Determinate-State Convolutional Codes[1]

O. Collins and M. Hizlan
Johns Hopkins University
Baltimore, Maryland

A determinate-state convolutional code is formed from a conventional convolutional code by pruning away some of the possible state transitions in the decoding trellis. The type of staged power transfer used in determinate-state convolutional codes proves to be an extremely efficient way of enhancing the performance of a concatenated coding system. This article analyzes the decoder complexity and free distances of these new codes and provides extensive simulation results of their performance at the low signal-to-noise ratios where a real communications system would operate. The article concludes with concise, practical examples.

I. Introduction

In a determinate-state convolutional code, some of the possible branches of the trellis are pruned away, usually by employing an outer algebraic code, and the remaining paths in the convolutional code's trellis gain power from these "pinned" state transitions. The beauty of this technique is that it allows a concatenated coding system's performance to approach more closely the power of the concatenated code viewed as a single entity, while the decoding complexity remains comparable to the traditional sequential approach. This article will concentrate on convolutional rate $1/N$ inner codes, but trellis codes, block codes suitable for soft decoding, or more complex state machine-based codes can all use the same technique. The examples in this article, with one exception, will use algebraic outer codes for the conventional reason, i.e., the existence of decoding algorithms whose computational effort is only a polynomial in the error-correcting capability of the code. A short illustration in the next section defines the basic concept of determinate-state convolutional codes; the article then goes on to explore the decoding complexity of these new codes and their free distances. The article also presents design techniques and practical illustrations. Simulation is the only way to explore codes at the very low signal-to-noise ratios at which the inner codes in high- performance concatenated systems operate. Hence, extensive figures appear at the end of the article; these data can be used to design coding systems beyond what the article presents. The Appendix explains how the simulations were performed.

II. The Elements of Power Transfer

Figure 1 shows what happens to the trellis of a constraint length-3 convolutional code when one of the bits going into the encoder is known to be zero.

[1] Work supported in part by a contract between Johns Hopkins University and JPL through the TDA Office, and by a grant from the National Science Foundation.

This known bit could come from the successful decoding of an outer code or from something much simpler, such as a synchronization pattern. The example of an interleaved synchronization pattern provides a quick introduction to the idea of determinate-state convolutional codes and is an extreme case of a whole continuum of different degrees of certainty which the decoder may have about the information bits. As an interesting aside, most of the simpler varieties of such outside information, such as the different letter frequencies in ASCII encoded text or the statistics of pixel differences in a binary image, are easy to incorporate in the decoding procedure.

Suppose, for instance, that every eighth bit of data going in a (7,1/2) convolutional encoder is part of a synchronization word and that the code is operating at an E_b/N_0 of 1.2 dB. Figure A-1 shows that the bit error rate out of the decoder will be 2.7 percent. After having acquired synchronization, one can incorporate the knowledge of every eighth bit into the decoding procedure. Figure A-5 gives the bit error rates of the remaining bits when one out of every eight is known and shows that the error rate of the remaining 7/8 of the bits drops after incorporating this information to a little under 0.92 percent. However, the power used in sending these known bits must be accounted for. The proper adjustment to the bottom axis of Fig. A-5, to allow direct power comparison with Fig. A-1, is 10 log (7/8), which is 0.58 dB. Looking back to Fig. A-1 at the 1.8-dB point, there is an error rate of 0.89 percent. Thus, almost all of the power from the known bits has transferred to those that remain, i.e., $1.8 - 1.2 = 0.6$. The overall effect is almost the same as sending a signal back to the encoder, stopping the transmission from framing information.

The successful decoding of an outer code can provide the same certainty as knowledge of a synchronization pattern. If every eighth bit in the data stream going into the (7,1/2) code discussed above is a symbol in an outer binary code, and the parameters of this code are such that it always decodes with extremely high probability when the error rate is 2.7 percent, then a second soft decoding operation gives those bits which are not doubly protected just the same error rate they had when every eighth bit was part of a synchronization pattern. The information bits of the outer code have, in effect, been sent for free. A moderately complex code will require about twice the minimum possible redundancy of $H(0.027) = 0.18$ at the 2.7-percent error rate. This simple example incorporates the essence of the article.

It is the concept of power transfer that makes determinate-state convolutional codes work. The idea of reinitializing (essentially an extreme form of state pinning) a constraint length-7 decoder by using the bytes from an 8-bit interleaved Reed-Solomon (RS) code word has been tested before in [1–3] with varying degrees of success. Performance gains were either not any larger than other easier decoder improvements, such as increasing the truncation depth, or required many decoding trials of the Reed-Solomon code, with different positions erased. Adjusting the rates of the different code words in a block of interleaved Reed-Solomon codes will significantly improve the performance of a system using decoder reinitialization, but, as this article goes on to show, when the pinned symbol size is greater than the constraint length, power transfer is always inefficient. The article proceeds with an analysis of the properties of both determinate-state convolutional codes and their decoders.

III. Decoding Complexity and Implementation

Figure 2 demonstrates what happens as known bits percolate through an encoder shift register. The checkered circles represent zeros or ones that are known in advance and the box follows the shift register window of a constraint length-3 convolutional encoder. No additional argument is required to show that the total number of encoder states is cut in half for every known bit in the shift register.

The situation is, however, slightly more complex for the decoder since it performs one computational step for every possible state transition of the encoder rather than for every state, and thus the real measure of computational cost is the number of branches in the trellis. A return to the pruned trellis diagram in Fig. 1 confirms that the correct measure of complexity for a determinate decoding operation is just the average of $2^{K'}$, where K' is the effective constraint length of the pruned code, i.e., the number of unknown bits in the shift register. In fact, considering the trellis, it is possible to collapse the row of states immediately preceding a known bit so that if a known bit comes along once every $K+1$ bits, the peak decoder memory can be halved, and if a known bit comes along once in every K bits, the state diagram could look just like that of a $K-1$ code.

The decoder for a convolutional code in which some of the bits are known is straightforward to construct since it is identical to the decoder for a time-varying code of lower constraint length. This result derives from the capacity to represent the encoder for a determinate-state convolutional code as a conventional convolutional encoder

with time-varying generator polynomials whose constraint length is the effective constraint length of the determinate code, i.e., the number of unknown bits in the shift register. The output from each of the encoder's generator polynomials is either inverted or not depending on the known bits in the shift register. Figure 3 shows a rate $1/2, K = 12$ encoder with four known bits.

The two constants in Fig. 3 can be computed using Eq. (1)

$$(\vec{g}_i \cdot \vec{k}) \oplus r_i = r'_i \tag{1}$$

where $\vec{k}$ is the vector of information bits that are known to be one, $\vec{g}_i$ is the ith generator polynomial, and r_i is the bit which will be sent over the channel as the ith symbol. The clocking of the shift register is stopped as known bits enter.

This unusual view of the encoder can be used to construct an efficient decoder by using Eq. (1) to modify the signs of the received symbols, i.e., r_i is the bit that indicates whether the received symbol is a zero or a one and r'_i is the sign bit fed to the decoder. Since the two exclusive-or operations will cancel each other out, this preprocessing step will eliminate any requirement that determinate bits must be fed into the decoder itself. Only the need to recirculate the accumulated state metrics, i.e., to update them without exchanging them in order to handle the pauses in the Fig. 3 encoder, distinguishes the determinate decoder from a conventional Viterbi decoder of reduced K.

Equipping a conventional decoder to perform a maximum likelihood estimate when some bits are known is also straightforward. The only necessary additions are two wires that make it possible to force all the states to choose either the zero branch or the one branch, thus overriding the normal selection of the lower of the two incoming metrics. The DSN constraint length-15 decoder, described in [4], incorporates precisely this feature. The forcing lines also simplify the testing of the decoder since they allow error bursts to be inserted artificially. This decoder was demonstrated in early 1991.

