Using Pseudocodewords to Transmit Information

Nathan Axvig

arXiv:1810.08086v1 [cs.IT] 18 Oct 2018

Abstract

The linear programming decoder will occasionally output fractional-valued sequences that do not correspond to binary codewords – such outputs are termed *nontrivial pseudocodewords*. Feldman et al. have demonstrated that it is precisely the presence of nontrivial pseudocodewords that prevents the linear programming decoder from attaining maximum-likelihood performance. The purpose of this paper is to cast a positive light onto these nontrivial pseudocodewords by turning them into beneficial additions to our codebooks. Specifically, we develop a new modulation scheme, termed *insphere modulation*, that is capable of reliably transmitting both codewords and pseudocodewords. The resulting non-binary, non-linear codebooks have higher spectral efficiencies than the binary linear codes from which they are derived and in some cases have the same or slightly better block error rates. In deriving our new modulation scheme we present an algorithm capable of computing the *insphere* of a polyhedral cone – which loosely speaking is the largest sphere contained within the cone. This result may be of independent mathematical interest.

I. Introduction

When simulating a block code, one often repeats the following procedure: generate a codeword, transmit it over a noisy channel, estimate a codeword that best explains the received vector, and finally compare the estimated codeword to the original. When the estimated codeword and the transmitted codeword are identical, we record an overall success for the coding scheme. In the case where the estimated codeword is not equal to the original codeword, we declare a block error. This paper explores a third option: what if the estimated codeword is not a codeword at all? What, if anything, can be gained from this?

The third option above is a very real possibility when using certain "modern" decoding algorithms such as min-sum and linear programming decoding. It has been observed (see, e.g., [2], [6]) that occasionally these decoders will output sequences that are not codewords at all – such non-codeword outputs are termed *nontrivial pseudocodewords*. For an example, consider transmission of a codeword from the binary linear code given by the null space of the following parity-check matrix over the additive white Gaussian noise (AWGN) channel:

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

If the received vector yields the vector of log-likelihood ratios $\left[0,0,0,-1,1,1,0,0,0\right]^T$, then the linear programming (LP) decoder will return $\left[\frac{1}{2},\frac{1}{2},\frac{1}{2},1,0,0,\frac{1}{2},\frac{1}{2},\frac{1}{2}\right]^T$ as an output. This output vector is clearly not a binary codeword since it has fractional values. In such a case Feldman et al. [6] suggest that we declare a block error and move on with our lives.

We henceforth adopt the mindset that any potential output of a decoder can be used to transmit information across a channel. To this end we construct a new modulation scheme, termed *insphere*

N. Axvig is with the Department of Mathematics, Concordia College, Moorhead, MN, 56562, USA (email: ndaxvig@cord.edu).

This paper was presented in part at the 2011 Fall Central Section Meeting of the American Mathematical Society (Lincoln, NE; October 2011) and at the 2012 Joint Mathematics Meetings (Boston, MA; January 2012).

This work has been submitted to the IEEE for possible publication. Copyright may be transferred without notice, after which this version may no longer be accessible.

modulation, that is capable of reliably transmitting pseudocodewords in conjunction with codewords. Employing insphere modulation therefore allows for larger codebooks of the same length as the original binary code, and as a result we see modest gains in both spectral efficiency and information rate. Moreover, we demonstrate that in some circumstances these gains in throughput can be made without sacrificing error-correcting performance. In order to compute inspheres for transmitting pseudocodewords, we develop an algorithm capable of computing the insphere of a polyhedral cone to any degree of accuracy. While we presently treat this result as a means to an end, it may be of independent mathematical interest.

The remainder of this paper is organized as follows. In Section II we review linear programming decoding and some key results from linear optimization. Section III is dedicated to determining which vectors are best suited to transmitting linear programming pseudocodewords across the AWGN channel, and Section IV derives an algorithm for computing the best of these vectors, which in our case turns out to be the center of the insphere of an appropriately defined polyhedral cone. Section V shows how the results of Section III can be expanded to greatly increase the size of our codebook. Section VI gives a detailed example, Section VII contains simulation results, and we conclude in Section VIII.

II. Background

To produce vectors suitable for reliably transmitting pseudocodewords, we must go "under the hood" of the linear programming decoder and its underlying algebra and geometry. The material in this section is largely adapted from [4], though it can also be found in most introductory texts on linear optimization. Notationally, we write $\mathbf{x}^T$ for the transpose of $\mathbf{x}$ and $\|\mathbf{x}\|$ for the usual Euclidean norm of $\mathbf{x}$. Unless stated otherwise, all vectors are assumed to be column vectors.

A. Background on Linear Programming

A *polyhedron* in $\mathbb{R}^n$ is the set of all points satisfying a finite set of linear equality and inequality constraints. Since an equality constraint of the form $\mathbf{a}^T\mathbf{x} = b$ can be replaced by the two inequalities $\mathbf{a}^T\mathbf{x} \leq b$ and $\mathbf{a}^T\mathbf{x} \geq b$, we can equivalently define a polyhedron as the set of all points satisfying a finite number of linear inequality constraints. Moreover, by multiplying constraints by -1 we may reformulate any description of a polyhedron in various compact forms involving matrix products such as $A_1\mathbf{x} \leq \mathbf{b}_1$ or $A_2\mathbf{x} \geq \mathbf{b}_2$.

If a polyhedron is bounded we refer to it as a *polytope*. On the other hand, if a polyhedron is closed under multiplication by non-negative scalars we call it a *polyhedral cone*. It is an easy exercise to see that polyhedra are convex sets, i.e., given two points $\mathbf{x}$ and $\mathbf{y}$ in a polyhedron $\mathcal{M}$ and a scalar $\lambda \in [0, 1]$, the convex combination $\lambda\mathbf{x} + (1 - \lambda)\mathbf{y}$ is also in $\mathcal{M}$. If a point $\boldsymbol{\omega}$ in a polyhedron $\mathcal{M}$ cannot be written as a convex combination of two different points in $\mathcal{M}$, we say that $\boldsymbol{\omega}$ is an *extreme point* or a *vertex* of the polyhedron. Given a point $\mathbf{y} \in \mathbb{R}^n$ and a constraint of the form $\mathbf{a}^T\mathbf{x} \leq b, \mathbf{a}^T\mathbf{x} \geq b,$ or $\mathbf{a}^T\mathbf{x} = b$, we say that the constraint is *active* or *binding* at $\mathbf{y}$ provided that the constraint is met with equality, i.e., if $\mathbf{a}^T\mathbf{y} = b$.

Given a finite set of vectors $\mathcal{V} = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_\ell\} \subseteq \mathbb{R}^n$ and a set of non-negative scalars $\{w_1, w_2, \ldots, w_\ell\}$, we say that $\sum_{i=1}^{\ell} w_i\mathbf{v}_i$ is a *conic combination* of the vectors in $\mathcal{V}$. The polyhedral cone generated by $\mathcal{V}$ is then the set of all conic combinations of vectors in $\mathcal{V}$. It is initially unclear that $\mathcal{K}(\mathcal{V})$ is a polyhedron since it is not explicitly described as the set of all vectors satisfying a finite set of equality and inequality constraints. A classical approach to affirming this fact is *Fourier-Motzkin elimination* (see, e.g., [4]), a technique that produces a complete set of linear equality and inequality constraints for a projection of a polyhedron onto a subset of its coordinates.

For our purposes we define an optimization problem to be the task of maximizing or minimizing an *objective function* over all points in a *feasible set*. A *linear program (LP)* is simply an optimization problem for which the objective function is linear and whose feasible set is a polyhedron. It is well known that if an optimal solution to a linear program exists, then there is at least one extreme point of the underlying polyhedron that achieves this optimal value. Conversely, given an extreme point $\boldsymbol{\omega}$ of a

polyhedron $\mathcal{M}$ there must exist some linear function $\mathbf{c}$ so that $\boldsymbol{\omega}$ is the unique minimizer of $\mathbf{c}^T\mathbf{x}$ over all $\mathbf{x} \in \mathcal{M}$ (see, e.g., pages 46-52 of [4]).

B. Background on Linear Programming Decoding

In [6] the *linear programming decoder* is introduced. This decoder operates by solving a linear programming relaxation of the maximum-likelihood (ML) decoding problem. In particular, the decoder outputs a solution to the following linear program:

$$\begin{array}{ll} \text{minimize} & \boldsymbol{\lambda}^T\mathbf{x} \\ \text{subject to} & \mathbf{x} \in \mathcal{P}(H) \end{array}$$

where $\boldsymbol{\lambda}$ is the vector of log-likelihood ratios determined from the channel output and $\mathcal{P}(H) \subseteq \mathbb{R}^n$ is a polytope whose constraints are determined by the parity-check matrix H chosen to represent the code.

Definition II.1 ([6])**.** Let C be a code presented by an $m \times n$ parity-check matrix $H = (h_{j,i})$. For any row $\mathbf{h}_j$ of H, define $N(\mathbf{h}_j)$ to be the set of all indices $i \in \{1, 2, \ldots, n\}$ with $h_{j,i} = 1$. The *fundamental polytope*[1] $\mathcal{P} = \mathcal{P}(H)$ is the set of all vectors $\mathbf{x} \in \mathbb{R}^n$ satisfying the following constraints:

- $0 \leq x_i \leq 1$ for all $i = 1, 2, \ldots, n$ and
- $\sum_{i \in S} x_i + \sum_{i' \in N(\mathbf{h}_j) \setminus S} (1 - x_{i'}) \leq |N(\mathbf{h}_j)| - 1$ for all pairs (j, S), where $j \in \{1, 2, \ldots, m\}$ and S is a subset of $N(\mathbf{h}_j)$ with odd cardinality.

Since the fundamental polytope $\mathcal{P}(H)$ is bounded the linear programming decoder will always have an optimal solution, and by the preceding discussion we can assume that the linear programming decoder always returns an extreme point of $\mathcal{P}(H)$ as its output. As in [3], we define a *linear programming (LP) pseudocodeword* to be any extreme point of $\mathcal{P}(H)$. In [6], it is shown that all codewords are extreme points of $\mathcal{P}(H)$ and conversely that all integer-valued vectors within $\mathcal{P}(H)$ must be codewords. Again following [3], we therefore refer to codewords as *trivial linear programming (LP) pseudocodewords* and any extreme points of $\mathcal{P}(H)$ with fractional values as *nontrivial linear programming (LP) pseudocodewords*. Since this paper is concerned only with the LP decoder, we will often refer to these simply as trivial and nontrivial pseudocodewords.

In [6], it is shown that if the linear programming decoder outputs an integer-valued codeword then this codeword must be a maximum-likelihood codeword. Since the set of codewords and the set of of integer-valued points in $\mathcal{P}(H)$ are identical, this implies that the presence of nontrivial LP pseudocodewords is precisely what prevents the LP decoder from achieving ML performance. Much research has therefore been devoted to examining the structure and properties of nontrivial pseudocodewords (see, e.g., [3], [8], [9], [10], [12], [13], [14]). The authors of [15] go so far as to determine whether for a given code $\mathcal{C}$ there exists a parity-check matrix H for which $P(H)$ contains no nontrivial pseudocodewords – for most binary linear codes, the answer is "no" [15]. Nontrivial pseudocodewords are ubiquitous, and yet for certain received vectors they *are* optimal solutions to the LP decoding problem. Taking the saying "If you can't beat them, join them" to heart, we seek a method of incorporating nontrivial pseudocodewords into our codebooks so as to increase information rates and spectral efficiency without sacrificing error-correcting performance.

III. How To Transmit a Pseudocodeword

When transmitting binary values over the additive white Gaussian noise channel it is common to employ binary phase shift keying (BPSK), which without loss of generality sends a binary 0 to $+1$ and a binary 1 to -1. As a first (and very naive) attempt at transmitting nontrivial pseudocodewords, we simply extended this modulation mapping via the affine map $x \mapsto 1 - 2x$, added Gaussian noise, and applied LP decoding

[1] In [6], this is referred to as the *projected* polytope.

to the resulting vector. The performance of this modulation scheme was terrible – the block error rate was consistently close to 1 at all decibel levels. On investigation, we discovered that even when *no noise* was added to our modulated vector the LP decoder was unable to recover the original pseudocodeword. We concluded that to have any hope of transmitting a pseudocodeword a fundamentally new modulation scheme would be required.

The key observation in deriving our new modulation scheme for the transmission of linear programming pseudocodewords over the additive white Gaussian noise channel is that every pseudocodeword, trivial or nontrivial, is an extreme point of the fundamental polytope $\mathcal{P}(H)$. By the discussion at the end of Section II-A, given a pseudocodeword $\boldsymbol{\omega} \in \mathcal{P}$ there must exist a vector of log-likelihood ratios $\boldsymbol{\lambda}$ so that $\boldsymbol{\omega}$ is the *unique* solution to the following optimization problem: minimize $\boldsymbol{\lambda}^T\mathbf{x}$ over all $\mathbf{x} \in \mathcal{P}$. We therefore aim to produce a modulated vector that is equal to such a vector of log-likelihood ratios, since this will ensure that when $\boldsymbol{\omega}$ is transmitted in a low-noise scenario the LP decoder will recover the pseudocodeword $\boldsymbol{\omega}$. Since on the AWGN channel the vector of log-likelihood ratios is proportional to the received vector, we can simply transmit the vector $\boldsymbol{\lambda}$ to accomplish this. For this reason we will conflate the terms "received vector" and "vector of log-likelihood ratios."

