N94-16509

Search for Optimal Distance Spectrum Convolutional Codes *

Matthew C. Connor
Lance C. Perez
Daniel J. Costello, Jr.

Department of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

Presented at the 4^{th} Annual Argonne Symposium for Undergraduates in Science, Engineering, and Mathematics

Argonne National Laboratory

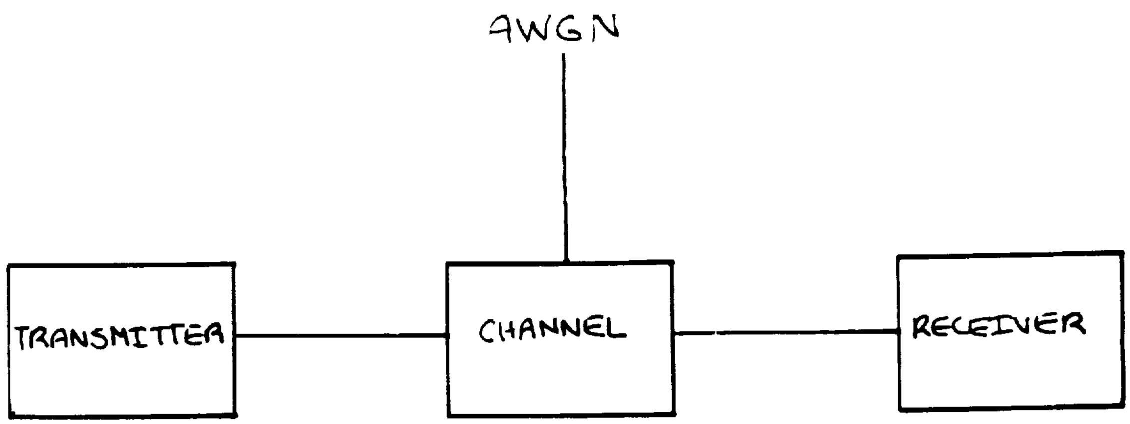
6 November 1993

*This work was supported in part by NASA Grant NAG5-557 and NSF Grant NCR89-03429

Introduction

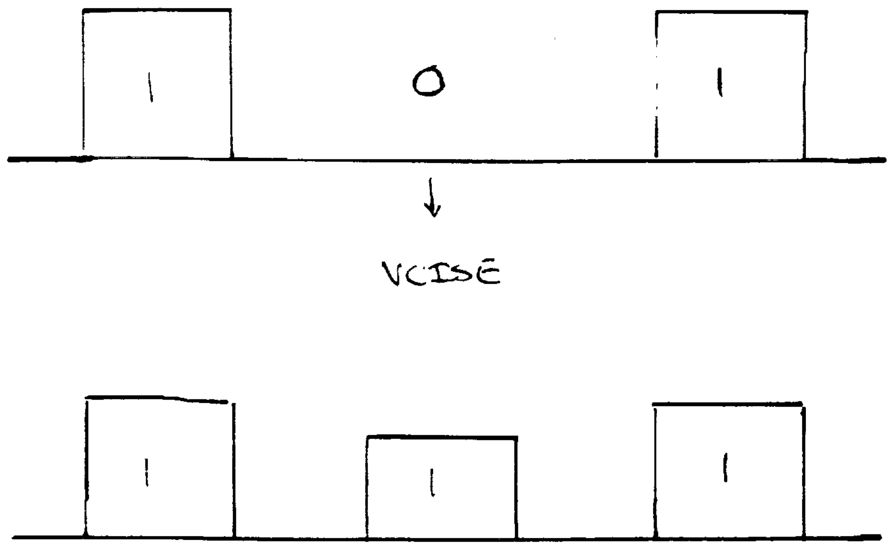
- In order to communicate reliably and to reduce the required transmitter power, NASA uses coded communication systems on most of their deep space satellites and probes (e.g. Pioneer, Voyager, Galileo, and the TDRSS network).
- These communication systems use binary convolutional codes. Better codes make the system more reliable and require less transmitter power.
- However, there are no good construction techniques for convolutional codes. Thus, to find good convolutional codes requires an exhaustive search over the ensemble of all possible codes.
- In this paper, an efficient convolutional code search algorithm was implemented on an IBM RS6000 Model 580. The combination of algorithm efficiency and computational power enabled us to find, for the first time, the optimal rate 1/2, memory 14, convolutional code.

