# PERFORMANCE OF SOME BLOCK CODES ON A GAUSSIAN CHANNEL*

**L.D. BAUMERT AND R.J. McELI ECE**
**Caltech's Jet Propulsion Laboratory**

**Summary.** In this paper we use a recent technique of Chase to evaluate the performance of several block codes, notably BCH codes of lengths 63 and 95, on a Gaussian channel. We conclude that such codes are close to being serious competition for convolutional codes on this channel.

**1. Introduction.** In this paper, aided by a recent technique of Chase [2], we shall evaluate the performance of several fairly long binary block codes on a wideband additive Gaussian channel. We were motivated to do this by the following considerations.

The current best practical coding system for the Gaussian channel is a convolutional code with Viterbi decoding. Unfortunately the complexity of Viterbi's algorithm is such that a code with constraint-length much more than 8 cannot be implemented practically. And while extremely good convolutional codes of quite long constraint-length are known, there is no known way to decode them in a manner that yields performance superior to the short-constraint length/Viterbi algorithm. (Sequential decoding is a practical method to reduce the bit error probability, but the bit erasure probability remains high. We choose to regard erasures as no better than errors.)

It is true that the performance of a convolutional code can be surpassed at relatively large signal-to-noise ratios by hybrid concatenation schemes. But the performance of the concatenation scheme is highly sensitive to the performance of the inner code at low signal-to-noise ratios. Hence a significant improvement in the inner code at bit error probabilities of the order $10^{-2}$ - $10^{-3}$ would yield a significantly improved concatenation system.

There being no breakthrough in the decoding of convolutional codes on the horizon, we turned to the other known class of codes, the block codes. Now block codes have never been serious competition for convolutional codes on the Gaussian for one simple reason: the known decoding algorithms for powerful block codes depend on the reception of

*This paper presents the results of one phase of research carried out at the Jet Propulsion Laboratory, California Institute of Technology, under Contract No. NAS 7-100, sponsored by the National Aeronautics and Space Administration.

binary data. (This is not true for sequential and Viterbi decoding.) And if a binary quantizer is added to a Gaussian channel its capacity is reduced by a factor of $\pi/2$ (~ 2dB). Recently several authors Weldon [5], Chase [2], Dorsch [3], Sundberg [4]) have begun to develop algorithms to alleviate this problem. In particular Chase has given a very interesting. algorithm for decoding any block code on a Gaussian channel, provided it already possesses a good binary decoding algorithm.

Intrigued by Chase's ideas, we set out to try his algorithm on the most powerful practical class of binary codes, the BCH codes with Berlekamp's decoding algorithm. In section 2, we describe Chase's algorithm and our selection of candidate codes. In section 3, we present our experimental results, and in section 4 we state our conclusions.

## 2. Chase's Algorithm.

We begin with an (n,k) binary (symbols: +1,-1) of rate R = k/n and minimum distance d. We assume there exists a "practical" algorithm which maps a binary vector at Hamming distance $\leq$ t from some codeword onto that codeword (necessarily $2t+1 \leq d$). [This is called the binary decoding algorithm (BDA).]

A codeword $\underline{x} = (x_1,x_2,...,x_n)$ is transmitted over the channel and is received as $\underline{y} = (y_1,y_2,...,y_n)$, $y_i = x_i + Z_i$, where $Z_1,Z_2,...,Z_n$ is a sequence of independent, identically distributed, Gaussian random variables with mean 0, variance $\sigma^2$. The bit signal-to-noise ratio $\gamma = E_b/N_o$ is given by

$$\gamma = 1/(2R\sigma^2) \quad . \tag{1}$$

If we were forced to guess the value of $x_i$, given only $y_i$, we would guess

$$\hat{x}_i = \begin{cases} +1 & \text{if } y_i > 0 \\ -1 & \text{if } y_i < 0 \end{cases} \quad . \tag{2}$$