IV. Determinate Code Properties

Another way of mechanizing the decoding process where subsets of the data stream are known is to give infinite weight to those branches that are not part of a possible path through the trellis. Since the shortest path between two points in a graph will remain the shortest when any line not on this path is lengthened,

$$S_i \mid \{S_\alpha, S_\beta, S_\gamma, \cdots\} = S_i \tag{2}$$

where S_i is the value of the ith symbol in the maximum likelihood estimate of the decoded data stream, and $S_i \mid \{S_\alpha, S_\beta, S_\gamma, \cdots\}$ is the estimate repeated with the knowledge that the bits in positions α, β, γ, etc. are correct. Thus, side information, e.g., known bits, is beneficial only if it requires a change in the estimate of the transmitted data stream. Similarly, if a sequence of k known symbols is sufficient to specify the encoder state, then

$$S_i \mid \{S_{i-k-1}, S_{i-k}, S_{i-k+1}, \cdots, S_{i-1}\} = S_i \mid \{Q\} \tag{3}$$

$$S_i \mid \{S_{i+1}, S_{i+2}, S_{i+3}, \cdots, S_{i+k}\} = S_i \mid \{Q'\} \tag{4}$$

where $\{Q\}$ is any subset of S_1 through S_{i-1} that includes S_{i-k-1} through S_{i-1}, and $\{Q'\}$ is any subset of S_j, where $j > i$, that includes S_{i+1} through S_{i+k}. Thus, if at any point in the decoding process enough side information is available to specify the encoder state, then the future decoding operations are no longer coupled to the past through the encoder memory. If the length of the sequence of known bits is shorter than that required to specify the encoder state fully, then the coupling will be decreased, but not eliminated, so re-decoding can relieve interleaving requirements. Equations (3) and (4) also show that there is intrinsic inefficiency in using 8-bit symbols with a $K = 7$ code, since pinning more than K bits in a row produces no additional power transfer.

The free distance of a determinate-state convolutional code can be obtained by making the same slight modifications to a decoder that are used to find the free distance of an ordinary convolutional code. While being fed all zero symbols, the decoder is forced out of the all-zero state into the state with all zeros and a single one. The total amount of distance that accumulates on the shortest path which brings the decoder back to the all-zero state is the free distance. Now, however, the decoder has to work with a pruned trellis, and so not all of the possible departure times from the all-zero state are equivalent; potentially, they may all have to be explored to find the free distance. When the information bit sequence that produces the minimum-weight burst is short, however, the problem of free-distance determination is trivial. If the run of unknown bits is longer than a sequence of information bits that produces a minimum-weight burst, the free distance of the root convolutional code cannot change as the

result of state pinning. In particular, if a single isolated one produces a minimum-weight burst, as it does for the NASA (7,1/2) code, then the free distance cannot increase no matter how many known bits there are. Even if a single bit remains unfixed, the minimum burst producing one will slide into that position.

V. First Example

The first example will replace a set of identical interleaved Reed-Solomon codes with a collection of different codes, which together have the same redundancy as the original set, but allow an overall gain of 0.5 dB over the old coding arrangement at its design error rate of 10^{-6} bit error rate (BER). The new system will actually have a lower error rate than 1 in a million. The inner code is the NASA (15,1/4) code used on the Galileo spacecraft. This is the (15,1/4) code whose performance is graphed and tabulated at the end of this article. The outer code used on the spacecraft to send compressed images is a (255,223) 8-bit Reed-Solomon code. The new outer codes will also be 8-bit Reed-Solomon codes, but with different amounts of redundancy. This example is chosen because an easily understood, very routine approach to code construction yields substantial improvements. For symbol errors within one Reed-Solomon code word to be almost independent of one another, they must be interleaved to a depth of 8, and to gain half a decibel, the redundancy will simply be shifted among the eight code words of an interleaving block, while keeping the average number of parity check symbols per code word fixed at 32. Since the redundancy of the outer code will remain constant, the energy gain will come from a lowering of the operating point on the inner code. The operating point of the inner code in the conventional system is 0.33 dB; in the proposed system, it is −0.2 dB.

In order to upper-bound the decoded bit error rate reliably, assume that if any Reed-Solomon code fails to decode, then all of its bits as well as the bits of any other undecoded Reed-Solomon code words in the frame experience a 50-percent error rate. Any concerns over frame-to-frame error propagation can be eliminated by inserting a sequence of 15 known bits after each frame; the frames are long enough so that the power penalty is negligible. Although this bound will be quite far above the actual error rate of the code, the difference measured in information bit energy will be small; i.e., little additional effort is required to lower the error rate of a coding system from one in a million to one in a billion. Figure A-10 shows how the 8-bit symbol error rate of the (15,1/4) code drops as additional side information becomes available. Table 1 shows the symbol error rates for each decoding operation and the contribution of the different Reed-Solomon code words to the average redundancy.

The decoder complexity of the new code design will be slightly less than three times the decoder complexity of the root constraint length-15 code; the last decoding step has complexity equal to that of a constraint length-7 code, and the second and third steps have complexity slightly below that of the first.

VI. Second Example

The second example, which has its source in the original Voyager communications system, addresses those situations where the outer code must be very simple, either because the encoder needs to be small, e.g., taking up only a small fraction of a chip, or because the short decoding delays characteristic of convolutional codes must be preserved; e.g., a compressed voice circuit operating at 2.5 Kbps should not have a coding delay of more than 250 bits. This example also shows how a communications system can be improved without changing the encoder. Voyager used a (7,1/2) convolutional code to send uncompressed images back to Earth at a bit error rate of 5×10^{-3}. Decoding errors, which very seldom affected more than two adjacent 8-bit pixels, were corrected either by low-pass filtering of the image or manually. A fraction of the data, however, came from other instruments that demanded no more than one error in a million bits, and these data were encoded first by a Golay code and then combined with the imaging data and sent to the convolutional encoder. For this example, one out of every eight bits will be covered with a Golay code, which is slightly larger than the fraction on Voyager. The bursts produced by a $K = 7$ code are short enough so that the bits of a Golay code word will experience independent errors when they are spaced 8 bits apart in the data stream.

A re-decoding operation using the information from a successful decoding of the (24,12) Golay code will cut the error rate of the (7,1/2) convolutional code by a factor of 3. The probability of five or more errors in a Golay code word is so small that incorrect decodings have a negligible effect on the error rate of the bits following a second decoding. The errors produced by an incorrect Golay decoding can be upper-bounded by assuming that an error burst covers all those bits intermixed with the Golay code word, plus the bits for several constraint lengths on either side. The convolutional code's constraint length would have to be increased to 9 in order to achieve the same improvement. If a similar interleaved Golay code was used on the (15,1/4)

convolutional code, the error rate would drop by a factor of 5. The second decoding operation increases the decoding complexity by a little more than a quarter since one determinate bit is always in the shift register, and two are in the shift register 7/8 of the time. Increasing the convolutional code's constraint length in order to achieve similar gains would raise the decoding complexity by a factor of 32.

In this example, the energy per information bit measured at an error rate of 10^{-3} does not decrease; instead, the second decoding step allows the error rate of 1/8 of the bits to be reduced to 10^{-6} without any increase in E_b/N_0 and with only an extremely small increase in encoder complexity. Variations on this scheme might be very useful for certain types of data compression systems that transmit both the parameters of a predictor and the differences between the predictor and the source; these differences can tolerate a much larger error rate.

VII. Third Example

The third example will give a constraint length-11 convolutional code the same performance as the constraint length-15 code put aboard Galileo. The outer code will remain an 8-bit Reed-Solomon code, and the average number of redundant symbols per 255 symbol code word will stay fixed at 32. For a constraint length-11 code, an interleaving depth of four is sufficient to achieve symbol independence, and the re-decoding operation will take place after one code word of an interleaving block has decoded and every fourth 8-bit symbol is thus known. Only a single determinate decoding operation is necessary to increase the performance of a constraint length-11 code to a little better than that of the constraint length-15 code.

The operating point of the $K = 11$ code is 0.3 dB, giving an 8-bit symbol error rate of 4.92 percent, and 68 parity check symbols will bring the probability of Reed-Solomon decoder failure below 7×10^{-8}. After the pinned decoding operation, the average error rate of the remaining symbols will drop to 1.25 percent. Twenty redundant symbols on each of the three remaining Reed-Solomon code words will reduce the bit error rate to 10^{-6}. Of course, if the first Reed-Solomon code fails to decode, then a determinate decoding operation will not be possible. The three remaining, much less powerful, code words will be left to cope with a symbol error rate, which will usually be worse than the average symbol error rate before a determinate decoding operation. Thus, when the first code fails, there is very little point in trying to decode any of the other three, and the decoder will just pass the bits from the convolutional encoder out without further processing. Simulation results show that the average bit error rate over a failed block is 3.2 percent. The effect of the primary Reed-Solomon code failure on the error rate of the bits in the other three code words is thus an increase of less than 0.22 percent. Reference [5] shows that Reed-Solomon decoder errors are so infrequent they can be ignored. The superior error rate of the bits in the primary code word will bring the aggregate system error performance below 1 in a million.