Digital Transmission Over a Noisy Channel



- When binary digital data is transmitted over a real channel, it is subject to noise (we will assume Additive White Gaussian Noise). The noise can cause errors to occur at the receiver.
- The acceptable bit-error-rate (BER) at the receiver depends on the type of data being transmitted. For example, video signals are more forgiving of errors than computer data.
- One of the goals of forward error correction (FEC) coding, is to allow the receiver to correct errors caused by the channel and thus to increase the reliability of the system and/or reduce the required signal energy.

Example (Binary Numbers)



- To transmit a 5 in binary, the codeword 101 would be sent. This sequence of transmitted bits is then subject to channel noise.
- If the channel noise is large enough relative to the transmitted signal energy (per bit), the receiver may interpret a transmitted 1 as a 0, or vice-versa.
- For example, an optimum receiver would interpret the received signal shown above as 111 or 8. In this case, the receiver makes one error which in turn causes one codeword, 101, to be converted into another codeword, 111.
- The probability that a 1 is received as a 0 and vice versa is called the channel transition probability, p, and is a function of the signal-to-noise ratio (SNR),

$$SNR = \frac{E_S}{N_0}$$

where E_S is the average transmitted signal energy per bit and N_0 is the one sided noise spectral density (a measure of the noise power).

Example (cont.)

Given a channel transition probability of p, the probability that 101 is transmitted and 111 is received is given by

$$P_{101,111} = P_1 = (1-p)^2 p$$

This probability can be reduced by increasing the SNR which in turn causes a reduction in p.

In this example, one bit error causes one codeword to be converted into another codeword. We say that this code has *minimum free Hamming distance*, of $d_f = 1$.

000	100
001	101
010	110
011	111

In general, each codeword in this code may be converted into 3 different codewords by a single bit error with probability P_1, 3 different codewords by two bit errors with probability

$$P_2 = (1-p)p^2 < P_1$$

and one other codeword with three bit errors with probability

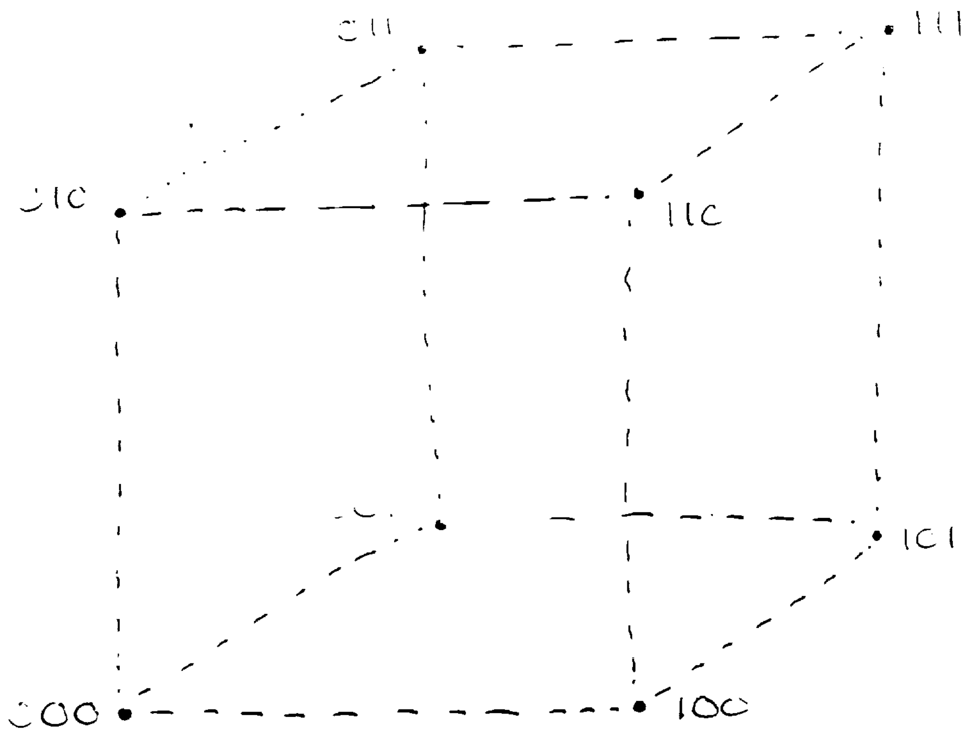
$$P_3 = (1-p)p^2 < P_2$$

The overall *probability of codeword error* is

$$P_C = 3P_1 + 3P_2 + 1P_3$$

which can be reduced by increasing the SNR.