Our confidence in this estimate is small, however, if $|y_i|$ is small, and so we define the (relative) reliability of the estimate 2) by

$$\alpha_i = |y_i| \quad . \tag{3}$$

Chase's decoding technique can now be described. It depends on a fixed parameter c, to be specified later. First, the received vector $\underline{y}$ is "hard quantized" into the binary vector $\underline{\hat{x}} = (\hat{x}_1,...,\hat{x}_n$. Next, the c "least reliable" components of $\underline{\hat{x}}$ are located , i.e., components $x_i$, $i \in I_c$, where $I_c \leq \{1,2,...,n\}$ is a set of cardinality c, and

(4) $$i \in I_c,\ j \notin I_c \Rightarrow \alpha_i \leq \alpha_j \quad .$$

For notational convenience, we shall hereafter assume $\alpha_1 \leq \alpha_2 \leq \cdots \alpha_n$, so that $I_c = \{1,2,\ldots,c\}$.

Next, $2^c$ binary vectors $\hat{x}(\epsilon_1,\epsilon_2,\ldots,\epsilon_c) = \hat{x}(\underline{\epsilon})$ are produced, where

(5) $$\hat{x}(\epsilon_1,\ldots,\epsilon_c) = (\epsilon_1,\epsilon_2,\ldots,\epsilon_c,\hat{x}_{c+1},\hat{x}_{c+2},\ldots,\hat{x}_n) \quad ,$$

and each $\epsilon_i = \pm 1$. Next, each $\hat{x}(\underline{\epsilon})$ is decoded by the BDA. The result of each application of the BDA will either be a codeword $\underline{x}$ or nothing (decoder failure). Hence after all $2^c$ applications oFthe BDA we will have produced a set $D = \{\underline{x}^{(1)}, \underline{x}^{(2)},\ldots,\underline{x}^{(c')}$ of at most (usually many fewer than) c distinct codewords. The decoder's output is that codeword for which the inner product $\underline{x}^{(i)} \cdot \underline{y}$ is largest.

If c = 0, this algorithm is merely hard-decision decoding of the given code. On the other hand if c = n, the algorithm is maximum-likelihood (but extremely inefficient). The idea is that ML decoding can be closely approximated by some much smaller value of c. Chase suggests c= t (the number of errors being corrected), but our simulations have shown this value of c to be too small, in general.

This algorihm can fail in two distinct ways. First, the transmitted codeword $\underline{x}$ may not lie in the set D. This type of error, which dominates the algorithm's performance for small values of c, we call a type I error. Second, even if the set D contains $\underline{x}$, it may happen that $\underline{x}^{(1)} \cdot \underline{y}$ is largest for some $\underline{x}^{(i)} \neq \underline{x}$. This type of error, which dominates the algorithm's performance for larger calues of c, we call a type II error.

Note that while the probability of a type II error is extremely difficult to compute, it is relatively easy to compute the probability of a type I error. For example, if the code is a t-error correcting BCH code of length n, it is safe to assume that the BDA will fail if more than t binary errors occur. Then the probability of a type I error can be computed as follows. Let $X_1,X_2,\ldots,X_n$ be n independent, mean +1, variance $\sigma^2$ Gaussian random variables. Let $X_{(1)} \leq X_{(2)} \leq \ldots \leq X_{(n)}$ be a reordering of the $X_i$'s into increasing order. Then the probability of a type I error is just the probability that $-X_{(t+1)} \geq |X_{(t+1+c)}|$. This probability, in turn, can be written as a messy but computable multiple integral. Thus to some extent the performance of Chase's algorithm can be computed analytically. Unfortunately, as c increases, the approximation suggested by the above argument becomes increasingly poor and it is necessary to resort to extensive Monte Carlo simulation to obtain performance curves, for it is the presence of type I errors that makes

Chase's algorithm inferior to MLD. When a substantial fraction of the decoder's errors are type II, it is clear that MLD has been closely approximated.