The average number of redundant symbols per code word is thus $1/4 \times 68 + 3/4 \times 20 = 32$, the same as the NASA standard outer code. The operating point of the inner convolutional code on Galileo is 0.33 dB, so the state pinned system is not only 1/8 as complex as the conventional, but performs slightly better. A second soft decoding operation could even improve the performance of the constraint length-11 code well beyond that of the Galileo code.

VIII. Summary

This article has examined what happens to a convolutional code when some of the state transitions are predetermined, usually by the successful decoding of an outer code. This type of technique has to justify itself by being more efficient than other methods, such as increasing the convolutional code's constraint length or passing erasure information to the outer code. The examples were therefore constructed to demonstrate improved performance lucidly, without the ambiguities that varying the overall code rate would introduce. The examples were not constructed to present an optimal system. Such a coding system would require codes between BCH and RS codes, i.e., with a symbol size of 4 or 5 bits and 100 or more symbols in a code word. The hybridized Reed-Solomon codes described in [4] can fulfill these requirements.

The results in this article show that in a concatenated coding system, a determinate decoding operation is a more efficient way to expend computational resources than an increase of the constraint length by 1. A second and third determinate decoding operation can often be justified. Other types of coupling between the inner and outer codes do not approach the gains possible with determinate decoding. Even perfect erasure declaration, which would cut the redundancy in the outer code by half, would not achieve the performance gains demonstrated.

Determinate decoding is an especially useful technique where the encoder must be extraordinarily simple or where all of the information bits do not require equal error pro-

tection. Many types of compressors produce such outputs. The bits coming out of a determinate decoding operation may have a low enough error rate so that no outer code is needed to cover them. Furthermore, techniques presented in this article offer an especially efficient way of using surplus speed in a Viterbi decoder or of trading decoding speed for performance; e.g., the same machine might be used for deep-space probes which demand the highest possible coding gains and for near-Earth satellites which demand high data rates.

Table 1. Symbol error rates

Symbols known	Error rate, percent	R/S redundancy				Average
None	10.4	104	x	1/8	=	13
Every eighth	6.25	68	x	1/8	=	8.5
Every fourth	1.13	26	x	1/4	=	6.5
Every second	0.106	8	x	1/2	=	4
Total						32

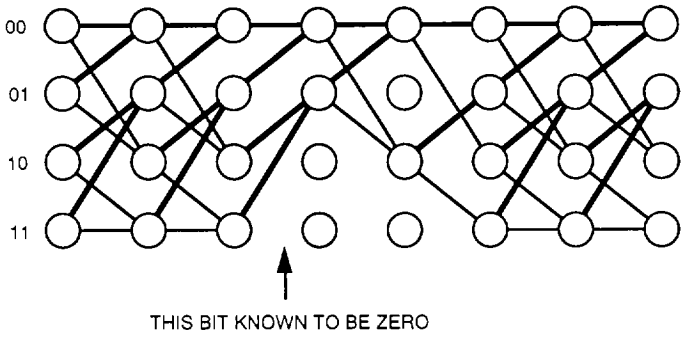

Fig. 1. Constraint length-3 convolutional code trellis.

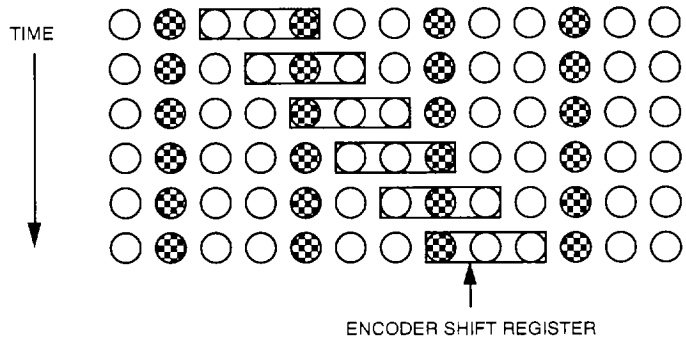

Fig. 2. Encoder shift register.

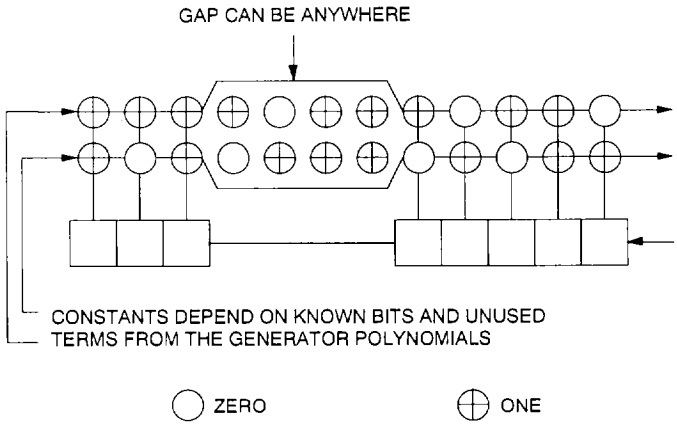

Fig. 3. A rate 1/2, K = 12 encoder with four known bits.

Appendix

The simulations in this article were performed by encoding sequences of pseudo-randomly generated information bits, adding white Gaussian noise for a specified signal-to-noise ratio, quantizing the resulting noisy symbols into 8-bit symbols, and decoding the result. The use of pseudo-random information bits rather than all zeros or all ones eliminated the possibility that a programming error would produce a higher than normal performance.

Random number generators for both the information bits and the noise were adapted from *Numerical Recipes in C* [6]. The pseudo-random bit generator employs the algorithm irbit2 [6] with a primitive polynomial modulo 2 of order 32 with nonzero coefficients (32, 7, 5, 3, 2, 1, 0). Hence, the resulting sequence of pseudo-random bits does not repeat for more than 4,000,000,000 bits. The white Gaussian noise generator uses gasdev [6] with the routine ran1 [6]. Again, the resulting sequence has a virtually infinite period for simulation purposes.

The noisy encoded symbols are quantized before decoding into 8-bit symbols based on the desired signal-to-noise ratio, so that the expected quantizer saturation rate is 0.1 percent. The decoder employs the Viterbi algorithm modified to do trace-backs rather than entire path updates as in [4]. This saves a tremendous amount of computer time and makes simulation of codes with constraint lengths as high as 15 feasible on available workstations. Decoding is done on blocks of 170 bits after a 170-bit trace back through the trellis. For a detailed description of the algorithm, as well as its hardware implementation, see [4]. The decoding algorithm used is a direct translation of the one given there. Free distances and related computation can be performed by employing this decoding algorithm in conjunction with a forced starting state. Such free distances will then always include the effects of state pinning.

Each code has been simulated for a set of signal-to-noise ratios likely to produce the most relevant range of error rates. The number of bits decoded for each signal-to-noise ratio run is 2,000,000 for constraint length-15 codes, 3,000,000 for constraint length-13 codes, and 4,000,000 for all other codes. In each case, a signal-to-noise ratio run consists of two runs with half the total number of bits and different seeds for random number generators. The 95-percent confidence interval for the simulation results, based on applying counting statistics to the error bursts, is not larger than 0.11 dB in the worst case, and is typically much smaller.

A subset of the simulation runs have been presented as figures. Figures A-1 to A-4, showing symbol error rate, are used to design coding systems with Reed-Solomon or other similar outer codes. Figures A-5 to A-7 showing bit error rates, are useful for comparing the effectiveness of power transfer for different patterns of forced bits, i.e., for exploring the differences between having the determinate bits spread out and having them occur in bunches. Figures A-8 to A-10 show how the effective signal-to-noise ratio of the inner code changes as different numbers of symbols are forced; thus, they allow an easy evaluation of the benefits of multiple re-decoding operations. The data from all of the simulation runs appear in Tables A-1 through A-12.