Geometric Interpretation and Hamming Distance



- Intuitive insight into the error mechanism can be obtained using a geometric perspective.
- From this point of view, each codeword in the previous example is considered a vector in a 3-dimensional vector space. The distance between two vectors is the *Hamming Distance*, d_H, which is just the number of positions in which two vectors differ.
- The probability that a codeword is converted into another codeword at a Hamming distance of d is

$$P_d = (1-p)^{3-d}p^d$$

- Notice, that as the Hamming distance between two codewords increases P_d decreases!
- For a fixed SNR and thus a fixed channel transition probability, p, the probability of a codeword error can be reduced by increasing the Hamming distance between all pairs of codewords.

Example: Repetition Coding

- A simple coding technique is known as repetition coding. In this scheme each bit is simply transmitted twice in succession.
- Continuing the previous example, repetition coding leads to the following set of codewords.

000000	110000
000011	110011
001100	111100
001111	111111

- The mininum free Hamming distance is now $d_f = 2$ and the overall probability of codeword error is

$$P'_C = 3P_2 + 3P_4 + 1P_6 < P_C,$$

because each codeword has 3 codewords at distance 2, 3 codewords at distance 4, and 1 codeword at distance 6.

- The enumeration of the distances between one codeword and all other codewords in the code is called the code *distance spectrum* and is usually depicted in the following way

d	2	3	4	5	6
N_d	3	0	3	0	1

Coding Performance Tradeoffs

- The probability of codeword error in digital communications systems on the AWGN channel is determined primarily by three factors:
  1. SNR,
  2. d_f, the code's minimum distance, and
  3. the code's distance spectrum.
- Historically, due to the expense of putting large power supplies in space and the relatively large amount of available bandwidth, NASA has chosen to improve system performance by using coding and expanding the required transmission bandwidth.
- With most of its satellites and deep space probes, NASA has chosen to use convolutional codes because of their superior performance characteristics in this application.

Convolutional Codes

- A binary linear convolutional code with rate k/n is a set of semi-infinite sequences generated by a finite state machine characterized by three parameters:

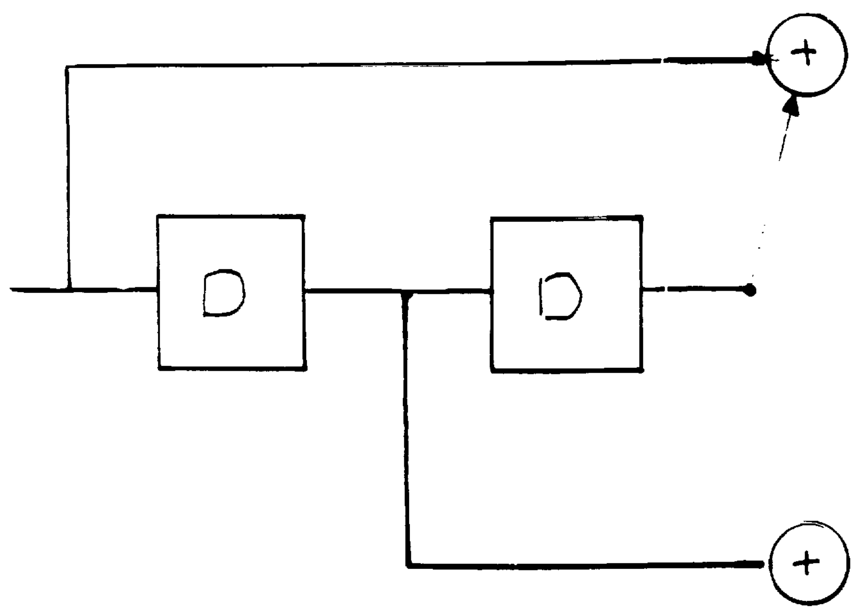
  1. k, the number of inputs bits per encoding interval,
  2. n, the number of output bits per encoding interval,
  3. m, the memory order of the finite state machine.

  The finite state machine has 2^m states.

- During each encoding interval, an (n, k, m) convolutional code encodes k information bits into n bits based on the current block of k bits and the past m blocks of k bits.

- The minimum distance between codewords and thus the performance of a convolutional code increases as the rate decreases and the memory increases.