As we prove in Proposition III.1 there are usually infinitely many choices for $\boldsymbol{\lambda}$. It is our task to find a choice that minimizes the probability of block error.

Proposition III.1. *Let $\mathcal{M} = \{\mathbf{x} \,|\, A\mathbf{x} \geq \mathbf{b}\}$ be a polyhedron in $\mathbb{R}^n$, let $\mathbf{x}^* \in \mathcal{M}$ be given, and define S to be the set of all indices i where $\mathbf{a}_i^T\mathbf{x}^* = b_i$ - i.e., the index set for all constraints that are active at $\mathbf{x}^*$. Consider the problem of minimizing $\mathbf{c}^T\mathbf{x}$ over all $\mathbf{x} \in \mathcal{M}$: the vector $\mathbf{x}^*$ is an optimal solution to this problem if and only if $\mathbf{c}^T$ is a conic combination of the rows of A indexed by S.*

Proof. Define A_S to be the matrix obtained by only considering those rows of A indexed by S and suppose that $\mathbf{c}^T$ is not a conic combination of the rows in A_S. This means that there is no vector $\mathbf{w}$ satisfying both $\mathbf{c}^T = \mathbf{w}^T A_S$ and $\mathbf{w} \geq \mathbf{0}$. By Farkas' Lemma (see, e.g., [4]), there must exist a vector $\mathbf{d}$ with $A_S\mathbf{d} \geq 0$ and $\mathbf{c}^T\mathbf{d} < 0$. Consider now the vector $\mathbf{x}^* + \theta\mathbf{d}$. Since $\mathbf{a}_i^T\mathbf{x}^* > b_i$ for all $i \notin S$, if we choose $\theta > 0$ to be sufficiently small then $\mathbf{x}^* + \theta\mathbf{d}$ will satisfy $A(\mathbf{x}^* + \theta\mathbf{d}) \geq \mathbf{b}$ and thus be an element of $\mathcal{M}$. Computing the objective value of this new vector in $\mathcal{M}$, we obtain

$$\mathbf{c}^T(\mathbf{x}^* + \theta\mathbf{d}) = \mathbf{c}^T\mathbf{x}^* + \theta\mathbf{c}^T\mathbf{d} < \mathbf{c}^T\mathbf{x}^*,$$

which implies that $\mathbf{x}^*$ cannot be an optimal solution to the minimization problem.

For the other direction, we use a modification of the proof that any basic feasible solution of a linear program is also a vertex (see, e.g., the discussion on page 52 of [4]). We now reproduce this proof with appropriate modifications for the sake of completeness. Suppose that $\mathbf{c}^T = \sum_{i \in S} w_i \mathbf{a}_i^T$ for some scalars w_i with $w_i \geq 0$ for all $i \in S$. If $\mathbf{x}$ is an arbitrary point in $\mathcal{M}$, we have $\mathbf{c}^T\mathbf{x} = \sum_{i \in S} w_i \mathbf{a}_i^T\mathbf{x} \geq \sum_{i \in S} w_i b_i$. The number $\sum_{i \in S} w_i b_i$ is therefore a lower bound on the objective value for any point in $\mathcal{M}$. But by the definition of S, we have that $\mathbf{a}_i^T\mathbf{x} = b_i$ for all $i \in S$. Thus, $\mathbf{c}^T\mathbf{x}^* = \sum_{i \in S} w_i \mathbf{a}_i^T\mathbf{x}^* = \sum_{i \in S} w_i b_i$. The vector $\mathbf{x}^*$ attains this bound and therefore must be an optimal solution to the problem of minimizing $\mathbf{c}^T\mathbf{x}$ subject to $A\mathbf{x} \geq \mathbf{b}$.

□

To apply Proposition III.1 to our coding problem, we make the following definition.

Definition III.2. Let $\mathcal{C}$ be a code presented by the parity-check matrix H, let $\mathcal{P}(H)$ be the associated fundamental polytope, and assume that $\mathcal{P}(H)$ has been written as $\mathcal{P}(H) = \{\mathbf{x} \,|\, A\mathbf{x} \geq \mathbf{b}\}$ for an appropriate matrix A. For any linear programming pseudocodeword $\boldsymbol{\omega}$, the *recovery cone* $\mathcal{K}_{\boldsymbol{\omega}}(H) = \mathcal{K}_{\boldsymbol{\omega}}$ of $\boldsymbol{\omega}$ is the cone generated by all row vectors of A for which the corresponding constraint of $\mathcal{P}(H)$ is binding at $\boldsymbol{\omega}$.

Let $t(\boldsymbol{\omega})$ be a point in the recovery cone $\mathcal{K}_{\boldsymbol{\omega}}$ and let $\boldsymbol{\eta}$ represent the additive Gaussian noise introduced by the channel. By Proposition III.1, the nontrivial pseudocodeword $\boldsymbol{\omega}$ can be correctly recovered if and only if the received vector $t(\boldsymbol{\omega}) + \boldsymbol{\eta}$ lies within $\mathcal{K}_{\boldsymbol{\omega}}$. If r is the radius of the largest sphere centered at

$t(\boldsymbol{\omega})$ yet contained completely within $K_{\boldsymbol{\omega}}$, then the probability of correct decoding is bounded below by the probability that $|\boldsymbol{\eta}| \leq r$.

A trivial way of driving the probability of correct decoding to 1 is simply to scale $t(\boldsymbol{\omega})$ by a large positive constant, which is tantamount to transmitting with more energy. It is more interesting to place an energy constraint on the transmitted vector and then search for a vector that maximizes the radius of the largest sphere centered at that vector and contained within $\mathcal{K}_{\boldsymbol{\omega}}$. Stated more formally, we seek to compute the *insphere* of $\mathcal{K}_{\boldsymbol{\omega}}$. This matter is treated separately in Section IV.

Our approach to computing the insphere of a polyhedral cone depends on having access to an explicit description of the cone in terms of linearly inequality constraints. While Fourier-Motzkin elimination (see, e.g., [4]) can produce such a description, it is terribly inefficient: for codes of small block length ($n \leq 20$) Fourier-Motzkin elimination produces the constraint matrix for $\mathcal{K}_{\boldsymbol{\omega}}$ in a reasonable amount of time (a few minutes to a few hours on a desktop computer). For longer codes, however, we approximate $\mathcal{K}_{\boldsymbol{\omega}}$ by examining a larger cone that contains it.

Definition III.3. Let $\mathcal{C}$ be a code presented by the parity-check matrix H, let $\boldsymbol{\omega}$ be a linear programming pseudocodeword, and let $\Psi = \{\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \ldots, \boldsymbol{\psi}_r\}$ be a set of linear programming pseudocodewords. The *approximation cone* $\mathcal{R}_{\boldsymbol{\omega},\Psi}(H) = \mathcal{R}_{\boldsymbol{\omega},\Psi}$ is defined to be the set of all vectors $\mathbf{x}$ satisfying $R_{\boldsymbol{\omega},\Psi}\mathbf{x} \geq 0$, where the ith row of $R_{\boldsymbol{\omega},\Psi}$ is given by $(\boldsymbol{\psi}_i - \boldsymbol{\omega})^T$.

Proposition III.4. *Let $\mathcal{C}$ be a code presented by the parity-check matrix H. If $\boldsymbol{\omega}$ is a linear programming pseudocodeword and $\Psi = \{\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \ldots, \boldsymbol{\psi}_r\}$ is a set of linear programming pseudocodewords, then the recovery cone $\mathcal{K}_{\boldsymbol{\omega}}$ is a subset of the approximation cone $\mathcal{R}_{\boldsymbol{\omega},\Psi}$.*

Proof. The inequality $(\boldsymbol{\psi}_i - \boldsymbol{\omega})^T\mathbf{y} \geq 0$ describes all cost functions $\mathbf{y}$ of the linear programming decoder for which $\boldsymbol{\omega}$ has an objective value that is no larger than that of $\boldsymbol{\psi}_i$. By Proposition III.1, it follows that every vector $\mathbf{x} \in \mathcal{K}_{\boldsymbol{\omega}}$ satisfies $(\boldsymbol{\psi}_i - \boldsymbol{\omega})^T\mathbf{x} \geq 0$. Intersecting over all i we obtain $\mathcal{K}_{\boldsymbol{\omega}} \subseteq \mathcal{R}_{\boldsymbol{\omega},\Psi}$. □

We can generate an approximation cone for the pseudocodeword $\boldsymbol{\omega}$ by repeatedly transmitting a vector in $\mathcal{K}_{\boldsymbol{\omega}}$ (say, the sum of all constraint vectors of $\mathcal{P}(H)$ that are binding at $\boldsymbol{\omega}$) in a low signal-to-noise environment and recording the pseudocodewords that arise as the LP decoder's output. If enough unique pseudocodewords are found, the insphere of the corresponding approximation cone ought to be a good approximation of the insphere of the recovery cone – discussion regarding the corresponding block error rates can be found in Sections VI and VII.

In summary, Proposition III.1 states that any conic combination of constraint vectors that are active at the pseudocodeword $\boldsymbol{\omega}$ can be used to reliably transmit $\boldsymbol{\omega}$. A lower bound for the probability of correctly recovering $\boldsymbol{\omega}$ over the AWGN channel can be maximized by transmitting a vector proportional to the center of the insphere of the recovery cone. With this in mind, we define *insphere modulation* as the method of mapping $\boldsymbol{\omega}$ to the center of an appropriate insphere, whether it is the insphere of the recovery cone or an approximation cone.

IV. A Method of Computing the Insphere of a Polyhedral Cone

In this section we develop an iterative algorithm capable of approximating the insphere of a polyhedral cone to any degree of accuracy. While our primary application will be to produce modulated versions of pseudocodewords that maximize a lower bound on the probability of block error, the results of this section may be of independent mathematical interest.

Throughout this section, we use $B_r(\mathbf{x})$ to denote the closed sphere (or ball) of radius r centered at $\mathbf{x}$, i.e., $B_r(\mathbf{x}) := \{\mathbf{y} \in \mathbb{R}^n \mid \|\mathbf{x} - \mathbf{y}\| \leq r\}$.

Definition IV.1 ([7])**.** Let $\mathcal{K}$ be a polyhedral cone in $\mathbb{R}^n$. The *insphere* of $\mathcal{K}$ is the largest sphere contained in $\mathcal{K}$ whose center is in $B_1(\mathbf{0})$, and the *inradius* of $\mathcal{K}$ is the radius of the insphere.

It is a consequence of Theorem 2.4 in [7] that any polyhedral cone with a non-empty topological interior possesses a unique insphere. Since polyhedral cones with empty interiors are of little use for transmitting

pseudocodewords, we assume that our polyhedral cone is full dimensional and thus has a unique insphere. Moreover, we observe that the center of such an insphere will necessarily have unit magnitude.

Throughout this section we will assume that we are in possession of an explicit description of $\mathcal{K}$ in terms of linear inequality constraints $K\mathbf{x} \geq \mathbf{0}$. By scaling we can ensure that every row $\mathbf{k}_i^T$ of K has unit length - as such, $\mathbf{k}_i^T\mathbf{x}$ measures the Euclidean distance from the point $\mathbf{x}$ to the hyperplane defined by the ith row of K. Following an idea similar to that employed to find inspheres of two-dimensional polyhedra given in [11], the insphere can be found by maximizing a lower bound z on the distance from a point in $\mathcal{K} \cap B_1(\mathbf{0})$ to any defining hyperplane. This idea is expressed succinctly by the following nonlinear convex optimization problem that, for reasons that will soon be become clear, we call Problem P_∞.

$$\begin{aligned} &\max \ z && (P_\infty) \\ &\text{subject to } \mathbf{k}_i^T\mathbf{x} \geq z \text{ for all } i \\ &\qquad \|\mathbf{x}\|^2 \leq 1 \end{aligned}$$

If $(\mathbf{x}_\infty, z_\infty)$ is an optimal solution to Problem P_∞, then $\mathbf{x}_\infty$ will give the center of the insphere and z_∞ will give the inradius.