The (7,1/2) code was used on Voyager, and the (15,1/4) is now flying on Galileo. The (11,1/4) code was made up at the computer keyboard by one of the authors. Tables are available from the authors containing a title bar with the code name, free distance of the code, the generating polynomial of the code, and the known symbol frequency (with the resulting power added in decibels). Bit error rates (BER) and symbol error rates (SER) are presented for 1-bit, 4-bit and 8-bit symbols as a function of signal-to-noise ratio (SNR) in each table. SER @ n stands for symbol error rate for the nth symbol after the known symbol.

References

[1] L. N. Lee, "Concatenated Coding Systems Employing a Unit Memory Convolutional Code and a Byte Oriented Decoding Algorithm," *IEEE Trans. Communications*, vol. COM-25, pp. 1064–1074, October 1977.

[2] G. W. Zeoli, "Coupled Decoding of Block-Convolutional Concatenated Codes," *IEEE Trans. Communications*, vol. COM-21, pp. 219–226, March 1973.

[3] E. Paaske, "Improved Decoding for a Concatenated Coding System Recommended by CCSDS," *IEEE Trans. Communications*, vol. COM-38, pp. 1138–1144, August 1990.

[4] O. Collins, "Coding Beyond the Computational Cutoff Rate," Ph.D. thesis, California Institute of Technology, Pasadena, California, 1989.

[5] R. J. McEliece and L. Swanson, "On the Decoder Error Probability for Reed-Solomon Codes," *IEEE Transactions on Information Theory*, vol. IT-32, pp. 701–703, September 1986.

[6] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Wettering, *Numerical Recipes in C—The Art of Scientific Computing*," Cambridge: Cambridge University Press, 1988.

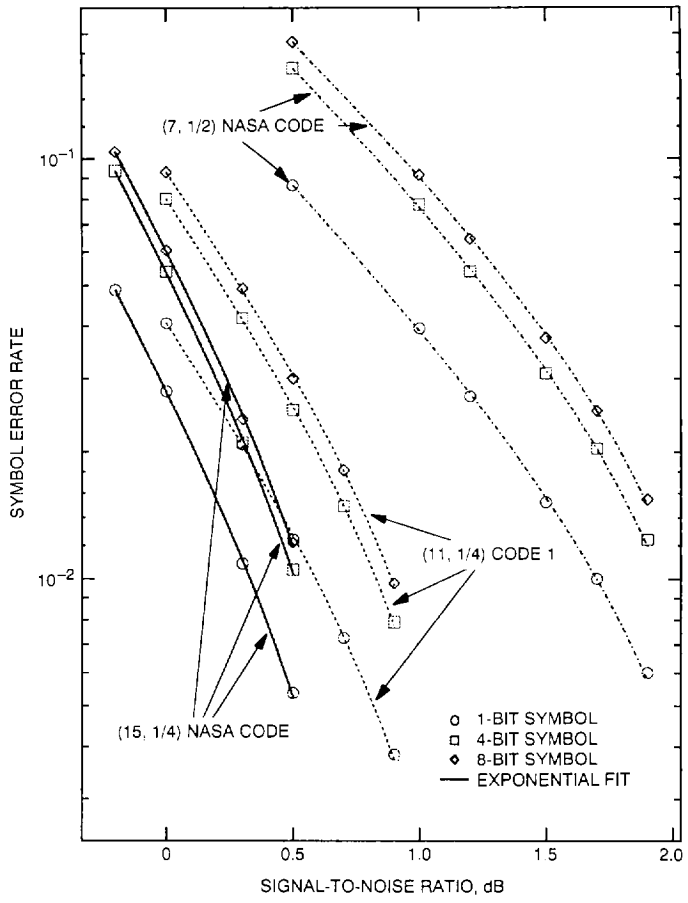

Fig. A-1. Symbol (and bit) error rate with no symbol known.

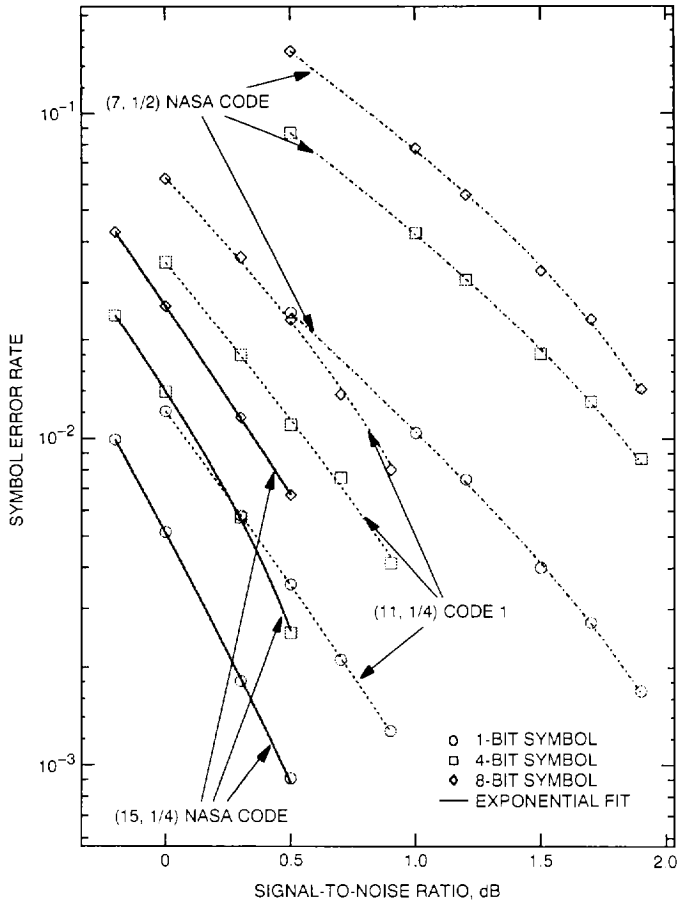

Fig. A-2. Symbol error rate with every eighth symbol known.

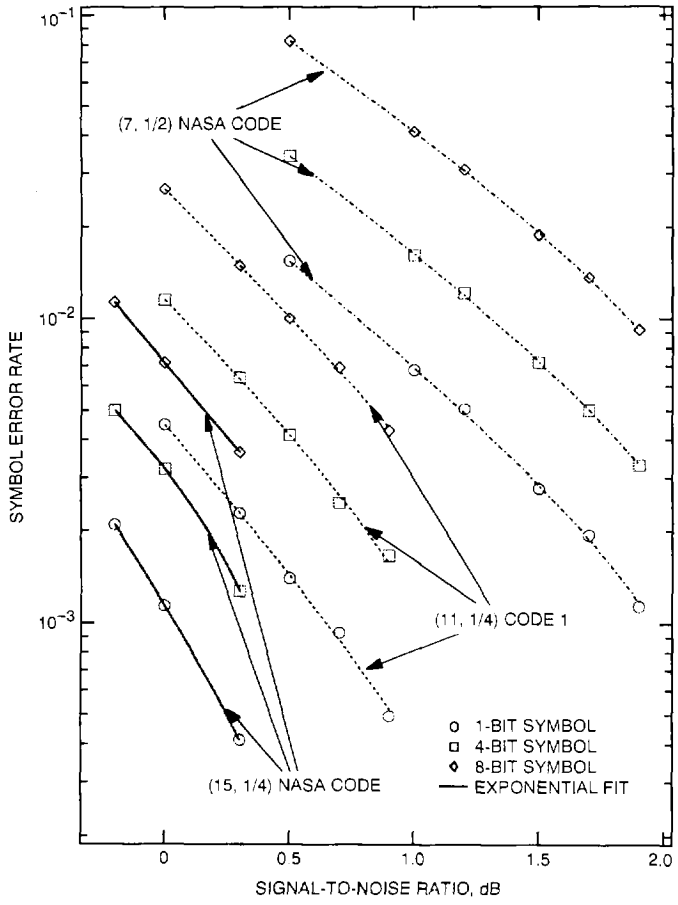

Fig. A-3. Symbol error rate with every fourth symbol known.