A (2,1,2) Convolutional Code



- A rate 1/2, convolutional code is specified by a pair of generators denoted by $(g1, g2)$ that describe the connections from the shift register to the output.
- The (2,1,2) code shown above has generators

$$\begin{aligned} g1 &= 101 = 5 \\ g2 &= 010 = 2 \end{aligned}$$

Optimal Distance Spectrum Codes

- The maximum free distance of a (2,1,14) code is known to be 18. Many good codes have been found that have this free distance. The goal of this research was to find the "best" rate 1/2, memory 14 convolutional code with free distance 18.
- One way to do this is by finding the distance spectrum of every possible code.
- Those codes with fewer paths at a given distance have a lower probability of error, and thus are considered better. If the number of paths are recorded for each code having a minimum free distance of 18, the list could then be sorted and the best code found.
- For example, the maximal free distance (2,1,14) code with generators $(g1, g2)$ =(56721,61713) has 33 paths of weight 18. If another (2,1,14) code with fewer weight 18 paths could be found, this code would be a better code.

The Problem with Finding Optimal Codes

- Finding the optimal code would be easy if the all of the codes' distance spectra could be evaluated and sorted in a reasonable amount of time. However, there are 1,073,741,824 possible codes of memory length 14.
- Finding a single (2,1,14) code's distance spectrum is a complicated process that takes approximately 30 CPU seconds on the IBM RS6000 Model 580. At this rate, a search of every code would take roughly one millenia, not including the sort routine to find the best code.
- Thus, to make any search feasible, it is necessary to first pare down the number of codes that must be tested by using other techniques for detecting inferior codes.
- In addition, all catastrophic codes must be eliminated before attempting to find their distance spectrum. Catastrophic codes are codes in which a finite weight information sequence can generate an infinite weight codeword.
- This characteristic causes an infinite loop in the distance spectrum algorithm; if not eliminated these codes would make the search impossible.
- Unfortunately, an algorithm to recognize catastrophic codes is very complicated and time consuming because it involves factoring.

Methods of Reducing the Number of Codes

The number of codes can be reduced by making certain restrictions regarding the structure of the codes. These restrictions are based on known properties of convolutional codes and do not affect the search results in any way.

The two primary restrictions used were

1. both generators must start with a 1, and
2. one generator must end with a 1.

These restrictions reduce the number of codes by a factor of 8.

Second, an upper bound on the free distance can be utilized to eliminate codes that cannot achieve the known maximal free distance. This bound uses the row distance function, which is a decreasing function whose limit is the free distance.

For most codes, the row distance function converges quickly and is a very effective way of reducing the number of codes.

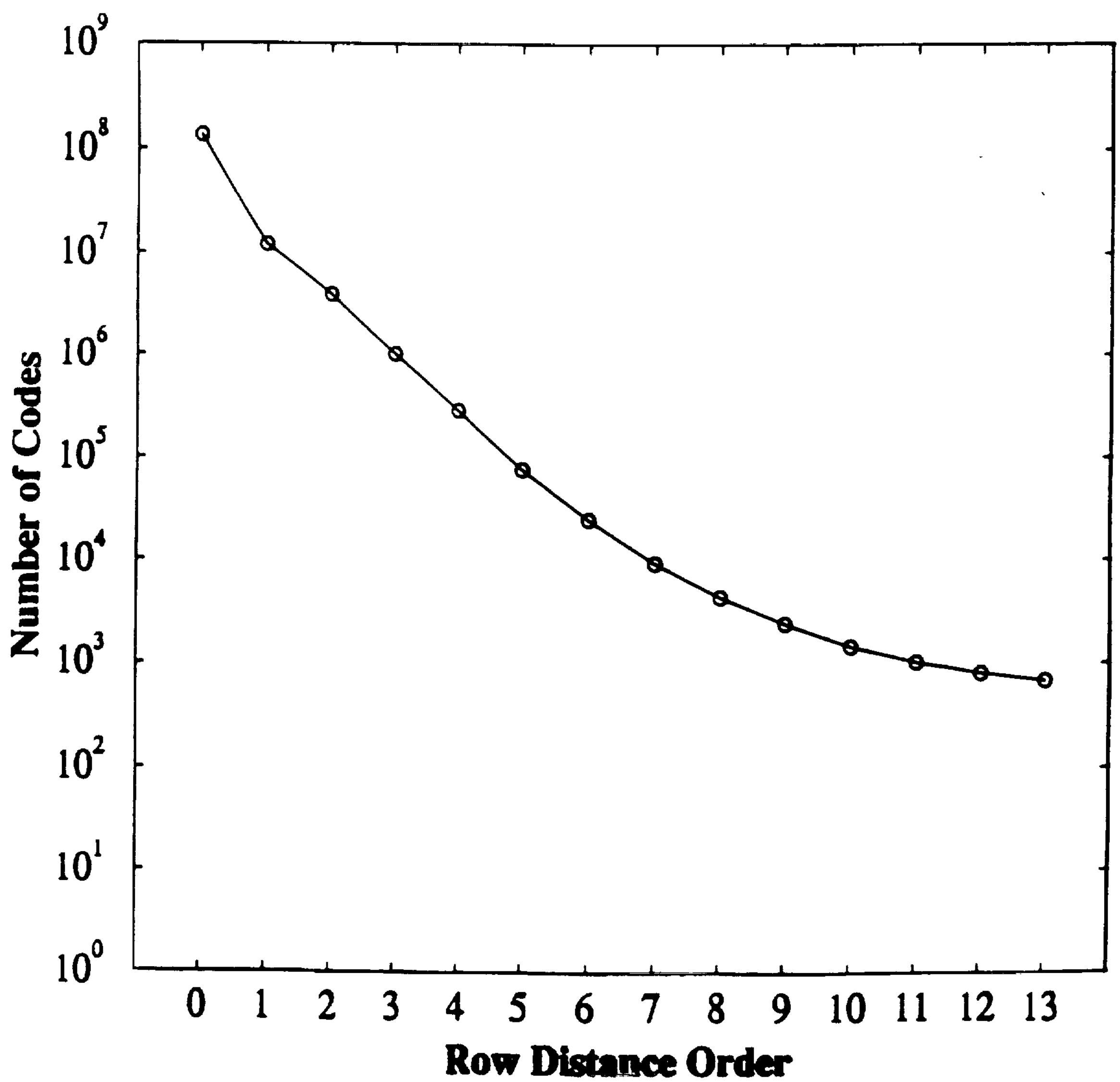
Third, codes whose generators are mirror images of each other can be eliminated, because they generate identical sets of codewords and thus identical distance spectrums.