Before giving our numerical results, we include a brief discussion of the selection of the codes to be evaluated. Our first requirement was that an efficient BDA be available. This basically limited the available codes to be either very short, or BCH codes. But we still needed some further preliminary figure of merit to screen candidate codes. Now it is known (see Chase [2], for example) that the bit error probability of a block code $P_e$ and the bit signal-to-noise ratio $\gamma = E_b/N_o$ are asymptotically related by the formula

$$\lim_{\gamma \to \infty} \log P_e = -Q \quad ,$$

where $Q = R \cdot d$, the product of the code's rate and minimum distance. (This formula also holds for convolutional codes with d = the free distance). Hence among codes of approximately equal implementational complexity we tended to favor those with large values of Q. Incidentally, since Berlekamp [1] showed that long BCH codes have $d \sim 2n \log R^{-1}/\log_2 n$, it follows that for large n the "best" BCH codes for a Gaussian channel have R = 1/3 = .368. Similarly if the asymptotic relationship between d and n for the best block codes is given by the Gilbert bound $R = 1 - H_2(d/n)$, where $H_2$ is the binary entropy function, it follows that the rate of the best long block codes for the Gaussian channel is R = .379 or so.

## 3. Numerical Results

We present in this section the results of applying Chase's algorithm to four binary codes. In each of our figures, we plot $\log_{10}(P_e)$ vs. $10 \log_{10}(\gamma)$ where $P_e$ is the bit error probability and $y = E_b/N_o = 1/(2R\sigma^2)$ is the bit signal-to-noise ratio. For reference purposes we have also included two "standard" dashed curves in each figure. The upper curve with the smaller slope) represents "no coding" which can be viewed as a (1,1) block code with Q = 1. The lower (steeper) curve represents the performance of a constraint-length 7, rate 1/2 ($d_{free} = 10$, Q = 5) convolutional code with MLD (Viterbi) decoding. This coded system is a good approximation to the "state of the art" on the Gaussian channel; indeed it will be implemented on all of NASA's Mariner-class interplanetary spacecraft after 1976.

**(23,12) Golay code** (figure 1). Although the MLD performance of this code was already evaluated by Chase, indeed even earlier by Weldon, we include it for completeness. The BDA we used was merely syndrome-table lookup and so was very fast. The nice thing about such a short code is that MLD is already extremely closely approximated (as Chase pointed out) at c = 4. In fact at c = 4 and $\gamma$ = 2.1 dB, more than 70% of all decoder errors were type II. At c = 7 the percentage is over 95%. It is interesting to note that this code is superior to the convolutional code if $\gamma < \sim 2$ dB and not much inferior out to about 3 dB.

Since the NASA application mentioned above is at about 2.5 dB, the Golay code with a c=4 Chase decoder could have been a non-trivial competitor for the convolutional code.

**(32,16) second-order Reed-Muller code** (figure 2); d=8, Q=4. The BDA was Reed's majority-logic algorithm. Already for such a short code we see that MLD is not approached nearly as quickly., For example at $\gamma = 2.0$ dB, at c = 4 only 5% of the errors were type II, and at c = 7 the percentage is still only 47%. The conclusion is to obtain near-MLD performance c = 7 or 8 is required.

**(63,36) 5-error correcting BCH code** (figure 3). Here d = 11, Q = 6.29. Here we see that at c = 10, the code is definitely superior to the convolutional code. Furthermore MLD is closely approximated, since at c = 10 and $\gamma = 2.1$ dB, 33% of the errors are type II.

**(95,39) 9-error correcting shortened BCH code** (figure 4). Here d = 19, Q = 7.8. Here we do not know where the MLD curve lies, for even at c = 20 no type II errors occurred! This fact, once discovered empirically, made the computation of the c = 20 curve somewhat easier, as mentioned in section 2.

## 4. Conclusions and Suggestions for Future Research

First, we conclude that there are many block codes of length $\leq 100$ with extremely attractive MLD performance on a Gaussian channel. (No doubt there are many whose performance is even much better than those in figures 1-4; but at present it is impossible to estimate MLD performance if a BDA for the code does not exist.) Second, we conclude that BCH codes decoded via Berlekamp's BDA and Chase's idea are "close" to being practical competitors to short-constraint length convolutional codes with Viterbi decoding. This conclusion is based partly on the results of this paper and partly on private communications with Berlekamp, who informs us that his algorithm can be modified to accommodate "bit Chasing" with c =15 and bit rates of the order of $10^5$/sec.