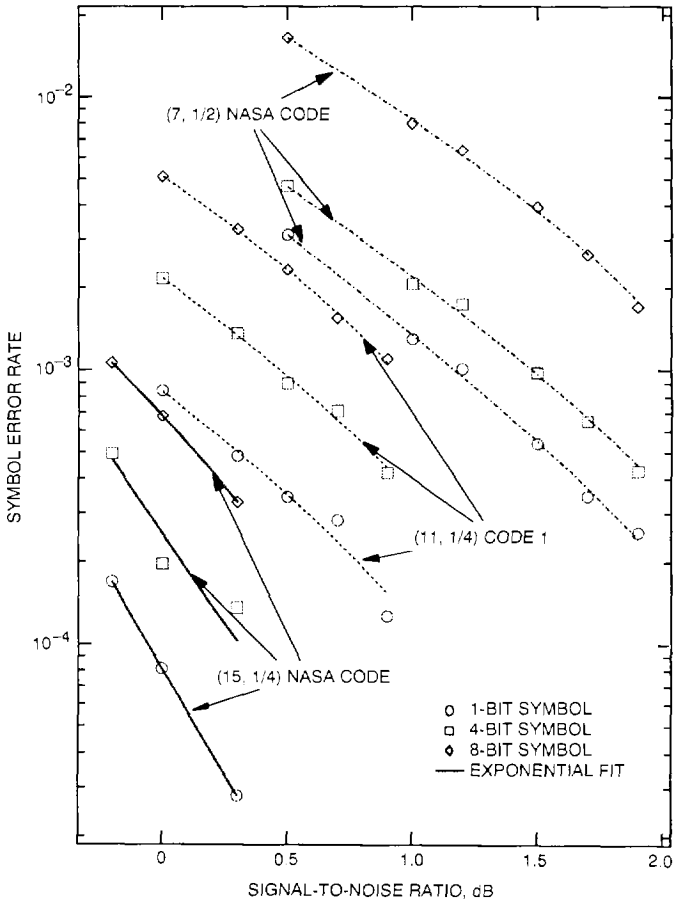

Fig. A-4. Symbol error rate with every other symbol known.

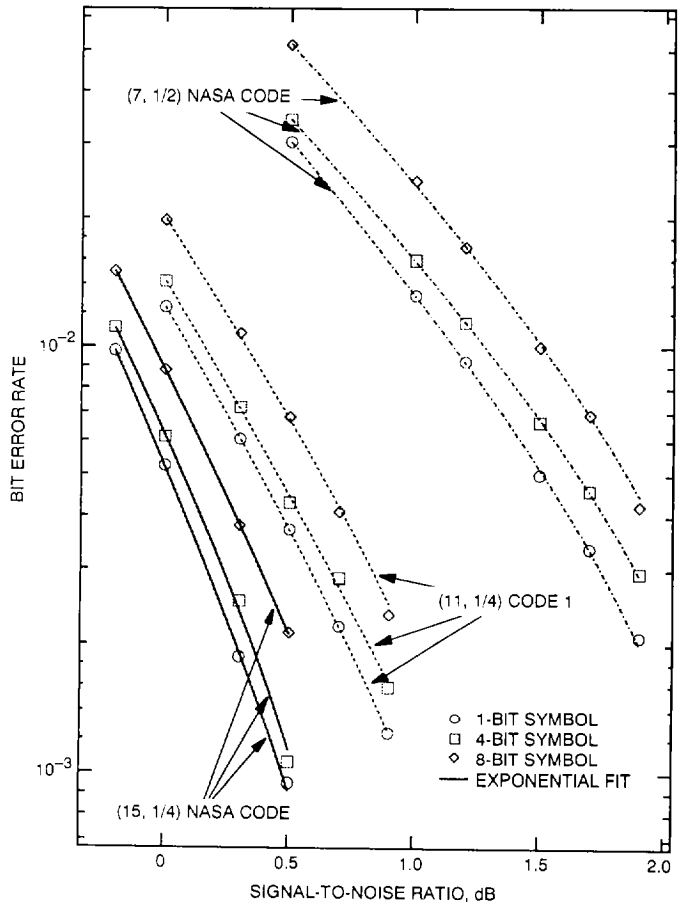

Fig. A-5. Bit error rate with every eighth symbol known.

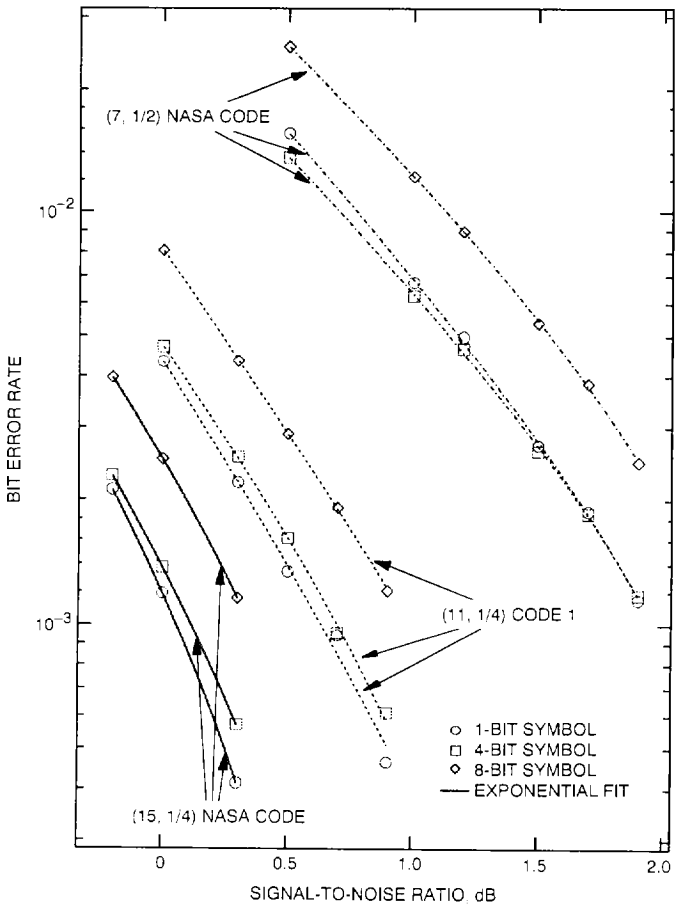

Fig. A-6. Bit error rate with every fourth symbol known.

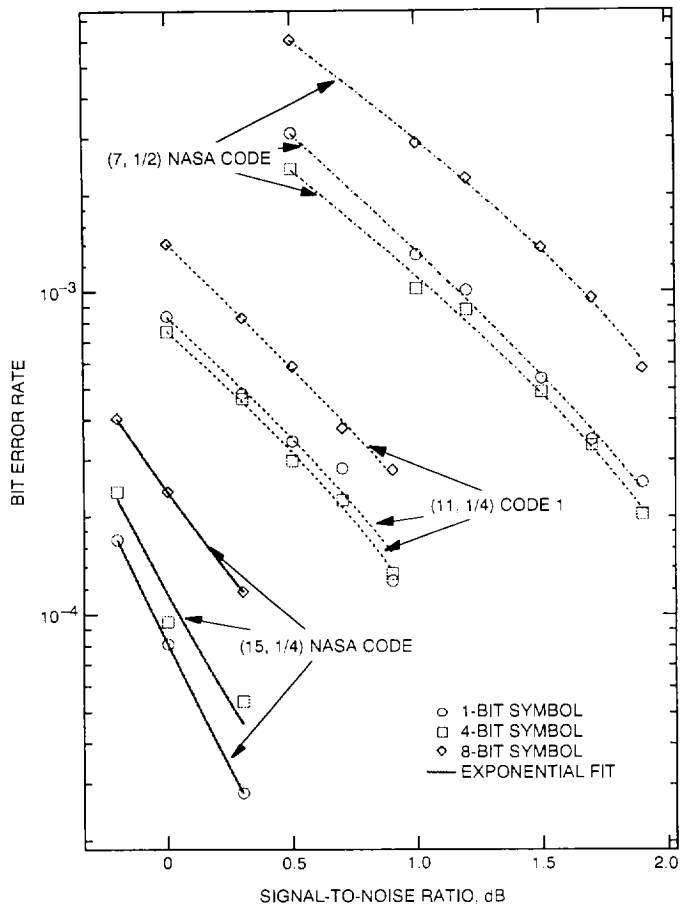

Fig. A-7. Bit error rate with every other symbol known.