Effectiveness of Schemes to Eliminate Codes

Initially, there are 1,073,741,824 possible codes and the search would have taken 1021 years.

After placing the two restrictions on the code generators, the number is reduced to 134,217,728. This search would have required 127 years.

The row distance evaluations, which require significant computational time, reduce the number of codes to a few hundred.



The Optimal Distance Search : FAST

With a reduced list of generators, evaluating the distance spectrum becomes feasible. This was done by implementing a version of the FAST algorithm (A Fast Algorithm for Searching a Tree) published by Cedervall and Johannesson.

Given a set of generators, the FAST algorithm builds and searches the code tree to determine the weight of all relevant code sequences. Using column distance function bounds to limit and speed the search, it ultimately returns the number of paths for the ten lowest weights.

Efficient programming and compiler optimization resulted in a CPU time of 30 seconds for the distance spectrum evaluation of one (2,1,14) code.

After using FAST to evaluate the candidate codes, the distance spectrum results must be sorted.

Search Results and Conclusions

- The (2,1,14) code with generators $(g1, g2)$ = (63057,44735) was found to be the optimal distance spectrum code.

- This code has only 26 weight 18 paths, as opposed to the previously known best (2,1,14) code which has 33 weight 18 paths. Thus, the new code is optimum for high signal-to-noise ratios.

- The new code is being simulated using computer models Notre Dame and a real decoder at the Jet Propulsion Laboratory in Pasadena, California, to determine if it is the best code for moderate SNR's.

- The techniques used in this code search are being refined and extended to find more complex codes for future NASA applications.

Optimal Rate 1/2, $K = 15$ ($m = 14$), Convolutional Codes

The rate 1/2, $K = 15$ (m=14) convolutional code found by Cedervall and Johanneson [1] is the optimum distance spectrum (ODS) code. The generators for this code are

$$\begin{aligned} g^{(1)} &= 63057 = 1 + D + D^4 + D^5 + D^9 + D^{11} + D^{12} + D^{13} + D^{14} \\ g^{(2)} &= 44735 = 1 + D^3 + D^6 + D^7 + D^8 + D^{10} + D^{11} + D^{12} + D^{14} \end{aligned}$$

and its distance spectrum is

d	18	19	20	21	22	23	24	25	26	27
N_d	26	0	165	0	845	0	4844	0	28513	0

The generators for the code in Lin and Costello [2] are

$$\begin{aligned} g^{(1)} &= 56721 = 1 + D^2 + D^3 + D^4 + D^6 + D^7 + D^8 + D^{10} + D^{14} \\ g^{(2)} &= 61713 = 1 + D + D^5 + D^6 + D^7 + D^8 + D^{11} + D^{13} + D^{14} \end{aligned}$$