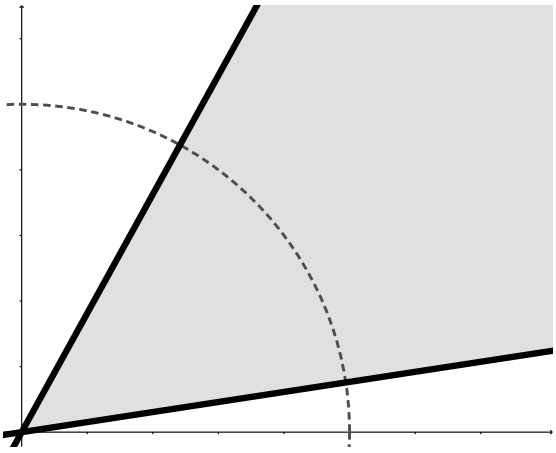
The special structure of Problem P_∞ admits an interesting algorithm that seeks to model the nonlinear boundary of the constraint $\|\mathbf{x}\|^2 = \sum_{i=1}^n x_i^2 \leq 1$ by iteratively adding linear inequality constraints. Our algorithm operates by defining a sequence of linear programming problems and examining the optimal solutions of each. At each stage, we also derive an upper and lower bounds on the inradius of $\mathcal{K}$. The primary result of this section is that the sequence of optimal solutions converges to the center of the insphere. Moreover, the upper and lower bounds on the inradius are easily computable and can be used to terminate computations once the desired degree of accuracy is attained.

To initialize our algorithm we form a linear program P_0 that contains all of the linear constraints of problem P_∞ but eschews the nonlinear constraint in favor of $2n$ linear constraints that confine potential solutions to a bounded hypercube. This is done to ensure that an optimal solution to this problem (and all subsequent problems) exist. Formally, we defined problem P_0 as the following linear program:

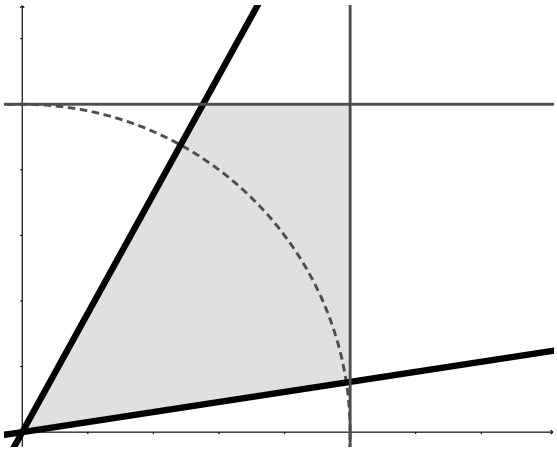
$$\begin{aligned} &\max \ z && (P_0) \\ &\text{subject to } \mathbf{k}_i^T\mathbf{x} \geq z \text{ for all } i \\ &\qquad -1 \leq x_i \leq 1 \text{ for all } i \end{aligned}$$

We formulate problem $P_{\ell+1}$ based off of problem P_ℓ as follows. We first compute an optimal solution $(\mathbf{x}_\ell, z_\ell)$ to problem P_ℓ (such a solution is readily obtained from any LP solver) and then compute the unit vector $\mathbf{u}_\ell$ in the direction of $\mathbf{x}_\ell$. We note that the optimality of $\mathbf{x}_\ell$ implies that $\|\mathbf{x}_\ell\| \geq 1$, which among other things implies that $\mathbf{x}_\ell \neq \mathbf{0}$ and so $\mathbf{u}_\ell$ is defined for all ℓ. Next, we record the value $w_\ell = \frac{z_\ell}{\|\mathbf{x}_\ell\|} = \min_i\{\mathbf{k}_i^T\mathbf{u}_\ell\}$, since w_ℓ is the radius of the largest sphere centered at the unit vector $\mathbf{u}_\ell$ and contained within $\mathcal{K}$ and thus forms a lower bound on the inradius of $\mathcal{K}$. We define Problem $P_{\ell+1}$ by duplicating Problem P_ℓ and requiring that the additional constraint $\mathbf{u}_\ell^T\mathbf{x} \leq 1$ also be met. The general form of Problem $P_{\ell+1}$ is given as follows:

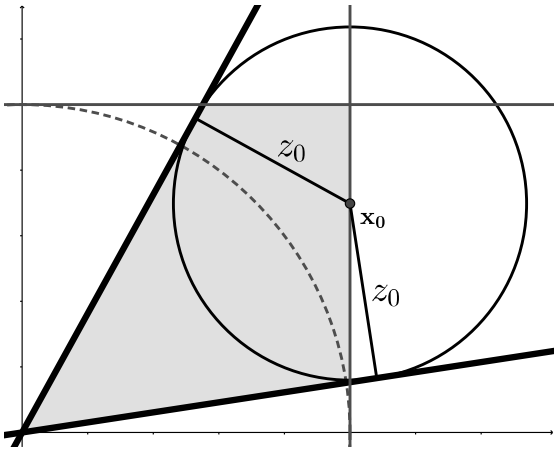
$$\begin{aligned} &\max \ z && (P_{\ell+1}) \\ &\text{subject to } \mathbf{k}_i^T\mathbf{x} \geq z \text{ for all } i \\ &\qquad -1 \leq x_i \leq 1 \text{ for all } i \\ &\qquad \mathbf{u}_0^T\mathbf{x} \leq 1 \\ &\qquad \vdots \\ &\qquad \mathbf{u}_{\ell-1}^T\mathbf{x} \leq 1 \\ &\qquad \mathbf{u}_\ell^T\mathbf{x} \leq 1 \end{aligned}$$



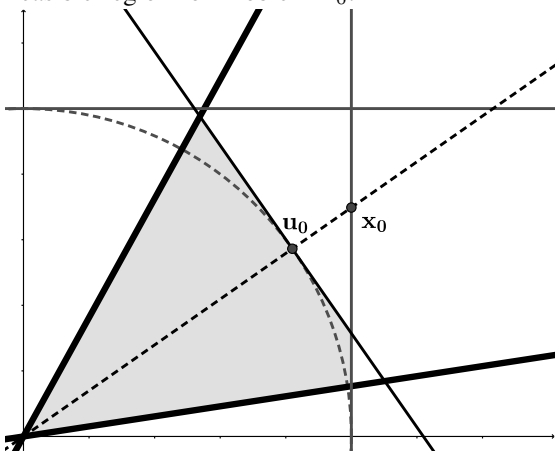
(a) A cone with a unit circle superimposed.



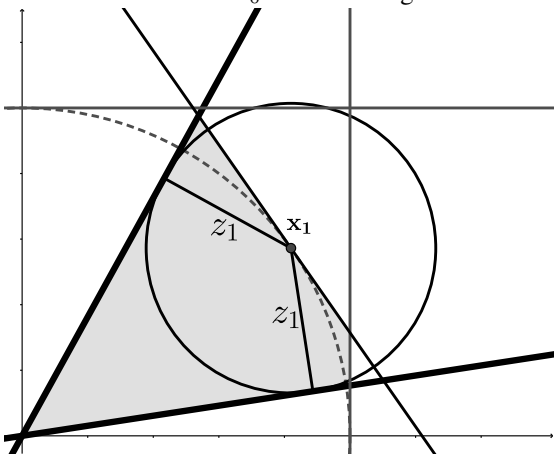
(b) Additional constraints are added to create the feasible region for Problem P_0.



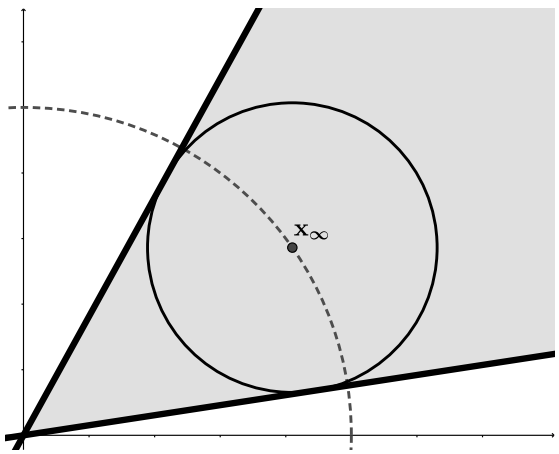
(c) The largest sphere contained in the cone and centered in Problem P_0's feasible region is found.



(d) A new constraint is added to form the feasible set of Problem P_1.



(e) The largest sphere contained in the cone and centered in Problem P_1's feasible region is found.



(f) Since $\mathbf{x}_1$ lies on the unit circle, the insphere of the cone has been found.

Fig. 1: An illustration of our iterative method for computing the insphere of a polyhedral cone to any degree of accuracy. For this small example in $\mathbb{R}^2$, the exact insphere is found in only two steps.

This process of iteratively approximating the insphere of a polyhedral cone is illustrated in Figure 1.

With these definitions one can see that the feasible sets for problems $P_0, P_1, \ldots, P_\ell, \ldots, P_\infty$ are nested within each other, forming a descending chain of sets with respect to inclusion. Since all are maximization problems, it follows that the sequence of z_ℓ's is monotonically decreasing. By comparison, the sequence of w_ℓ's has been observed to be generally increasing in nature, though not always monotonically (see Figure 2). Still, we know from a previous observation that w_ℓ forms a lower bound on z_∞. We obtain Proposition IV.2 by combining these results.

Proposition IV.2. *Let $\mathcal{K}$ be a polyhedral cone with non-empty topological interior and let z_∞ denote its inradius. Let $(\mathbf{x}_\ell, z_\ell)$ be an optimal solution to Problem P_ℓ, and define $w_\ell = \frac{z_\ell}{\|\mathbf{x}_\ell\|}$. With this notation, we have that*

$$z_0 \geq z_1 \geq \cdots \geq z_\ell \geq \cdots \geq z_\infty.$$

Moreover, for any pair of integers m and n we have

$$z_m \geq z_\infty \geq w_n.$$

Proposition IV.2 states that the inradius always lies between w_ℓ and z_ℓ. To compute the inradius to within δ of its true value, once can compute solutions to these problems until $z_\ell - w_\ell \leq \delta$. Computational results, such as those summarized in Figure 2, suggest that $z_\ell - w_\ell$ can always be driven to be arbitrarily small. As we shall see from Theorem IV.4, this is indeed the case. To prove this theorem, we first present a lemma.

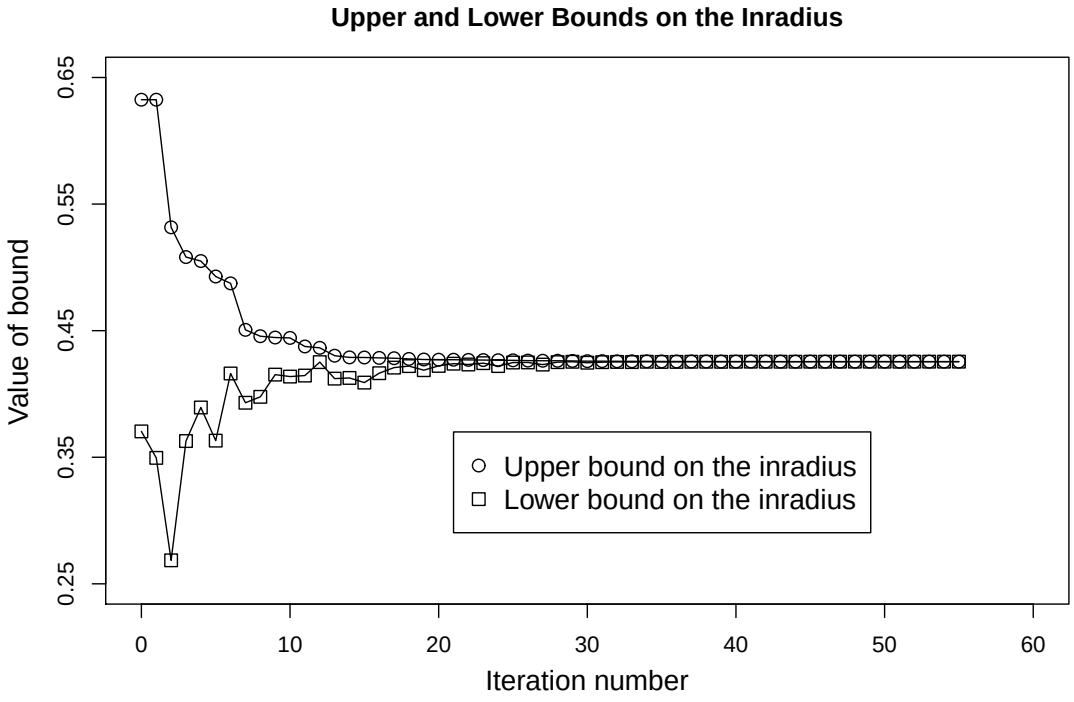


Fig. 2: A plot of upper bounds z_ℓ and lower bounds w_ℓ on the inradius for an approximation cone used to compute a transmittable version of the pseudocodeword $\boldsymbol{\omega}_3$ (see Section VI). The cone is described by 35 constraints in $\mathbb{R}^{16}$. In this example, it takes 56 iterations to ensure that the upper and lower bounds are within 10^{-5} of one another.

