-Gurion University of the Negev, Israel. His research interests include algebraic coding, combinatorial structures, and digital sequences.

Prof. Schwartz received the B.A. (*summa cum laude*), M.Sc., and Ph.D. degrees from the Technion – Israel Institute of Technology, Haifa, Israel, in 1997, 1998, and 2004 respectively, all from the Computer Science Department. He was a Fulbright post-doctoral researcher in the Department of Electrical and Computer Engineering, University of California San Diego, and a post-doctoral researcher in the Department of Electrical Engineering, California Institute of Technology. While on sabbatical 2012–2014, he was a visiting scientist at the Massachusetts Institute of Technology (MIT).

Prof. Schwartz received the 2009 IEEE Communications Society Best Paper Award in Signal Processing and Coding for Data Storage.

Eitan Yaakobi (S'07–M'12–SM'17) is an Assistant Professor at the Computer Science Department at the Technion — Israel Institute of Technology. He received the B.A. degrees in computer science and mathematics, and the M.Sc. degree in computer science from the Technion — Israel Institute of Technology, Haifa, Israel, in 2005 and 2007, respectively, and the Ph.D. degree in electrical engineering from the University of California, San Diego, in 2011. Between 2011-2013, he was a postdoctoral researcher in the department of Electrical Engineering at the California Institute of Technology. His research interests include information and coding theory with applications to non-volatile memories, associative memories, data storage and retrieval, and voting theory. He received the Marconi Society Young Scholar in 2009 and the Intel Ph.D. Fellowship in 2010-2011.