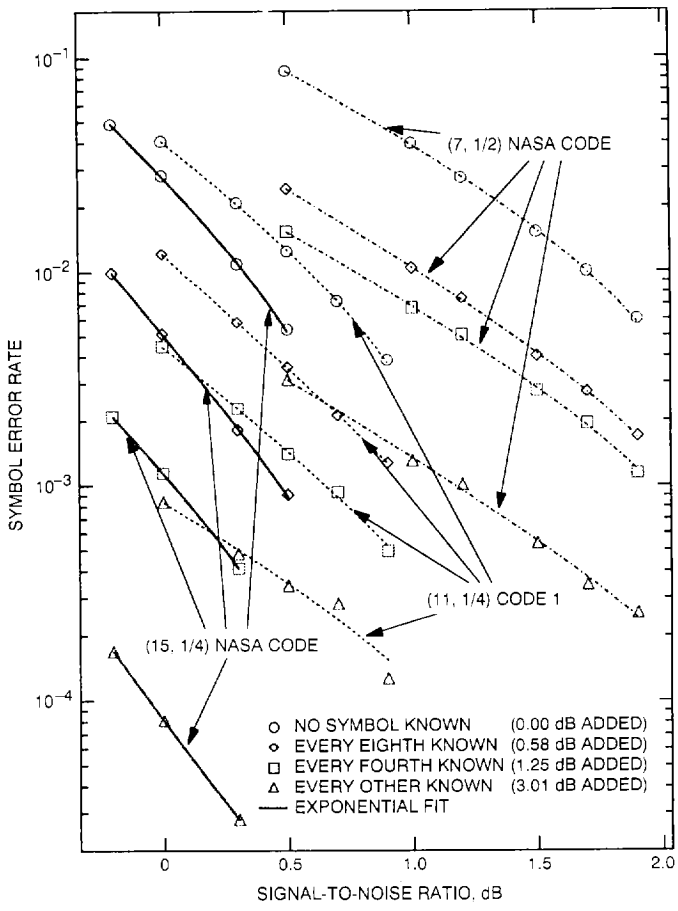

Fig. A-8. One-bit symbol error rates with different numbers of known bits.

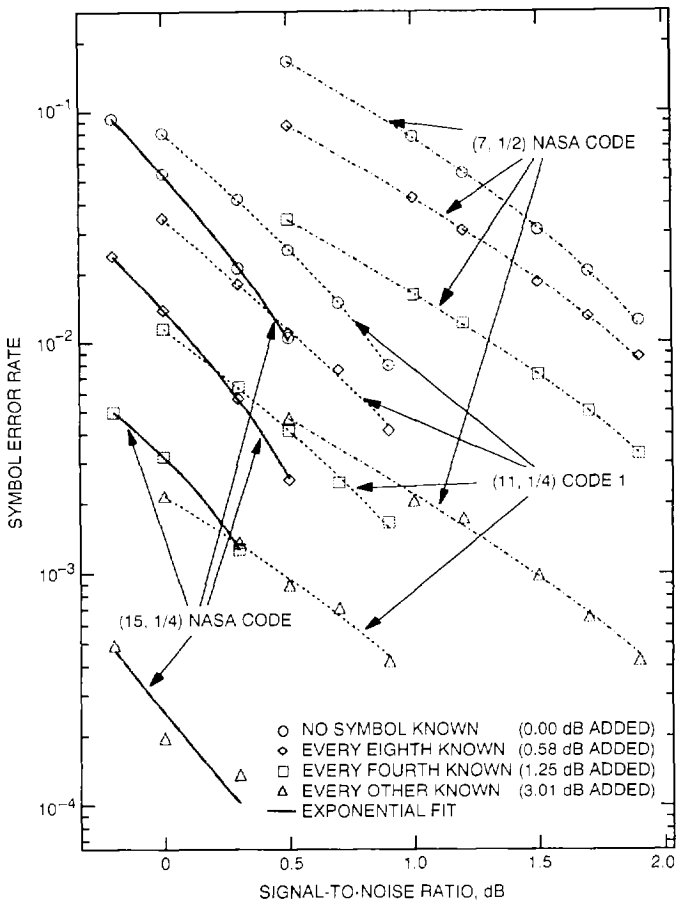

Fig. A-9. Four-bit symbol error rates with different numbers of known bits.

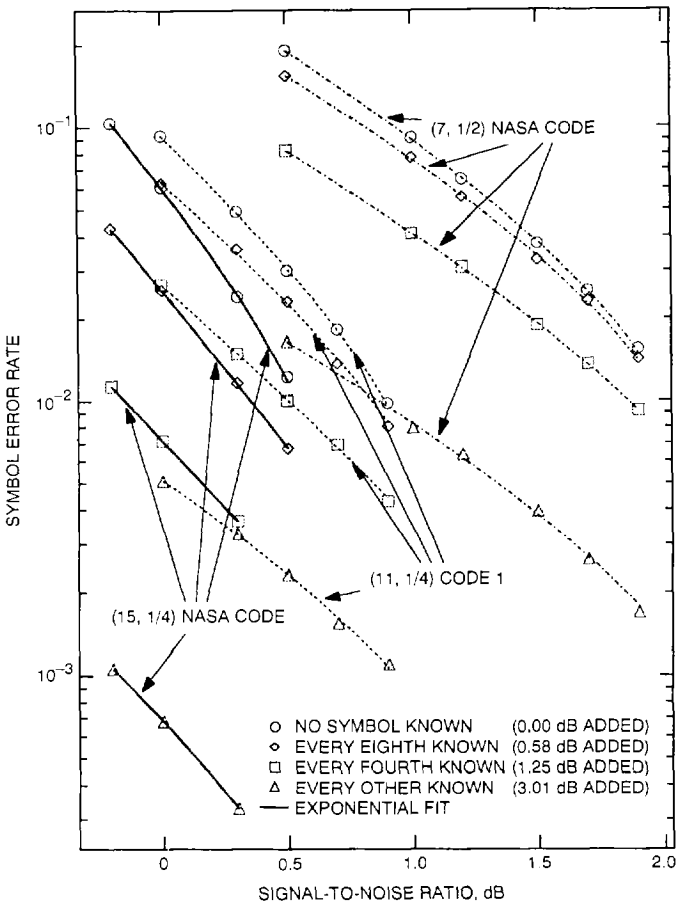

Fig. A-10. Eight-bit symbol error rates with different numbers of known bits.

Table A-1. No symbol known to the decoder, (7, 1/2) NASA code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		No symbol known to the decoder. (dB added = 0.00)			
(7, 1/2) NASA Code	10	1111001 1011011					
SYMBOL SIZE	SNR (dB)	0.5	1.0	1.2	1.5	1.7	1.9
1-Bit	SER	8.65e-02	3.95e-02	2.72e-02	1.53e-02	1.00e-02	6.00e-03
4-Bit	SER	1.66e-01	7.77e-02	5.41e-02	3.08e-02	2.04e-02	1.24e-02
8-Bit	SER	1.91e-01	9.15e-02	6.46e-02	3.75e-02	2.51e-02	1.55e-02