and its distance spectrum is

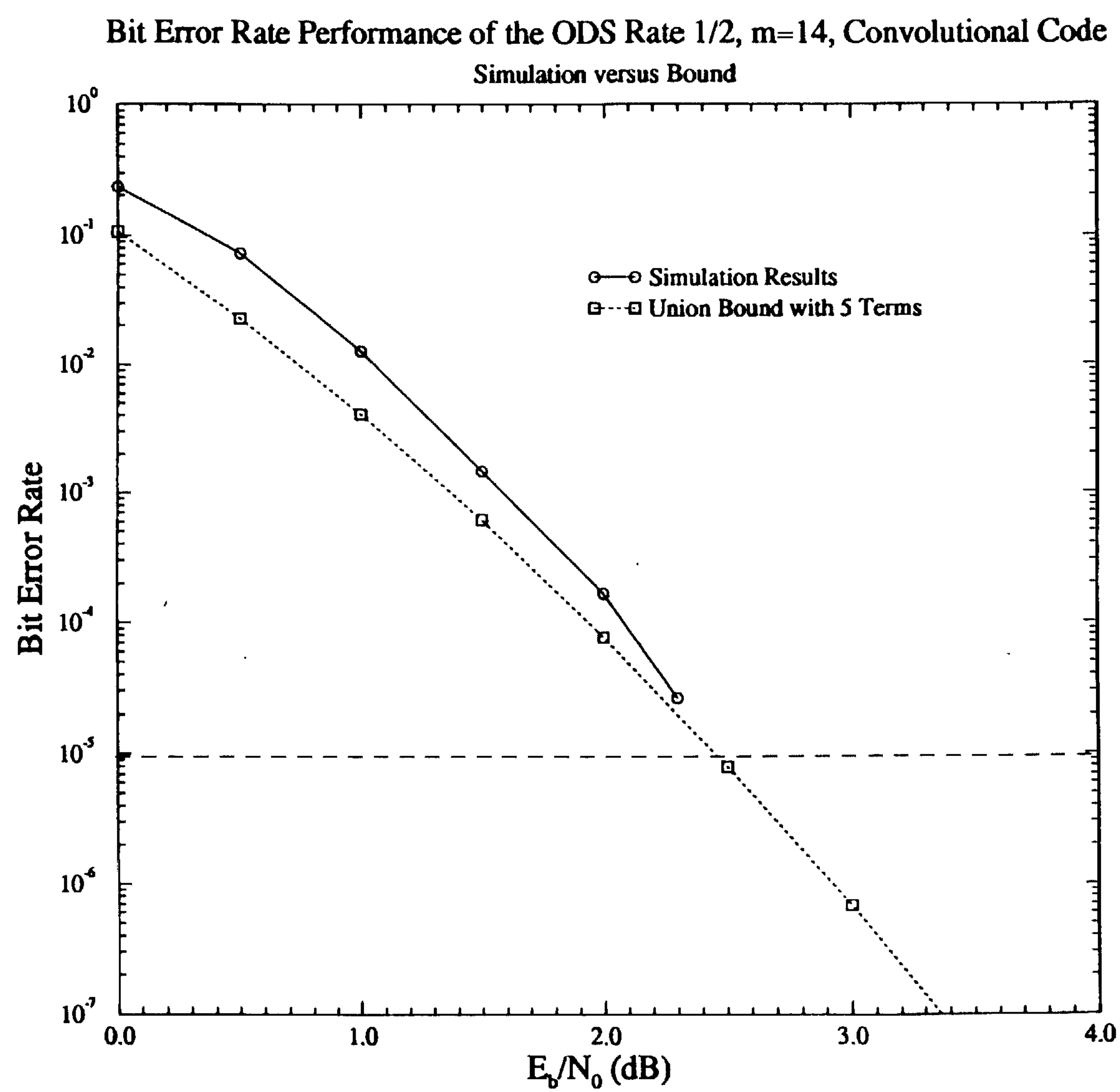
d	18	19	20	21	22	23	24	25	26	27
N_d	33	0	136	0	835	0	4787	0	27941	0