Lemma IV.3. *Let $\mathcal{K}$ be a polyhedral cone with non-empty topological interior, let $(\mathbf{x}_\ell, z_\ell)$ be an optimal solution to Problem P_ℓ, and define $\mathbf{u}_\ell = \frac{\mathbf{x}_\ell}{\|\mathbf{x}_\ell\|}$. For any r with $0 \leq r < \|\mathbf{x}_\ell - \mathbf{u}_\ell\|$, $B_r(\mathbf{x}_\ell)$ and the feasible set for Problem $P_{\ell+1}$ are disjoint.*

The proof of Lemma IV.3 is virtually identical to the proof of the separating hyperplane theorem - see, e.g., pages 170-172 of [4]. This theorem states that given a non-empty, closed convex set S in $\mathbb{R}^n$ and a point $\mathbf{x} \notin S$, there exists a hyperplane in $\mathbb{R}^n$ for which $\mathbf{x}$ is on one side and the entirety of S lies on the other. Lemma IV.3 merely makes this specific, stating that a small sphere around $\mathbf{x}_\ell$ can be separated from $\mathbf{u}_\ell$ by the inequality $\mathbf{u}_\ell^T \mathbf{x} \leq 1$, which is precisely the inequality added to Problem P_ℓ to form Problem $P_{\ell+1}$.

Theorem IV.4. *Let $\mathcal{K}$ be a polyhedral cone with non-empty topological interior, and let $\mathbf{x}_\infty$ and z_∞ denote the center of its insphere and its inradius, respectively. If $(\mathbf{x}_\ell, z_\ell)$ is an optimal solution to Problem P_ℓ, then $\mathbf{x}_\ell$ converges to $\mathbf{x}_\infty$ and z_ℓ converges to z_∞.*

Proof. Define $\mathbf{u}_\ell := \frac{\mathbf{x}_\ell}{\|\mathbf{x}_\ell\|}$ and suppose for the sake of a contradiction that $\mathbf{x}_\ell - \mathbf{u}_\ell$ does not converge to the zero vector. This implies that there is an infinite subsequence ℓ_k and some $\epsilon_0 > 0$ so that $\|\mathbf{x}_{\ell_k} - \mathbf{u}_{\ell_k}\| \geq \epsilon_0$ for all k. Since the feasible sets of all Problems P_ℓ are contained within the feasible set for Problem P_0, which is bounded, the sequence $\mathbf{x}_{\ell_k}$ is bounded. The Bolzano-Weierstrass Theorem implies that there is a subsequence $\mathbf{x}_{\ell_{k_j}}$ of $\mathbf{x}_{\ell_k}$ that converges to some vector.

Since $\mathbf{x}_{\ell_{k_j}}$ is a convergent sequence, it must be a Cauchy sequence. Thus, there a J so that for all $j', j'' \geq J$ we have $\left\|\mathbf{x}_{\ell_{k_{j'}}} - \mathbf{x}_{\ell_{k_{j''}}}\right\| \leq \epsilon_0/4$. The triangle inequality implies that for all $j' \geq J$ the vector $\mathbf{x}_{\ell_{k_{j'}}}$ is in $B_{\epsilon_0/2}(\mathbf{x}_{\ell_{k_J}})$. Since $\left\|\mathbf{x}_{\ell_{k_j}} - \mathbf{u}_{\ell_{k_j}}\right\| \geq \epsilon_0$ for all j, Lemma IV.3 implies that $B_{\epsilon_0/2}(\mathbf{x}_{\ell_{k_J}})$ is disjoint from the feasible set of Problem $P_{\ell_{j'}}$ for all $j' > J$, which implies that $\mathbf{x}_{\ell_{k_{J+1}}}$ is infeasible for Problem $P_{\ell_{k_{J+1}}}$. This is a contradiction, since $\mathbf{x}_{\ell_{k_{J+1}}}$ was taken to be an optimal solution to Problem $P_{\ell_{k_{J+1}}}$. We conclude that $\mathbf{x}_\ell - \mathbf{u}_\ell$ converges to the zero vector.

Since $\mathbf{x}_\ell - \mathbf{u}_\ell$ converges to the zero vector and the individual sequences $\mathbf{x}_\ell$ and $\mathbf{u}_\ell$ are both bounded, we can conclude that the individual sequences $\mathbf{x}_\ell$ and $\mathbf{u}_\ell$ converge to a common limit $\mathbf{x}^*$, which is necessarily a unit vector. With $w_\ell = \frac{z_\ell}{\|\mathbf{x}_\ell\|}$, the convergence of $\mathbf{x}_\ell$ and $\mathbf{u}_\ell$ to a common limit implies that z_ℓ and w_ℓ converge to a common limit, which by Proposition IV.2 must be the inradius of $\mathcal{K}$. This, along with the uniqueness of the insphere (see Theorem 2.4 in [7]) implies that $\mathbf{x}^* = \mathbf{x}_\infty$. □

V. Extending The Modulation Scheme via $\mathcal{C}$-Symmetry

Using the results of Sections III and IV, one can obtain a transmitted vector that can reliably recover a linear programming pseudocodeword. While finding such a vector takes some work, once computed it can be easily modified to allow for the broadcast of many more pseudocodewords.

In order to show that the probability of block error is independent of the codeword transmitted, the concept of *$\mathcal{C}$-symmetry* is introduced in [5][2].

Definition V.1 ([5])**.** Let $\mathcal{C}$ be a code presented by the parity-check matrix H, and let $\mathbf{c} \in \mathcal{C}$ be given. For any point $\mathbf{x} \in \mathcal{P}(H)$, the *relative point* $\mathbf{x}^\mathbf{c}$ of $\mathbf{x}$ with respect to $\mathbf{c}$ is the point whose ith coordinate is given by $|x_i - c_i|$ for all $i = 1, 2, \ldots, n$.

Theorem V.2 ([5])**.** *Let $\mathcal{C}$ be a code presented by the parity-check matrix H. For any point $\mathbf{x} \in \mathcal{P}(H)$ and any codeword $\mathbf{c} \in \mathcal{C}$, the relative point $\mathbf{x}^\mathbf{c}$ is also an element of $\mathcal{P}(H)$. Further, if $\boldsymbol{\omega}$ is an extreme point of $\mathcal{P}(H)$, then $\boldsymbol{\omega}^\mathbf{c}$ is also an extreme point of $\mathcal{P}(H)$.*

Given a codeword $\mathbf{c} \in \mathcal{C}$, define the map $\zeta_\mathbf{c} : \mathcal{P}(H) \to \mathcal{P}(H)$ by $\zeta_\mathbf{c}(\mathbf{x}) = \mathbf{x}^\mathbf{c}$. Since $\zeta_\mathbf{c}$ is an isometry of $\mathbb{R}^n$, it is also an isometry of $\mathcal{P}(H)$. Additionally, the map $\zeta_\mathbf{c}$ defines a group action of the code $\mathcal{C}$ (viewed as an abelian group under vector addition) on $\mathcal{P}(H)$ as well as on the set of extreme points of $\mathcal{P}(H)$, i.e., the set of LP pseudocodewords.

For a row $\mathbf{h}_j$ of H and an odd-sized subset S of $N(\mathbf{h}_j)$, let (j, S) denote the constraint of the fundamental polytope given by $\sum_{i \in S} x_i + \sum_{i' \in N(\mathbf{h}_j) \setminus S} (1 - x_{i'}) \leq |N(\mathbf{h}_j)| - 1$, which we rewrite as $-\sum_{i \in S} x_i +$

[2]Though the term "$\mathcal{C}$-symmetry" is never used in [6], the relevant concepts are present there as well.

$\sum_{i' \in N(\mathbf{h}_j) \setminus S} x_{i'} \geq 1 - |S|$. For a codeword $\mathbf{c} \in \mathcal{C}$, let $S^{\mathbf{c}} := S \triangle (\mathrm{supp}(\mathbf{c}) \cap N(\mathbf{h}_j))$, where the "$\triangle$" operator denotes the symmetric difference of two sets. By following these definitions, one can prove the following proposition:

Proposition V.3. *Let $\mathcal{C}$ be a binary code presented by the parity-check matrix H with fundamental polytope $\mathcal{P}(H)$. Let j be the index of a row $\mathbf{h}_j$ of H and S be an odd-sized subset of $N(\mathbf{h}_j)$, and let $\mathbf{x} \in \mathcal{P}(H)$ and $\mathbf{c} \in C$ be given. If the constraint defined by the pair (j, S) is active at $\mathbf{x}$, then the constraint $(j, S^{\mathbf{c}})$ is active at the relative point $\mathbf{x}^{\mathbf{c}}$.*

Using Proposition V.3, we see that the set of constraint vectors of the fundamental polytope that are active at $\boldsymbol{\omega}^{\mathbf{c}}$ can be obtained by first taking the set of constraints vectors that are active at $\boldsymbol{\omega}$ and multiplying each component in the support of $\mathbf{c}$ by -1. Moreover, if $\Psi = \{\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \ldots, \boldsymbol{\psi}_r\}$ is a set of LP pseudocodewords and $\Psi^{\mathbf{c}} := \{\boldsymbol{\psi}_1^{\mathbf{c}}, \boldsymbol{\psi}_2^{\mathbf{c}}, \ldots, \boldsymbol{\psi}_r^{\mathbf{c}}\}$, then the approximation cone $\mathcal{R}_{\boldsymbol{\omega}^{\mathbf{c}}, \Psi^{\mathbf{c}}}$ can be obtained by applying the same transformation to $\mathcal{R}_{\boldsymbol{\omega}, \Psi}$. These facts are summarized in the following proposition.

Proposition V.4. *Let $\mathcal{C}$ be a code presented by the parity-check matrix H, let $\mathbf{c}$ be a codeword and $\boldsymbol{\omega}$ be a linear programming pseudocodeword, and let $\Psi = \{\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \ldots, \boldsymbol{\psi}_r\}$ be a set of linear programming pseudocodewords. The map $\phi_{\mathbf{c}} : \mathbb{R}^n \to \mathbb{R}^n$ defined by $x_i \to (-1)^{c_i} x_i$ for all $i = 1, 2, \ldots, n$ is an isometry between the recovery cones $\mathcal{K}_{\boldsymbol{\omega}}$ and $\mathcal{K}_{\boldsymbol{\omega}^{\mathbf{c}}}$ as well as an isometry between the approximation cones $\mathcal{R}_{\boldsymbol{\omega}, \Psi}$ and $\mathcal{R}_{\boldsymbol{\omega}^{\mathbf{c}}, \Psi^{\mathbf{c}}}$.*

Since the probability density function for additive white Gaussian noise with a given power spectral density is symmetric about any such isometry defined in Proposition V.4, we have the following result.

Theorem V.5. *Let $\mathcal{C}$ be a code presented by the parity-check matrix H, let $\mathbf{c}$ be a codeword and $\boldsymbol{\omega}$ be a linear programming pseudocodeword, and define the map $\phi_{\mathbf{c}} : \mathbb{R}^n \to \mathbb{R}^n$ by $x_i \to (-1)^{c_i} x_i$ for all $i = 1, 2, \ldots, n$. If $t(\boldsymbol{\omega})$ is a vector in the recovery cone $\mathcal{K}_{\boldsymbol{\omega}}$ suitable for transmitting $\boldsymbol{\omega}$ over the additive white Gaussian noise channel, then $\phi_{\mathbf{c}}\Big(t(\boldsymbol{\omega})\Big)$ is a vector in $\mathcal{K}_{\boldsymbol{\omega}^{\mathbf{c}}}$ suitable for transmitting $\boldsymbol{\omega}^{\mathbf{c}}$. Moreover, the probability of correctly decoding to $\boldsymbol{\omega}$ when $t(\boldsymbol{\omega})$ is transmitted is equal to the probability of correctly decoding to $\boldsymbol{\omega}^{\mathbf{c}}$ when $\phi_{\mathbf{c}}\Big(t(\boldsymbol{\omega})\Big)$ is transmitted.*

Theorem V.5 has a practical consequence: once we have found a vector that provides a low probability of error in recovering the pseudocodeword $\boldsymbol{\omega}$, we can simply apply the map $\phi_{\mathbf{c}}$ to this vector to transmit another pseudocodeword in $\boldsymbol{\omega}$'s $\mathcal{C}$-symmetric orbit.

VI. A Detailed Example

In this section we provide a detailed example of a code, the orbits of its pseudocodewords, the modulated vectors used to transmit these pseudocodewords, and the probability of block error when using said vectors. We use a *cycle code* because a complete characterization of their linear programming pseudocodewords is given in [1], which allows for easy access to a pool of known pseudocodewords.

Definition VI.1. A *cycle code* is a binary linear code $\mathcal{C}$ equipped with a parity-check matrix H with a uniform column weight of 2.