Table A-2. Every eighth symbol known to the decoder, (7, 1/2) NASA code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		Every eighth symbol known to the decoder. (dB added = 0.58)			
(7, 1/2) NASA Code	10	1111001 1011011					
SYMBOL SIZE	SNR (dB)	0.5	1.0	1.2	1.5	1.7	1.9
1-Bit	BER	3.03e-02	1.32e-02	9.26e-03	5.01e-03	3.36e-03	2.09e-03
	SER @ 1	3.00e-02	1.31e-02	9.30e-03	4.90e-03	3.33e-03	2.15e-03
	SER @ 2	3.20e-02	1.41e-02	9.75e-03	5.35e-03	3.59e-03	2.22e-03
	SER @ 3	3.17e-02	1.39e-02	9.72e-03	5.29e-03	3.57e-03	2.18e-03
	SER @ 4	2.43e-02	1.04e-02	7.49e-03	4.02e-03	2.73e-03	1.69e-03
	SER @ 5	3.01e-02	1.31e-02	9.16e-03	4.99e-03	3.27e-03	2.10e-03
	SER @ 6	3.23e-02	1.42e-02	9.84e-03	5.38e-03	3.62e-03	2.18e-03
	SER @ 7	3.15e-02	1.36e-02	9.56e-03	5.12e-03	3.42e-03	2.08e-03
4-Bit	BER	3.42e-02	1.59e-02	1.14e-02	6.66e-03	4.59e-03	2.95e-03
	SER @ 1	4.68e-02	2.15e-02	1.55e-02	8.64e-03	5.62e-03	3.83e-03
	SER @ 2	7.00e-02	3.35e-02	2.44e-02	1.43e-02	9.98e-03	6.42e-03
	SER @ 3	8.31e-02	4.03e-02	2.94e-02	1.74e-02	1.22e-02	7.86e-03
	SER @ 4	8.66e-02	4.26e-02	3.06e-02	1.82e-02	1.30e-02	8.66e-03
	SER @ 5	8.35e-02	4.03e-02	2.86e-02	1.74e-02	1.20e-02	7.95e-03
	SER @ 6	7.11e-02	3.38e-02	2.38e-02	1.47e-02	9.70e-03	6.35e-03
	SER @ 7	4.79e-02	2.13e-02	1.53e-02	8.94e-03	5.76e-03	3.52e-03
8-Bit	BER	5.14e-02	2.45e-02	1.72e-02	1.00e-02	6.93e-03	4.22e-03
	SER @ 1	7.91e-02	3.62e-02	2.75e-02	1.64e-02	1.12e-02	6.78e-03
	SER @ 2	1.27e-01	6.10e-02	4.52e-02	2.76e-02	1.87e-02	1.14e-02
	SER @ 3	1.48e-01	7.36e-02	5.24e-02	3.27e-02	2.18e-02	1.32e-02
	SER @ 4	1.55e-01	7.76e-02	5.57e-02	3.27e-02	2.32e-02	1.42e-02
	SER @ 5	1.46e-01	7.43e-02	5.26e-02	3.09e-02	2.28e-02	1.43e-02
	SER @ 6	1.25e-01	6.31e-02	4.41e-02	2.62e-02	1.91e-02	1.28e-02
	SER @ 7	7.91e-02	3.84e-02	2.61e-02	1.58e-02	1.13e-02	6.91e-03

Table A-3. Every fourth symbol known to the decoder, (7, 1/2) NASA code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		Every fourth symbol known to the decoder. (dB added = 1.25)			
(7, 1/2) NASA Code	10	1111001 1011011					
SYMBOL SIZE	SNR (dB)	0.5	1.0	1.2	1.5	1.7	1.9
1-Bit	BER	1.56e-02	6.74e-03	4.99e-03	2.71e-03	1.88e-03	1.15e-03
	SER @ 1	1.48e-02	6.31e-03	4.68e-03	2.54e-03	1.70e-03	1.11e-03
	SER @ 2	1.54e-02	6.77e-03	5.05e-03	2.76e-03	1.93e-03	1.14e-03
	SER @ 3	1.66e-02	7.14e-03	5.23e-03	2.83e-03	2.01e-03	1.20e-03
4-Bit	BER	1.36e-02	6.25e-03	4.67e-03	2.62e-03	1.86e-03	1.18e-03
	SER @ 1	2.60e-02	1.20e-02	8.93e-03	5.09e-03	3.60e-03	2.40e-03
	SER @ 2	3.42e-02	1.61e-02	1.21e-02	7.18e-03	5.00e-03	3.29e-03
	SER @ 3	2.62e-02	1.22e-02	9.22e-03	5.30e-03	3.53e-03	2.26e-03
8-Bit	BER	2.53e-02	1.22e-02	8.96e-03	5.37e-03	3.85e-03	2.47e-03
	SER @ 1	5.94e-02	2.94e-02	2.18e-02	1.32e-02	9.74e-03	6.25e-03
	SER @ 2	8.23e-02	4.10e-02	3.08e-02	1.88e-02	1.36e-02	9.18e-03
	SER @ 3	6.02e-02	2.92e-02	2.15e-02	1.36e-02	9.11e-03	6.13e-03

Table A-4. Every other symbol known to the decoder, (7, 1/2) NASA code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		Every other symbol known to the decoder. (dB added = 3.01)			
(7, 1/2) NASA Code	10	1111001 1011011					
SYMBOL SIZE	SNR (dB)	0.5	1.0	1.2	1.5	1.7	1.9
1-Bit	BER	3.11e-03	1.30e-03	1.01e-03	5.37e-04	3.46e-04	2.55e-04
	SER @ 1	3.11e-03	1.30e-03	1.01e-03	5.37e-04	3.45e-04	2.55e-04
4-Bit	BER	2.41e-03	1.02e-03	8.78e-04	4.87e-04	3.32e-04	2.04e-04
	SER @ 1	4.68e-03	2.06e-03	1.74e-03	9.78e-04	6.52e-04	4.26e-04
8-Bit	BER	6.05e-03	2.90e-03	2.25e-03	1.37e-03	9.52e-04	5.77e-04
	SER @ 1	1.65e-02	7.95e-03	6.36e-03	3.94e-03	2.64e-03	1.70e-03

Table A-5. No symbol known to the decoder, (11, 1/4) first code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		No symbol known to the decoder. (dB added = 0.00)		
(11, 1/4) First Code	23	10001011011 11101010001 10110101001 11000100101				
SYMBOL SIZE	SNR (dB)	0.0	0.3	0.5	0.7	0.9
1-Bit	SER	4.07e-02	2.09e-02	1.24e-02	7.25e-03	3.84e-03
4-Bit	SER	8.03e-02	4.17e-02	2.52e-02	1.49e-02	7.92e-03
8-Bit	SER	9.30e-02	4.92e-02	3.00e-02	1.82e-02	9.77e-03

Table A-6. Every eighth symbol known to the decoder, (11, 1/4) first code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		Every eighth symbol known to the decoder. (dB added = 0.58)		
(11, 1/4) First Code	23	10001011011 11101010001 10110101001 11000100101				
SYMBOL SIZE	SNR (dB)	0.0	0.3	0.5	0.7	0.9
1-Bit	BER	1.24e-02	6.05e-03	3.72e-03	2.21e-03	1.24e-03
	SER @ 1	1.18e-02	5.89e-03	3.68e-03	2.08e-03	1.20e-03
	SER @ 2	1.26e-02	6.08e-03	3.77e-03	2.21e-03	1.30e-03
	SER @ 3	1.27e-02	6.11e-03	3.85e-03	2.26e-03	1.25e-03
	SER @ 4	1.21e-02	5.80e-03	3.57e-03	2.11e-03	1.27e-03
	SER @ 5	1.24e-02	6.24e-03	3.69e-03	2.33e-03	1.24e-03
	SER @ 6	1.27e-02	6.21e-03	3.75e-03	2.27e-03	1.23e-03
	SER @ 7	1.23e-02	6.05e-03	3.69e-03	2.22e-03	1.20e-03
4-Bit	BER	1.42e-02	7.18e-03	4.29e-03	2.87e-03	1.59e-03
	SER @ 1	2.29e-02	1.15e-02	6.42e-03	4.47e-03	2.26e-03
	SER @ 2	2.93e-02	1.54e-02	9.21e-03	6.26e-03	3.38e-03
	SER @ 3	3.30e-02	1.72e-02	1.08e-02	7.38e-03	3.98e-03
	SER @ 4	3.46e-02	1.80e-02	1.10e-02	7.58e-03	4.14e-03
	SER @ 5	3.34e-02	1.71e-02	1.05e-02	7.50e-03	4.10e-03
	SER @ 6	2.93e-02	1.50e-02	9.04e-03	6.20e-03	3.61e-03
	SER @ 7	2.30e-02	1.14e-02	6.78e-03	4.37e-03	2.51e-03
8-Bit	BER	1.98e-02	1.07e-02	6.82e-03	4.10e-03	2.36e-03
	SER @ 1	3.04e-02	1.68e-02	1.10e-02	7.04e-03	4.19e-03
	SER @ 2	4.95e-02	2.78e-02	1.72e-02	1.10e-02	6.94e-03
	SER @ 3	6.00e-02	3.38e-02	2.16e-02	1.32e-02	8.13e-03
	SER @ 4	6.25e-02	3.58e-02	2.31e-02	1.37e-02	8.03e-03
	SER @ 5	5.96e-02	3.36e-02	2.19e-02	1.28e-02	7.39e-03
	SER @ 6	4.95e-02	2.72e-02	1.78e-02	1.15e-02	6.02e-03
	SER @ 7	3.08e-02	1.72e-02	1.10e-02	7.18e-03	4.02e-03