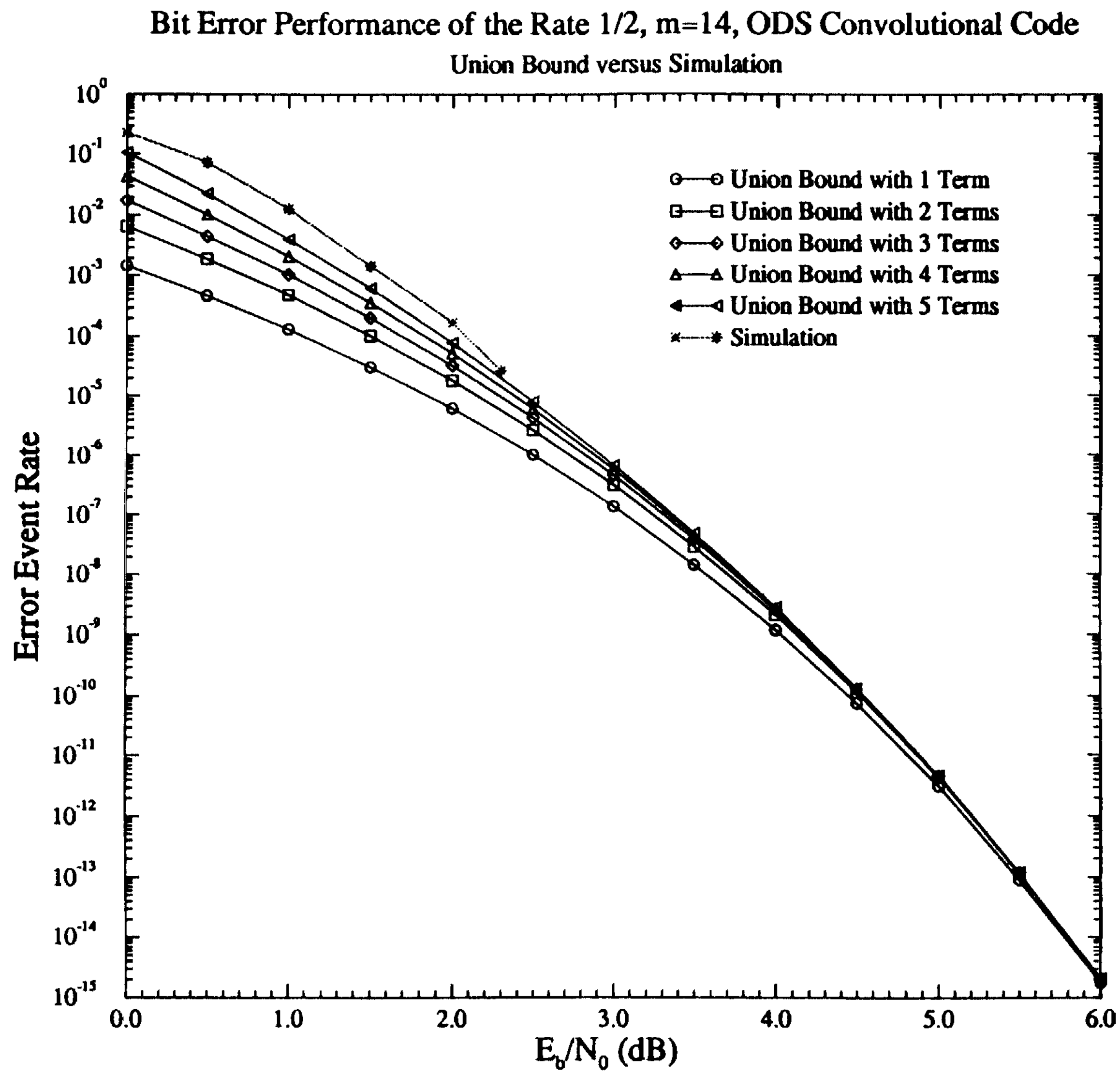
Both of these codes are invariant to 180^0 rotations of the QPSK signal set.