Given a parity-check matrix H of a cycle code, one can view H as the vertex-edge incidence matrix of a graph, the so-called *normal graph* N of the code. It is an easy exercise to show that a binary vector $\mathbf{c}$ is in the null space of H if and only if the support of $\mathbf{c}$ indexes an edge-disjoint union of cycles in N. For our example, we will consider the cycle code $\mathcal{C}_1$ of length 16 and dimension 5 presented as the null

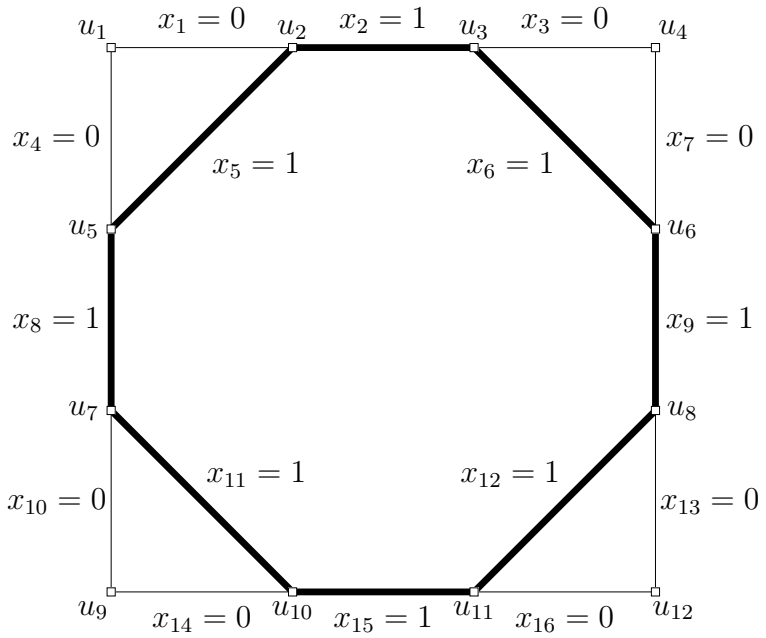


Fig. 3: The normal graph N_1 corresponding to the parity-check matrix H_1. The edges corresponding to the support of the codeword $\mathbf{c} = [0,1,0,0,1,1,0,1,1,0,1,1,0,0,1,0]^T$ are given in bold to emphasize the correspondence between edge-disjoint unions of cycles in the normal graph and codewords.

space of the parity-check matrix

$$H_1 = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

When viewed as the vertex-edge incidence matrix for a graph N_1 with vertices $u_1, u_2, \ldots, u_{12}$ and edges $x_1, x_2, \ldots, x_{16}$, we obtain the normal graph N_1 in Figure 3.

As mentioned before, cycle codes derive their name from the bijective correspondence between binary codewords and edge-disjoint unions of cycles in the normal graph. In [1], a graph-based characterization of both trivial and nontrivial LP pseudocodewords for cycle codes is given.

Theorem VI.2 ([1])**.** *Let C be a cycle code of length n presented by the parity-check matrix H, and let $\mathcal{P}$ and N denote its fundamental polytope and normal graph, respectively. A vector $\mathbf{x} \in \mathcal{P}$ is a linear programming pseudocodeword of $\mathcal{P}$ if and only if the following three conditions hold:*

(a) $\mathbf{x} \in \{0, \frac{1}{2}, 1\}^n$.

(b) With $\mathcal{H}_\mathbf{x}$ defined to be $\{i \,|\, x_i = \frac{1}{2}\}$, the subgraph Γ of N induced by the edges corresponding to the indices in $\mathcal{H}_\mathbf{x}$ is 2-regular. Equivalently, Γ is a union of vertex-disjoint simple cycles $\gamma_1, \gamma_2, \ldots, \gamma_\ell$.

(c) With $U^{\mathrm{o}}_{\gamma_i}$ defined as set of vertices in γ_i that are incident with an odd number of edges assigned a value of 1 by $\mathbf{x}$, the cardinality of $U^{\mathrm{o}}_{\gamma_i}$ is itself odd for all simple cycles γ_i in Γ.

Using Theorem VI.2, one can see that the vector $\boldsymbol{\omega}_1$ given by

$$\boldsymbol{\omega}_1 = \begin{bmatrix} \frac{1}{2} & 1 & 1 & \frac{1}{2} & \frac{1}{2} & 0 & 1 & 0 & 1 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{bmatrix}^T$$

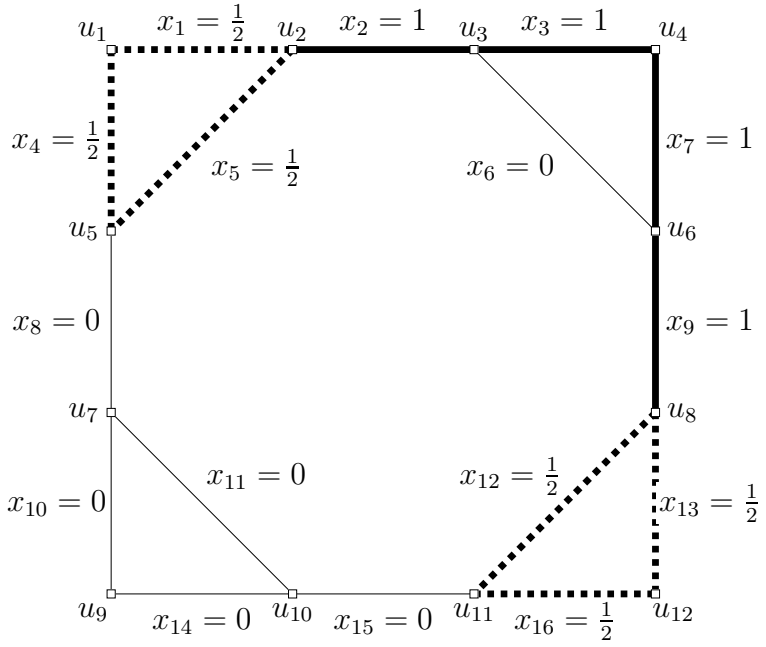


Fig. 4: A graphical representation of the pseudocodeword $\boldsymbol{\omega}_1$.

must be a linear programming pseudocodeword - Figure 4 gives a graphical representation. We note that, loosely speaking, $\boldsymbol{\omega}_1$ consists of a pair of cycles along with a path linking these cycles. The edges on the cycles are all labeled with $\frac{1}{2}$, and those on the path between the cycles carry a value of 1. A similar structure holds for all other pseudocodewords of cycle codes, and this structure is exploited by the *cycle-path method* of [1], allowing for the explicit construction of LP pseudocodewords for cycle codes.

We now turn our attention to the task of modulating pseudocodewords; that is, mapping them to vectors suitable for transmission. Unless stated otherwise the terms "modulated vector" and "modulated pseudocodeword" are taken to mean the center of an insphere for an appropriately chosen approximation cone.

The modulated vector $t(\boldsymbol{\omega}_1)$ suitable for transmitting $\boldsymbol{\omega}_1$ across the additive white Gaussian noise channel was found by a randomized, iterative procedure. The set of pseudocodewords Ψ_1 was initialized as the empty set and the vector $\mathbf{f}_1$ was set to be a conic combination of constraint vectors of $\mathcal{P}$ that were active at $\boldsymbol{\omega}_1$. From here, each iteration consisted of the following steps. Gaussian noise was repeatedly added to $\mathbf{f}_i$ and the resulting vectors were fed into the LP decoder. The pseudocodewords obtained as output vectors were recorded, and the unique polyhedral neighbors[3] of $\boldsymbol{\omega}_1$ were added to the set Ψ_i. Following the algorithm outlined in Section IV, the inradius of $\mathcal{R}_{\boldsymbol{\omega}_1,\Psi_i}$ was computed to within 10^{-5}. The conic combination $\mathbf{f}_{i+1}$ of constraints used for the next iteration was then updated to be the center of the insphere of $\mathcal{R}_{\boldsymbol{\omega}_1,\Psi_i}$, and the set of pseudocodewords Ψ_{i+1} was initialized at Ψ_i.

Below is the center of the insphere for $\mathcal{R}_{\boldsymbol{\omega}_1,\Psi_{10}}$, recorded to two decimal places for ease of viewing:

$$t(\boldsymbol{\omega}_1) = \begin{bmatrix} 0 & -0.35 & -0.25 & 0 & 0 & 0.35 & -0.26 & 0.35 & -0.35 & 0.26 & 0.35 & 0 & 0 & 0.26 & 0.35 & 0 \end{bmatrix}^T.$$

The corresponding inradius was 0.49; however, we note that since this is the inradius of an approximation cone and not that of the true recovery cone it only serves to provide an upper bound on the true inradius. We may therefore use such inradii only as rough gauges of error-correcting performance. For a concrete measure we rely on simulations (see Section VII).

[3]The pseudocodeword $\boldsymbol{\psi}$ is a polyhedral neighbor of $\boldsymbol{\omega}_1$ if both $\boldsymbol{\psi}$ and $\boldsymbol{\omega}_1$ comprise the two end points of one of the fundamental polytope's edges. This occurs precisely when the rank of constraint vectors that are active at *both* vectors is one less than the dimension of the ambient space. We restricted our sets Ψ_i to such neighboring pseudocodewords since they are the most likely erroneous outputs of the LP decoder in low-noise scenarios.

Following Section V, we can now apply $\mathcal{C}$-symmetry to $\boldsymbol{\omega}_1$ to obtain additional pseudocodewords as well as to $t(\boldsymbol{\omega}_1)$ to obtain modulated versions of these pseudocodewords. Using $\mathcal{C}$-symmetry about the codeword $\mathbf{c}$ of Figure 3 we obtain

$$\boldsymbol{\omega}_1^{\mathbf{c}} = \begin{bmatrix} \frac{1}{2} & 0 & 1 & \frac{1}{2} & \frac{1}{2} & 1 & 1 & 1 & 0 & 0 & 1 & \frac{1}{2} & \frac{1}{2} & 0 & 1 & \frac{1}{2} \end{bmatrix}^T$$

and

$$t(\boldsymbol{\omega}_1^{\mathbf{c}}) = \begin{bmatrix} 0 & 0.35 & -0.25 & 0 & 0 & -0.35 & -0.26 & -0.35 & 0.35 & 0.26 & -0.35 & 0 & 0 & 0.26 & -0.35 & 0 \end{bmatrix}^T.$$

There are more pseudocodewords in the $\mathcal{C}$-symmetry orbit of $\boldsymbol{\omega}_1$; however, there are only $2^3 = 8$ in total. This can be seen by observing that the operation of $\mathcal{C}$ symmetry will never affect any coordinate of a pseudocodeword which is assigned a value of $\frac{1}{2}$. Since the locations where $\boldsymbol{\omega}_1$ is equal to $\frac{1}{2}$ consists of two vertex disjoint cycles in the normal graph we can conclude that $\boldsymbol{\omega}_1$'s stabilizer subgroup has size $2^2 = 4$ and thus its orbit has size $2^5/2^2 = 8$.

Applying Theorem VI.2 and the cycle-path method of [1], we see that the linear programming pseudocodewords of this cycle code can be partitioned into 8 disjoint orbits under $\mathcal{C}$ symmetry: 7 orbits of nontrivial pseudocodewords and a separate orbit for the binary codewords. A set of representatives for these orbits is given as follows:

$$\begin{aligned}
\boldsymbol{\omega}_1 &= \begin{bmatrix} \frac{1}{2} & 1 & 1 & \frac{1}{2} & \frac{1}{2} & 0 & 1 & 0 & 1 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{bmatrix}^T \\
\boldsymbol{\omega}_2 &= \begin{bmatrix} 1 & 1 & \frac{1}{2} & 1 & 0 & \frac{1}{2} & \frac{1}{2} & 1 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 \end{bmatrix}^T \\
\boldsymbol{\omega}_3 &= \begin{bmatrix} \frac{1}{2} & 1 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \\
\boldsymbol{\omega}_4 &= \begin{bmatrix} \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 1 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 \end{bmatrix}^T \\
\boldsymbol{\omega}_5 &= \begin{bmatrix} 0 & 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 1 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} \end{bmatrix}^T \\
\boldsymbol{\omega}_6 &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & 1 & \frac{1}{2} \end{bmatrix}^T \\
\boldsymbol{\omega}_7 &= \begin{bmatrix} \frac{1}{2} & 1 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & 0 & 0 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & 1 & \frac{1}{2} \end{bmatrix}^T \\
\boldsymbol{\omega}_8 &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T.
\end{aligned}$$

The sizes of these orbits are 8, 8, 8, 8, 8, 8, 2, and 32, respectively, making for a total of 82 pseudocodewords.