Table A-7. Every fourth symbol known to the decoder, (11, 1/4) first code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		Every fourth symbol known to the decoder. (dB added = 1.25)		
(11, 1/4) First Code	23	10001011011 11101010001 10110101001 11000100101				
SYMBOL SIZE	SNR (dB)	0.0	0.3	0.5	0.7	0.9
1-Bit	BER	4.34e-03	2.20e-03	1.34e-03	9.42e-04	4.63e-04
	SER @ 1	4.17e-03	2.15e-03	1.27e-03	9.81e-04	4.39e-04
	SER @ 2	4.47e-03	2.28e-03	1.40e-03	9.31e-04	4.94e-04
	SER @ 3	4.36e-03	2.16e-03	1.35e-03	9.15e-04	4.56e-04
4-Bit	BER	4.71e-03	2.54e-03	1.61e-03	9.55e-04	6.12e-04
	SER @ 1	9.96e-03	5.41e-03	3.40e-03	2.09e-03	1.34e-03
	SER @ 2	1.15e-02	6.40e-03	4.13e-03	2.47e-03	1.66e-03
	SER @ 3	9.86e-03	5.37e-03	3.54e-03	2.10e-03	1.33e-03
8-Bit	BER	8.05e-03	4.35e-03	2.90e-03	1.91e-03	1.21e-03
	SER @ 1	2.00e-02	1.12e-02	7.83e-03	5.21e-03	3.16e-03
	SER @ 2	2.66e-02	1.49e-02	1.00e-02	6.90e-03	4.28e-03
	SER @ 3	2.01e-02	1.13e-02	7.52e-03	5.32e-03	3.33e-03

Table A-8. Every other symbol known to the decoder, (11, 1/4) first code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		Every other symbol known to the decoder. (dB added = 3.01)		
(11, 1/4) First Code	23	10001011011 11101010001 10110101001 11000100101				
SYMBOL SIZE	SNR (dB)	0.0	0.3	0.5	0.7	0.9
1-Bit	BER	8.40e-04	4.83e-04	3.42e-04	2.82e-04	1.27e-04
	SER @ 1	8.40e-04	4.83e-04	3.42e-04	2.82e-04	1.27e-04
4-Bit	BER	7.54e-04	4.64e-04	2.98e-04	2.25e-04	1.33e-04
	SER @ 1	2.16e-03	1.36e-03	8.92e-04	7.10e-04	4.20e-04
8-Bit	BER	1.41e-03	8.30e-04	5.86e-04	3.76e-04	2.79e-04
	SER @ 1	5.10e-03	3.26e-03	2.32e-03	1.55e-03	1.10e-03

Table A-9. No symbol known to the decoder, (15, 1/4) NASA code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		No symbol known to the decoder. (dB added = 0.00)	
(15, 1/4) NASA Code	35	100010110011001 100111010100101 111011011110011 101110101000111			
SYMBOL SIZE	SNR (dB)	-0.2	0.0	0.3	0.5
1-Bit	SER	4.88e-02	2.80e-02	1.09e-02	5.37e-03
4-Bit	SER	9.36e-02	5.41e-02	2.11e-02	1.05e-02
8-Bit	SER	1.04e-01	6.07e-02	2.40e-02	1.22e-02

Table A-10. Every eighth symbol known to the decoder, (15, 1/4) NASA code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		Every eighth symbol known to the decoder. (dB added = 0.58)	
(15, 1/4) NASA Code	35	100010110011001 100111010100101 111011011110011 101110101000111			
SYMBOL SIZE	SNR (dB)	-0.2	0.0	0.3	0.5
1-Bit	BER	9.77e-03	5.24e-03	1.87e-03	9.48e-04
	SER @ 1	9.46e-03	5.20e-03	2.06e-03	9.96e-04
	SER @ 2	9.60e-03	5.32e-03	1.78e-03	9.52e-04
	SER @ 3	9.95e-03	5.34e-03	1.93e-03	9.76e-04
	SER @ 4	9.89e-03	5.14e-03	1.81e-03	9.08e-04
	SER @ 5	9.86e-03	5.34e-03	1.84e-03	9.80e-04
	SER @ 6	9.88e-03	5.23e-03	1.84e-03	9.60e-04
	SER @ 7	9.78e-03	5.10e-03	1.80e-03	8.64e-04
4-Bit	BER	1.11e-02	6.12e-03	2.53e-03	1.06e-03
	SER @ 1	1.88e-02	1.00e-02	4.05e-03	1.50e-03
	SER @ 2	2.18e-02	1.26e-02	5.10e-03	2.05e-03
	SER @ 3	2.40e-02	1.32e-02	5.57e-03	2.61e-03
	SER @ 4	2.38e-02	1.39e-02	5.71e-03	2.54e-03
	SER @ 5	2.34e-02	1.33e-02	6.00e-03	2.50e-03
	SER @ 6	2.15e-02	1.22e-02	5.30e-03	2.21e-03
	SER @ 7	1.80e-02	9.68e-03	4.18e-03	1.71e-03
8-Bit	BER	1.50e-02	8.80e-03	3.80e-03	2.13e-03
	SER @ 1	2.29e-02	1.38e-02	4.61e-03	3.33e-03
	SER @ 2	3.40e-02	2.04e-02	8.38e-03	5.06e-03
	SER @ 3	4.05e-02	2.46e-02	1.10e-02	6.59e-03
	SER @ 4	4.29e-02	2.54e-02	1.16e-02	6.69e-03
	SER @ 5	4.12e-02	2.47e-02	1.15e-02	6.27e-03
	SER @ 6	3.47e-02	1.98e-02	9.82e-03	4.83e-03
	SER @ 7	2.27e-02	1.27e-02	6.18e-03	3.10e-03

Table A-11. Every fourth symbol known to the decoder, (15, 1/4) NASA code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		Every fourth symbol known to the decoder. (dB added = 1.25)	
(15, 1/4) NASA Code	35	100010110011001 100111010100101 111011011110011 101110101000111			
SYMBOL SIZE	SNR (dB)	-0.2	0.0	0.3	
1-Bit	BER	2.11e-03	1.19e-03	4.13e-04	
	SER @ 1	2.14e-03	1.24e-03	3.92e-04	
	SER @ 2	2.09e-03	1.14e-03	4.12e-04	
	SER @ 3	2.10e-03	1.18e-03	4.34e-04	
4-Bit	BER	2.28e-03	1.37e-03	5.70e-04	
	SER @ 1	4.48e-03	2.66e-03	1.14e-03	
	SER @ 2	4.98e-03	3.19e-03	1.27e-03	
	SER @ 3	4.38e-03	2.62e-03	1.15e-03	
8-Bit	BER	3.97e-03	2.50e-03	1.16e-03	
	SER @ 1	9.23e-03	5.71e-03	2.58e-03	
	SER @ 2	1.13e-02	7.14e-03	3.63e-03	
	SER @ 3	8.86e-03	5.62e-03	2.74e-03	

Table A-12. Every other symbol known to the decoder, (15, 1/4) NASA code

CODE	FREE DISTANCE	GENERATING POLYNOMIAL		Every other symbol known to the decoder. (dB added = 3.01)	
(15, 1/4) NASA Code	35	100010110011001 100111010100101 111011011110011 101110101000111			
SYMBOL SIZE	SNR (dB)	-0.2	0.0	0.3	
1-Bit	BER	1.69e-04	8.10e-05	2.80e-05	
	SER @ 1	1.69e-04	8.10e-05	2.80e-05	
4-Bit	BER	2.39e-04	9.50e-05	5.40e-05	
	SER @ 1	4.92e-04	1.96e-04	1.36e-04	
8-Bit	BER	4.03e-04	2.39e-04	1.18e-04	
	SER @ 1	1.06e-03	6.80e-04	3.28e-04	