References

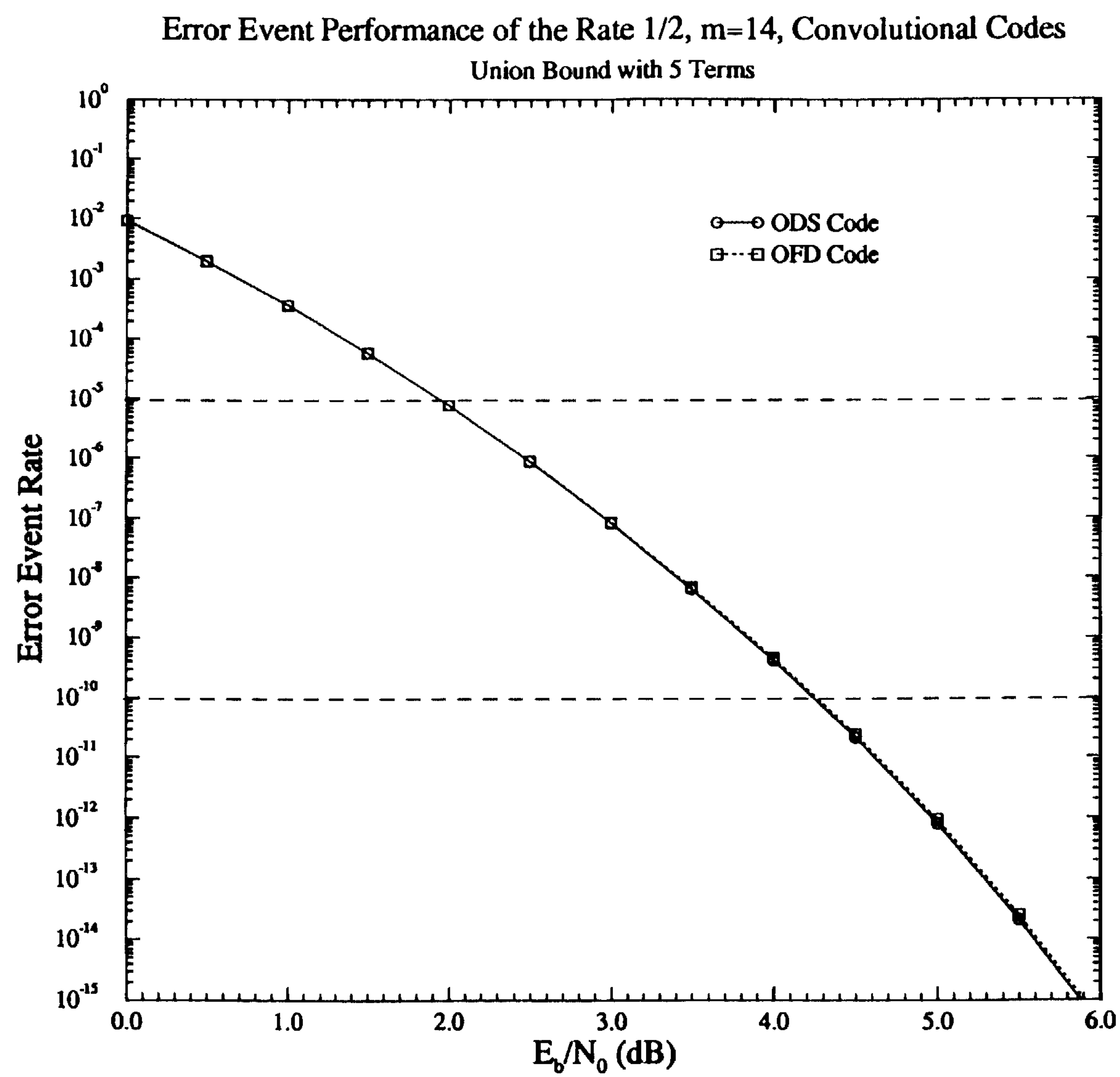
[1] M. Cedervall and R. Johanneson, "A Fast Algorithm for Computing Distance Spectrum of Convolutional Codes," *IEEE Trans. Inform. Theory*, **IT-35**, pp. 1146-1159, November 1989.

[2] S. Lin and D. J. Costello, Jr., *Error Control Coding: Fundamentals and Applications*, Prentice Hall, New Jersey, 1983.


Bit Error Rate Performance of the ODS Rate 1/2, m=14, Convolutional Code
Simulation versus Bound
Simulation Results
Union Bound with 5 Terms
Bit Error Rate
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
0.0
1.0
2.0
3.0
4.0
E_b/N_0 (dB)



Bit Error Performance of the Rate 1/2, m=14, ODS Convolutional Code
Union Bound versus Simulation
Union Bound with 1 Term
Union Bound with 2 Terms
Union Bound with 3 Terms
Union Bound with 4 Terms
Union Bound with 5 Terms
Simulation
Error Event Rate
E_b/N_0 (dB)
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
10^{-8}
10^{-9}
10^{-10}
10^{-11}
10^{-12}
10^{-13}
10^{-14}
10^{-15}
0.0
1.0
2.0
3.0
4.0
5.0
6.0


Error Event Performance of the Rate 1/2, m=14, Convolutional Codes
Union Bound with 5 Terms
ODS Code
OFD Code
Error Event Rate
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
10^{-8}
10^{-9}
10^{-10}
10^{-11}
10^{-12}
10^{-13}
10^{-14}
10^{-15}
0.0
1.0
2.0
3.0
4.0
5.0
6.0
E_b/N_0 (dB)