Modulated versions of $\boldsymbol{\omega}_2, \boldsymbol{\omega}_3, \ldots, \boldsymbol{\omega}_8$ were found in a fashion similar to that described above for $\boldsymbol{\omega}_1$. The inradii of the corresponding approximation cones were 0.49, 0.43, 0.43, 0.43, 0.43, 0.32, and 0.50, respectively – the apparent multiplicity of some of these inradii is due to the rotational symmetry of the normal graph. It is interesting to note that the modulated version of $\boldsymbol{\omega}_8$, which is simply the all-zeros codeword, is not proportional to the vector of all ones, which is the vector used to transmit the all-zeros codeword under BPSK modulation. To two decimal places, we have

$$t(\boldsymbol{\omega}_8) = \begin{bmatrix} 0.27 & 0.04 & 0.28 & 0.28 & 0.31 & 0.31 & 0.27 & 0.04 & 0.04 & 0.27 & 0.31 & 0.31 & 0.27 & 0.27 & 0.04 & 0.28 \end{bmatrix}^T.$$

With respect to the approximation cone used to produce $t(\boldsymbol{\omega}_8)$, the radius of the largest sphere centered at $t(\boldsymbol{\omega}_8)$ and contained within the approximation cone was 0.50. The radius of the largest sphere centered at the unit vector in the direction of the BPSK vector and contained within the same approximation cone was only 0.43, which suggests that the use of $t(\boldsymbol{\omega}_8)$ will result in a lower block error rate than traditional BPSK modulation. This suspicion is confirmed by the simulation results displayed in Figure 5.

Figure 5 reveals some interesting facts. Most basically, pseudocodewords can be reliably recovered when insphere modulation is used for transmission. Second, pseudocodewords in different $\mathcal{C}$-symmetry orbits display a wide range of error-correcting ability: $\boldsymbol{\omega}_7$ has terrible block error rates, while $\boldsymbol{\omega}_1$'s error rates are very close to the error rates provided by BPSK modulation for codewords. Finally, for low levels of noise we note that using insphere modulation for codewords results in slightly lower error rates than BPSK modulation. As we will see in Section VII, it is often the case where codewords themselves can benefit by exchanging BPSK modulation for insphere modulation.

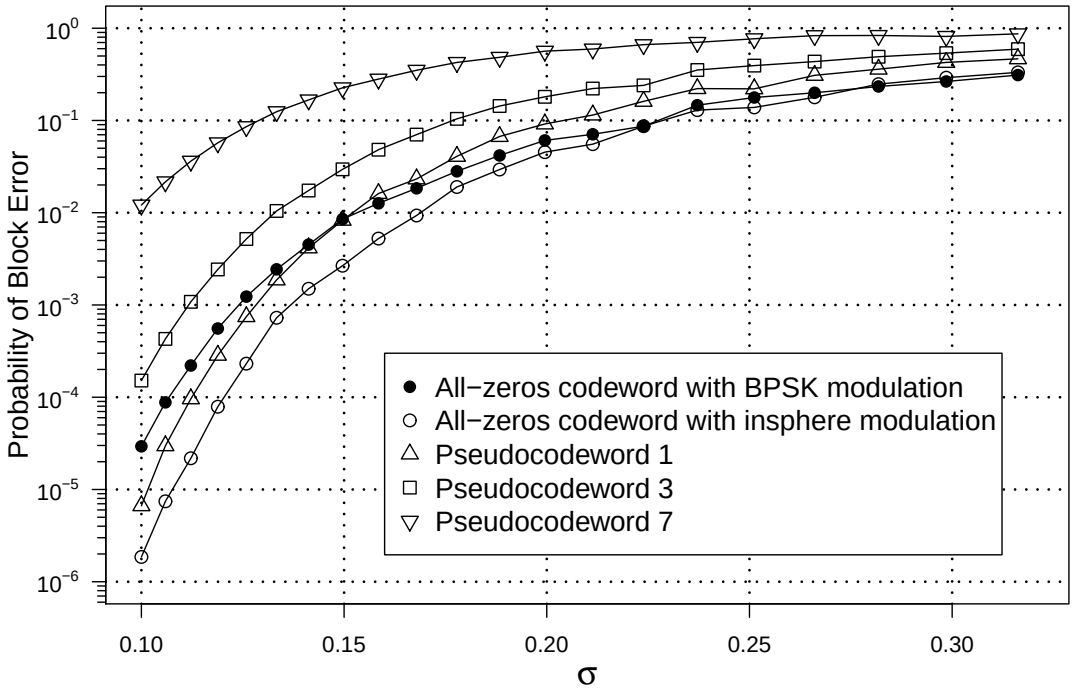


Fig. 5: A comparison of block error rates when transmitting the all-zeros codeword, $\boldsymbol{\omega}_1, \boldsymbol{\omega}_3$, and $\boldsymbol{\omega}_7$ with insphere modulation. Also included is the block error rate of the all-zeros codeword under traditional BPSK modulation. The horizontal axis measures the standard deviation of the additive white Gaussian noise.

VII. Performance Analysis

In Section V we saw how $\mathcal{C}$-symmetry can be leveraged to produce a codebook that is larger than the original binary code but still has the same length (i.e., the same number of real dimensions needed for its signal constellation), and a specific example was given in Section VI in which a cycle code of size 32 could be extended to a codebook consisting of as many as 82 different messages. This increase in spectral efficiency must be weighed against the potential loss of error-correcting performance. For the simulations in this section, we are concerned only with the probability of a block error.

We begin with some notation. Let $\mathcal{C}$ be a code presented by a parity-check matrix H, and let $\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_m$ denote m mutually disjoint orbits of LP pseudocodewords (trivial or nontrivial) under $\mathcal{C}$-symmetry. Suppose we wish to transmit information across the AWGN channel using the codebook $\mathcal{M} = \cup_{i=1}^{m} \mathcal{O}_i$. By Theorem V.5, we may assume that for a fixed noise level the probability of block error will be uniform across each orbit $\mathcal{O}_i$ – we denote this probability by p_i. It is of the utmost importance to observe that p_i is completely independent of the other orbits comprising the codebook $\mathcal{M}$. Indeed, under LP decoding *all* points in the fundamental polytope, and thus all LP pseudocodewords, are made available as potential explanations of the received vector regardless of whether these points were included in the codebook $\mathcal{M}$. Because of this independence, if we assume a uniform distribution on the $M = \sum_{i=1}^{m} |\mathcal{O}_i|$ pseudocodewords from $\mathcal{M}$ we then have that the probability of block error is $p = \frac{1}{M} \sum_{i=1}^{m} |\mathcal{O}_i| p_i$.

This formula for the overall probability of block error gives great flexibility to the researcher when searching through possible codebooks. First, we choose a set of linear programming pseudocodewords $\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_m$ from distinct $\mathcal{C}$-symmetry orbits and then compute modulated versions for each (see Sections III and IV). For each pseudocodeword chosen we simulate transmission across an additive white Gaussian noise channel, taking care to ensure that all modulated vectors have unit energy and that the same level of Gaussian noise is added to every pseudocodeword. Once simulation results are obtained,

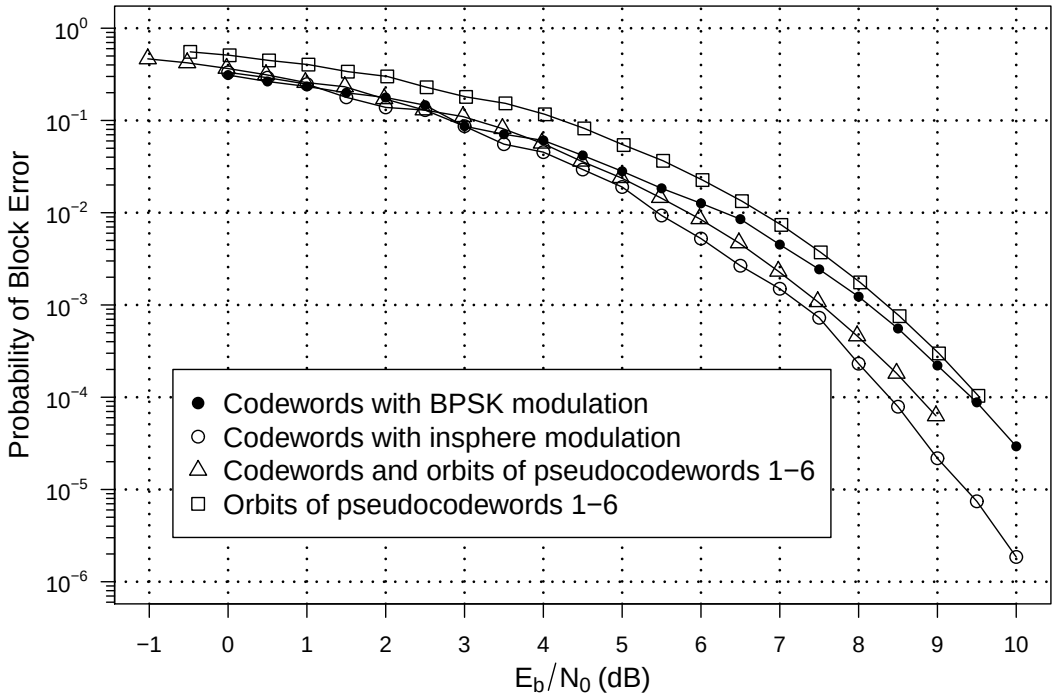


Fig. 6: A comparison of block error rates for four codebook/modulation pairs, all of which were derived from the cycle code $\mathcal{C}_1$ of Section VI.

we have approximations of all p_i's. From here we can chose any collection of the m mutually disjoint orbits represented by $\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_m$ to combine into an overall codebook whose probability of error is given above. The corresponding signal-to-noise ratios are computed according to the formula $\frac{E_b}{N_0} = \frac{1}{2\log_2(M)\sigma^2}$, where M is the size of the overall codebook and σ^2 the variance of the Gaussian noise added to the transmitted pseudocodewords.

A. The cycle code $\mathcal{C}_1$

For our first round of simulations we use the cycle code $\mathcal{C}_1$ from Section VI. In Figure 6 we compare the performance of four codebook/modulation schemes. Two of these use only the set of 32 codewords, but differ in that one utilizes BPSK modulation while the other employs insphere modulation. The third scheme's codebook is comprised of all 32 codewords in addition to the 48 nontrivial pseudocodewords in the $\mathcal{C}$-symmetric orbits of $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_6$, making for an overall codebook of size 80 (all vectors in this codebook were modulated using insphere modulation). The final combination listed is the codebook consisting only of the 48 nontrivial pseudocodewords in the $\mathcal{C}$-symmetric orbits of $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_6$ – no binary codewords whatsoever[4].

We can make several observations from Figure 6. First, we see that insphere modulation enjoys a modest coding gain over BPSK modulation when it comes to transmitting codewords. Second, we see that when pseudocodewords are broadcast in conjunction with codewords, it is possible to attain error rates better than those of codewords alone under BPSK modulation. Moreover, these better error rates are coupled with a higher spectral efficiency: the coding scheme consisting of codewords and the $\mathcal{C}$-symmetric orbits of $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_6$ has a spectral efficiency of $\log_2(80)/16 = 0.3951$, whereas the spectral efficiency of the codebooks using only binary codewords is 0.3125. The spectral efficiency of the codebook consisting only of the union of the $\mathcal{C}$-symmetry orbits of $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_6$ is $\log_2(48)/16 = .3491$, thus landing this

[4]The two pseudocodewords in the orbit of $\boldsymbol{\omega}_7$ were not used due to their relatively poor block error rates (see Figure 5).

codebook in a bit of gray area: when compared to codewords under BPSK modulation, it is slightly better with respect to spectral efficiency yet slightly worse with respect to block error rate.

B. The cycle code $\mathcal{C}_2$

A second [20, 7] cycle code $\mathcal{C}_2$ was also investigated. The parity-check matrix chosen for $\mathcal{C}_2$ corresponds to the normal graph shown in Figure 7. Using Theorem VI.2 and the cycle-path method of [1] one can show that this code possesses a total of 2,376 pseudocodewords: 128 binary codewords, $\binom{8}{4} = 70$ disjoint $\mathcal{C}$-symmetry orbits consisting of $2^5 = 32$ nontrivial pseudocodewords each, and a final orbit consisting of $2^3 = 8$ nontrivial pseudocodewords. It is therefore possible to construct a coding scheme from this code that possesses a spectral efficiency of $\log_2(2376)/20 = 0.5607$, a fair bit higher than 0.35, the spectral efficiency of the original binary code. Many of these non-trivial pseudocodewords, however, have approximation cones with small inradii, which would drive the block error rates of a coding scheme using all 2,376 pseudocodewords to unacceptably high levels. The task at hand, and indeed for any coding scheme we try to extend by using LP pseudocodewords, is to determine which of these non-trivial pseudocodewords enjoy a performance advantage over binary codewords under BPSK modulation.

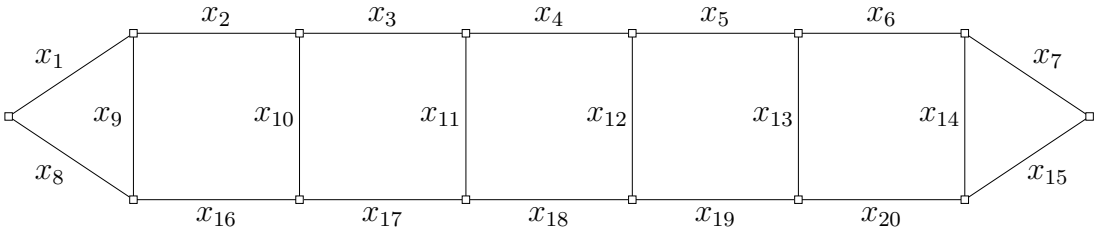


Fig. 7: The normal graph corresponding to the [20, 7] cycle code $\mathcal{C}_2$.

Many pseudocodewords were examined, and we present results on the following three for illustration:

$$\boldsymbol{\omega}_A = \left[\tfrac{1}{2}\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ \tfrac{1}{2}\ \ \tfrac{1}{2}\ \ \tfrac{1}{2}\ \ 0\ \ 0\ \ 0\ \ 0\ \ \tfrac{1}{2}\ \ \tfrac{1}{2}\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\right]^T$$
$$\boldsymbol{\omega}_B = \left[\tfrac{1}{2}\ \ 1\ \ 1\ \ 1\ \ \tfrac{1}{2}\ \ 0\ \ 0\ \ \tfrac{1}{2}\ \ \tfrac{1}{2}\ \ 0\ \ 0\ \ \tfrac{1}{2}\ \ \tfrac{1}{2}\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ \tfrac{1}{2}\ \ 0\right]^T$$
$$\boldsymbol{\omega}_C = \left[0\ \ \tfrac{1}{2}\ \ \tfrac{1}{2}\ \ 1\ \ \tfrac{1}{2}\ \ \tfrac{1}{2}\ \ 0\ \ 0\ \ \tfrac{1}{2}\ \ 0\ \ \tfrac{1}{2}\ \ \tfrac{1}{2}\ \ 0\ \ \tfrac{1}{2}\ \ 0\ \ \tfrac{1}{2}\ \ \tfrac{1}{2}\ \ 0\ \ \tfrac{1}{2}\ \ \tfrac{1}{2}\right]^T.$$

Modulated versions of these pseudocodewords as well as for the all-zeros codeword were found in a manner similar to that outlined for $\mathcal{C}_1$ in Section VI, with the inradius of each approximation cone computed to within 10^{-5}. To two decimal places, the inradii for $\boldsymbol{\omega}_A, \boldsymbol{\omega}_B, \boldsymbol{\omega}_C$, and the all-zeros codeword were 0.46, 0.39, 0.21, and 0.45, respectively. The radius of the largest sphere centered at the unit vector in the direction of the vector used to transmit the all-zeros codewords under BPSK modulation and contained within the approximation cone developed for the all-zeros codeword was only 0.39. These inradii suggest that $\boldsymbol{\omega}_A$ will achieve the best error rates, followed by codewords broadcast with insphere modulation, a near-tie between $\boldsymbol{\omega}_B$ and the all-zeros codeword under BPSK modulation, and finally $\boldsymbol{\omega}_C$. These predictions based on inradii of approximation cones are only partly born out in simulation - see Figure 8. Still, it is interesting to observe that insphere modulation of the all-zeros codeword as well as of $\boldsymbol{\omega}_A$ beat out BPSK modulation at moderate to low levels of Gaussian noise.

We present various coding schemes based off of $\mathcal{C}_2$ and its pseudocodewords in Figure 9. The spectral efficiencies of the both schemes utilizing only codewords is 0.35. The spectral efficiency of the scheme whose codebook consists of both codewords as well as pseudocodewords in the orbit of $\boldsymbol{\omega}_A$ is $\log_2(128 + 32)/20 = 0.3661$, while that of the scheme whose codebook consists of codewords as well as pseudocodewords in the orbits of $\boldsymbol{\omega}_A$ and $\boldsymbol{\omega}_B$ is $\log_2(128 + 32 + 32)/20 = 0.3792$. All of the coding

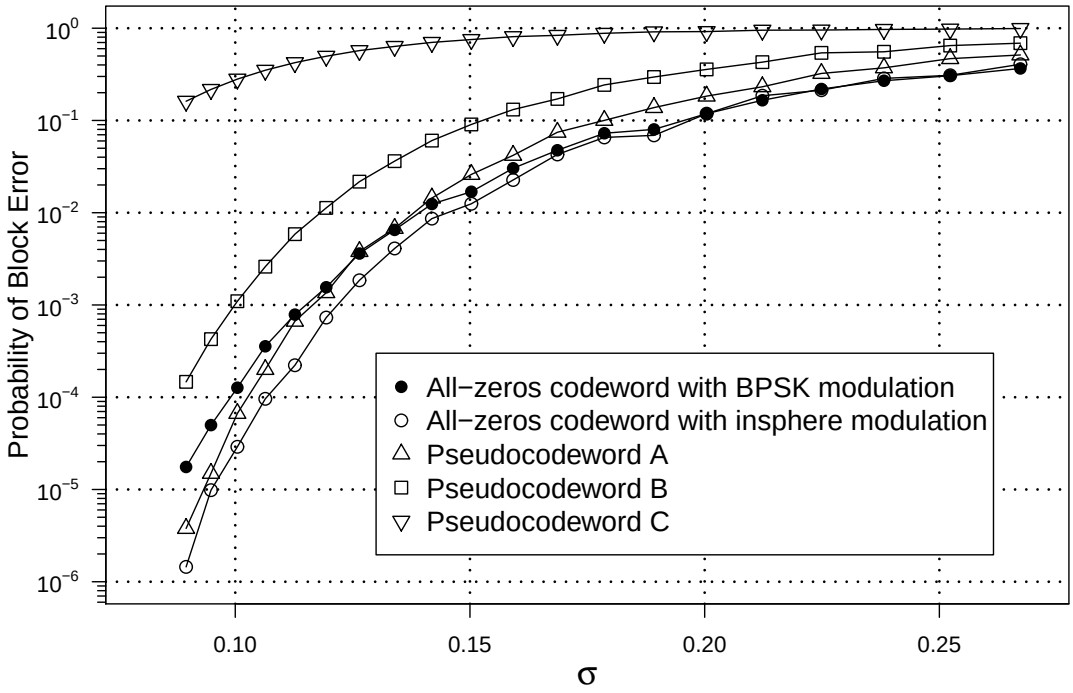


Fig. 8: A comparison of block error rates when transmitting the all-zeros codeword, $\boldsymbol{\omega}_A, \boldsymbol{\omega}_B$, and $\boldsymbol{\omega}_C$ with insphere modulation. Also included is the block error rate of the all-zeros codeword under traditional BPSK modulation. The horizontal axis measures the standard deviation of the additive white Gaussian noise.

schemes utilizing insphere modulation have spectral efficiencies that meet or exceed that of the traditional BPSK modulation, and they do so while performing better with respect to block error rate (see Figure 9). It is also interesting that the coding scheme consisting of codewords as well as pseudocodewords in the orbit of $\boldsymbol{\omega}_A$ performs *slightly* better than codewords under insphere modulation while also enjoying a higher spectral efficiency. This is encouraging, for it gives us hope that incorporating pseudocodewords into coding schema may be able to produce results that extend beyond those obtained by simply optimizing the manner in which we transmit binary codewords.

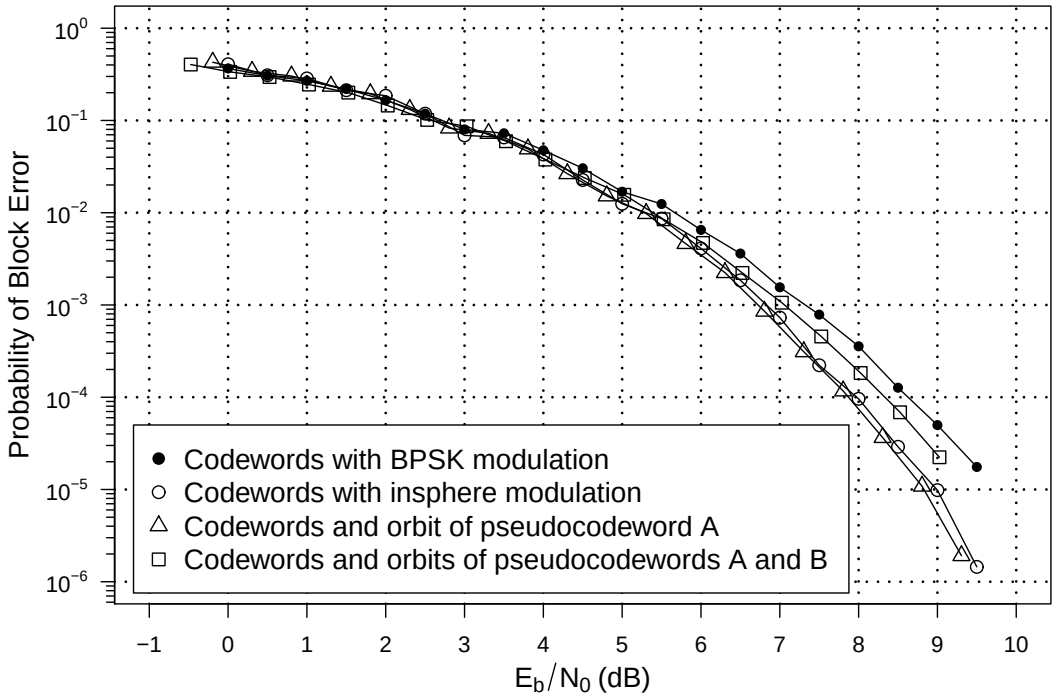


Fig. 9: A comparison of block error rates for four codebook/modulation pairs derived from the cycle code $\mathcal{C}_2$.

C. A Randomly Generated [30, 10] Code

In order to branch out from cycle codes we randomly generated a $[30, 10]$ code $\mathcal{C}_3$ with the following parity-check matrix:

$$H_3 = \begin{bmatrix}
0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0\\
0&0&1&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&1&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&1
\end{bmatrix}.$$

Pseudocodewords for this code/parity-check matrix combination were generated via a random search. The two we highlight are given below:

$$\boldsymbol{\omega}_D = \begin{bmatrix}0&1&0&0&\frac{1}{3}&0&1&\frac{2}{3}&0&0&\frac{1}{3}&\frac{1}{3}&0&1&\frac{1}{3}&\frac{1}{3}&1&0&0&1&1&\frac{1}{3}&1&1&\frac{2}{3}&\frac{2}{3}&1&1&1&1\end{bmatrix}^T$$

$$\boldsymbol{\omega}_E = \begin{bmatrix}1&\frac{4}{5}&0&1&0&1&0&\frac{3}{5}&1&1&1&0&1&0&1&0&\frac{1}{5}&\frac{3}{5}&\frac{3}{5}&\frac{3}{5}&0&1&\frac{1}{5}&1&1&\frac{1}{5}&0&\frac{1}{5}&1&1\end{bmatrix}^T.$$

Since neither $\boldsymbol{\omega}_D$ nor $\boldsymbol{\omega}_E$ contains any fractional entries equal to $\frac{1}{2}$, each has a $\mathcal{C}$-symmetry orbit of the same size as the binary code itself. The codebooks that included these nontrivial pseudocodewords,

however, all had much higher error rates than the binary codewords. Figure 10 shows the performance of $\boldsymbol{\omega}_D$ and $\boldsymbol{\omega}_E$ as compared to the performance of the all-zeros codeword under both BPSK and insphere modulation. The inradii of the approximation cones used to produce transmittable versions of $\boldsymbol{\omega}_D$ and $\boldsymbol{\omega}_E$ were 0.24 and 0.20, respectively. The corresponding inradii for the all-zeros codeword under insphere modulation was, by comparison, 0.43. The gap between these numbers is likely the reason for the marked difference in block error rates observed in Figure 10. Since none of the nontrivial pseudocodewords we found came close to matching the performance of codewords, we were unable to increase the spectral efficiency of this coding scheme without greatly sacrificing performance - see Figure 11.

As mentioned before, the inradius of the approximation cone used to produce a transmittable version of the all-zeros codeword was 0.43. The largest sphere contained within this same cone but centered at the unit vector in the direction of the BPSK vector was only 0.26. This suggests that insphere modulation of codewords should display lower block error rates than those of codewords under BPSK modulation. As we see in Figure 10, this is indeed the case. A potential explanation of this improvement in performance is that insphere modulation gives a method of identifying individual bit locations that are particularly susceptible to errors and "boosting" them, while in turn lowering the energy used to transmit other bits to balance things out. For instance, we note that $\mathcal{C}_3$ has a codeword of Hamming weight 2 whose support lies only in the first and tenth positions. The unit vector used to transmit the all-zeros vector under insphere modulation assigns values of 0.3048 and 0.3065 to the first and tenth positions, respectively. These values are significantly larger than $1/\sqrt{30} = 0.1826$, which is the value that the unit vector in the direction of the BPSK vector assigns uniformly to all positions.

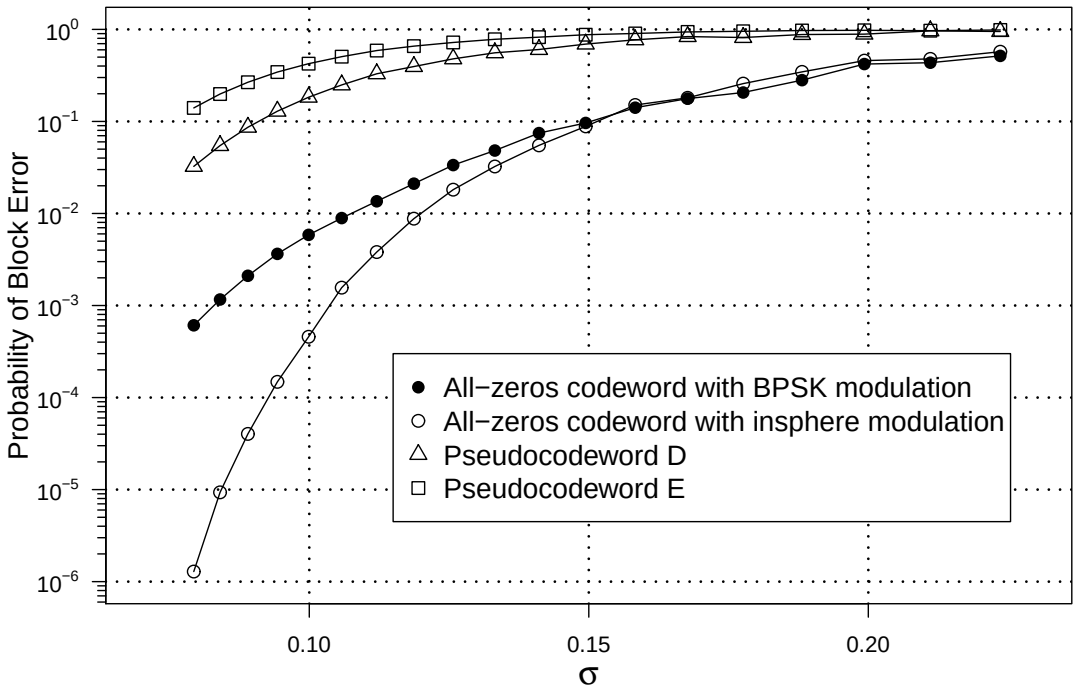


Fig. 10: A comparison of block error rates for the randomly generated [30, 10] code when transmitting the all-zeros codeword and two nontrivial pseudocodewords $\boldsymbol{\omega}_D$ and $\boldsymbol{\omega}_E$ with insphere modulation. Also included is the block error rate of the all-zeros codeword under traditional BPSK modulation. The horizontal axis measures the standard deviation of the additive white Gaussian noise.

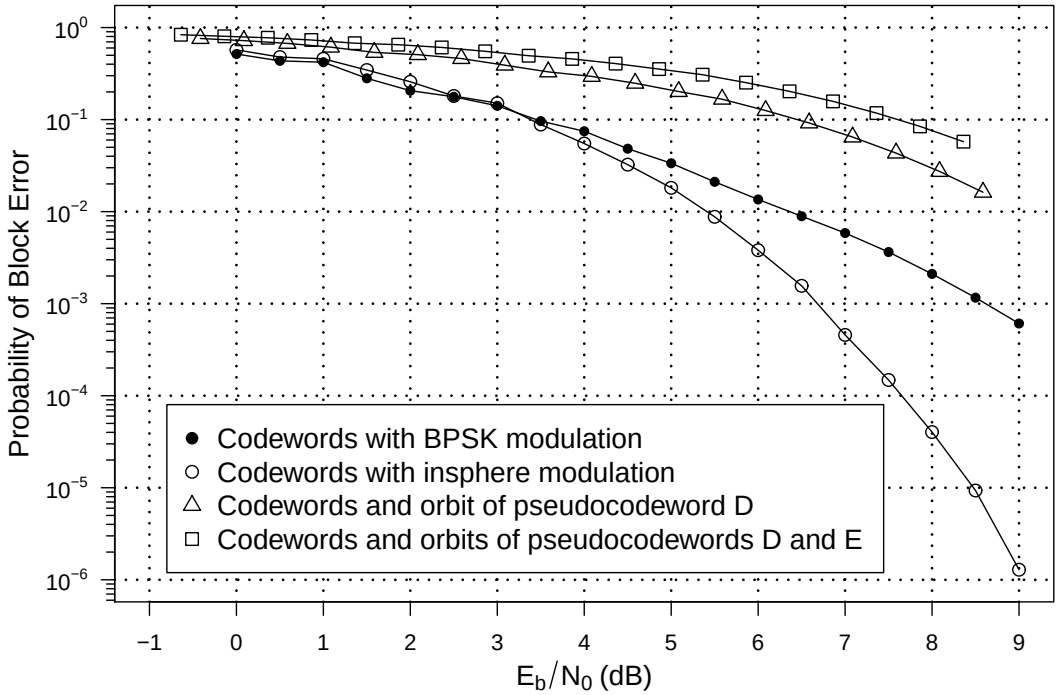


Fig. 11: A comparison of block error rates for four codebook/modulation pairs derived from the code $\mathcal{C}_3$. Insphere modulation is used for all elements of a codebook unless explicitly noted otherwise.

D. A Randomly Generated [30, 17] Code

In the past three examples, insphere modulation applied to codewards has always outperformed binary phase shift keying at higher signal-to-noise ratios. In this section, we give a cautionary example to demonstrate that this is not always the case.

Consider the $[30, 17]$ code $\mathcal{C}_4$ presented by the following parity-check matrix:

$$H_4 = \begin{bmatrix}
1&0&1&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&1&0&1&0&0&1&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&1&0\\
0&0&0&1&0&0&0&0&1&0&0&0&0&1&1&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0\\
0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&0&0&0&1&1&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&1&1\\
0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&0&1&1&0&0&0&0&0&1&0&0\\
0&1&0&0&0&0&0&0&1&0&1&1&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0\\
0&0&1&0&1&1&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&1&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&1&1&0&0&0&0\\
0&0&0&0&1&0&0&0&0&1&0&0&1&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&1&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&1&0&0&1&0&0&1
\end{bmatrix}.$$

All pseudocodewords tested for this code performed abysmally - the best was still giving block error rates above 0.1 at 8.5 decibels. Moreover, as we can see in Figure 12 the performance of insphere modulation and BPSK modulation were virtually identical. Thus, no real gains in either spectral efficiency or block error rate can be gained by using insphere modulation in conjunction with this code.

VIII. Conclusion

In this paper we have defined insphere modulation, a modulation scheme that is suitable for transmitting nontrivial linear programming pseudocodewords in addition to binary codewords. In doing so, we have developed a method for approximating the insphere of a polyhedral cone to any degree of accuracy. Simulation results confirm that certain codebooks utilizing a combination of nontrivial LP pseudocodewords

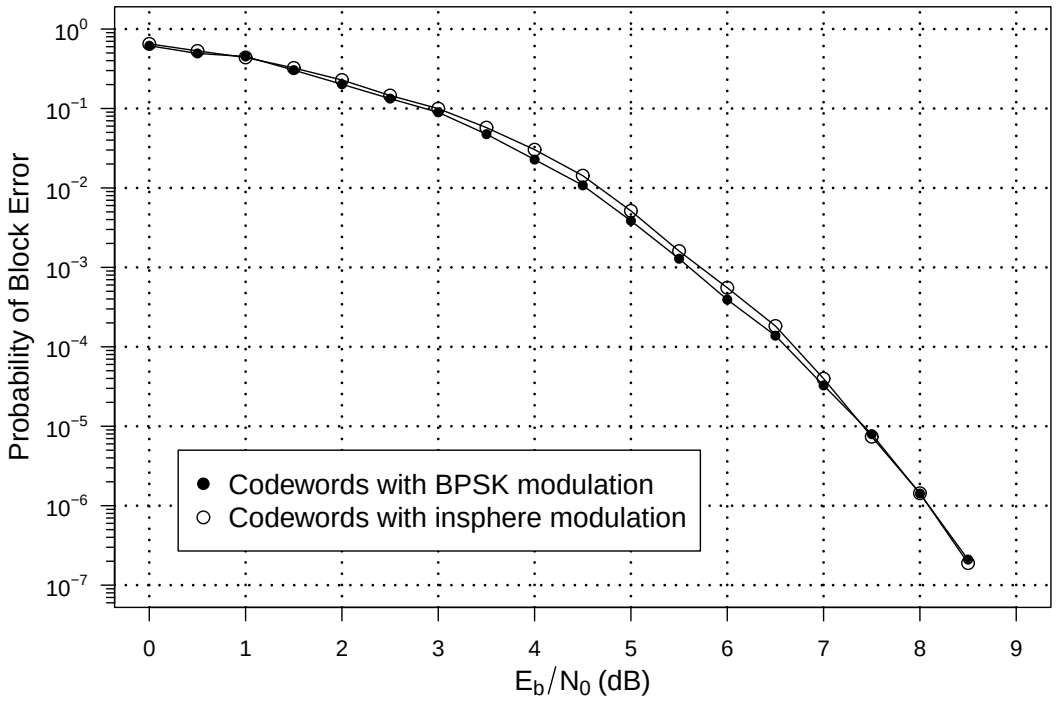


Fig. 12: A comparison of block error rates for the randomly generated $[30, 17]$ code under both BPSK and insphere modulation.

in conjunction with codewords give modest gains in spectral efficiency without sacrificing error correcting performance. While originally introduced as a means to transmit nontrivial pseudocodewords, insphere modulation can in some instances be used to lower the block error rates of binary codewords relative to their performance under traditional BPSK modulation.

It is the author's belief that the characterization of linear programming pseudocodewords for cycle codes of [1] makes it easier for the researcher to search through the vast number of pseudocodewords and possibly stumble upon some that can be beneficial. A broader characterization of LP pseudocodewords for general binary codes, should it exist, may prove useful in determining the overall efficacy of using hybrid codebooks comprised of both binary codewords and nontrivial pseudocodewords.

REFERENCES

[1] N. Axvig and D. Dreher. Graphical characterizations of linear programming pseudocodewords for cycle codes. *IEEE Transactions on Information Theory*, 59(9):5917–5934, September 2013.

[2] N. Axvig, D. Dreher, K. Morrison, E. Psota, L.C. Pérez, and J.L. Walker. Average min-sum decoding of LDPC codes. In *Turbo Codes and Related Topics, 2008 5th International Symposium on*, pages 356–361, September 2008.

[3] N. Axvig, D. Dreher, K. Morrison, E. Psota, L.C. Pérez, and J.L. Walker. Analysis of connections between pseudocodewords. *IEEE Transactions on Information Theory*, 55(9):4099–4107, September 2009.

[4] D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1997.

[5] J. Feldman. *Decoding Error-Correcting Codes via Linear Programming*. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 2003.

[6] J. Feldman, M.J. Wainwright, and D.R. Karger. Using linear programming to decode binary linear codes. *IEEE Transactions on Information Theory*, 51(3):954–972, March 2005.

[7] R. Henrion and A Seeger. On properties of different notions of centers for convex cones. *Set-Valued and Variational Analysis*, 18(2):205–231, June 2010.

[8] C. Kelley and D. Sridhara. Eigenvalue bounds on the pseudocodeword weight of expander codes. *Adv. Math. Commun.*, 1(3):287–306, 2007.

[9] C. Kelley and D. Sridhara. Pseudocodewords of Tanner graphs. *IEEE Transactions on Information Theory*, 53:4013–4038, November 2007.

[10] R. Koetter, W.-C.W. Li, P.O. Vontobel, and J.L. Walker. Characterizations of pseudo-codewords of (low-density) parity-check codes. *Advances in Mathematics*, 213:205–229, 2007.

[11] M. Orlowski and M. Pachter. Linear programming in $\mathbb{R}^3$ and the skeleton and largest incircle of a convex polygon. *Computers and Mathematics with Applications*, 13(4):401–405, 1987.

[12] R. Smarandache and P.O. Vontobel. Pseudo-codeword analysis of Tanner graphs from projective and euclidean planes. *IEEE Transactions on Information Theory*, 53(7):2376–2393, July 2007.

[13] P.O. Vontobel and R. Koetter. Graph-cover decoding and finite-length analysis of message-passing iterative decoding of LDPC codes. CoRR, http://arxiv.org/abs/cs/0512078, December 2005.

[14] S.-T. Xia and F.-W. Fu. Minimum pseudo-codewords of LDPC codes. In *Proceedings of the 2006 IEEE Information Theory Workshop*, pages 109–113, Chengdu, China, October 2006.

[15] J. Zumbrägel, V. Skachek, and M.F. Flanagan. On the pseudocodeword redundancy of binary linear codes. *IEEE Transactions on Information Theory*, 58(7):4848–4861, July 2012.