Finally, we believe this subject to be extremely ripe for further research. One question which definitely merits attention is this. It is possible to view Chase's algorithm as a way to help the binary decoder by "guessing" the locations of some of the errors on the basis of the reliabilities $\alpha_i$. Chase suggests guessing all possible error patterns which are wholly confined to the least reliable c digits of the received codeword, a total of $2^c$ guesses. Surely, given a willingness to make $2^c$ guesses, there must be a "better" set of guesses to try. But what is a better set? We do not know. Also, since it is clear that the value of c necessary to approximate MLD increases rapidly with n, we are motivated to find good BDA's for powerful <u>short</u> codes. For example, the (48,24) quadratic residue code has d =12, Q = 6, and MLD could no doubt be well approximated by c = 8 or so. But so far as we know, no decent BDA exists for this code!

**Acknowledgments.** The authors happily acknowledge the assistance rendered us during the preparation of this paper by E. R. Berlekamp, H. C. Rumsey, Jr., E. R. Rodemich, and L. R. Welch.

**References.**

[Note: all references are to the IEEE Trans. Information Theory (IT)]

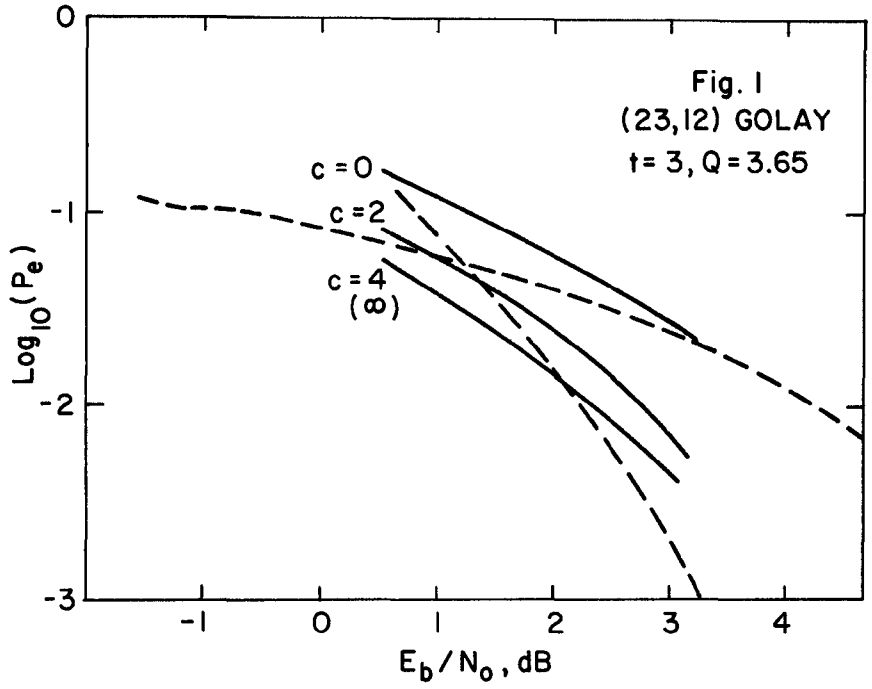
[1] Berlekamp,_J. R., "Long primitive BCH codes have distance d ~ 2nℓn $R^{-1}$/log n...," IT-18 (1972), pp. 415-426.

[2] Chase, D., "A class of algorithms for decoding block codes with channel measurement information," IT-18 (1972), pp. 170-182.

[3] Dorsch, B. G., "A decoding algorithm for binary block codes and J-ary output channels," IT-20 (1974), pp. 391-394.

[4] Sundberg, C. E., "One-step majority-logic decoding with symbol reliability information" IT-21 (1975), pp. 236-242.

[5] Weldon, E. J., "Decoding binary block codes on Q-ary output channels," IT-17 (1971), pp. 713-718.



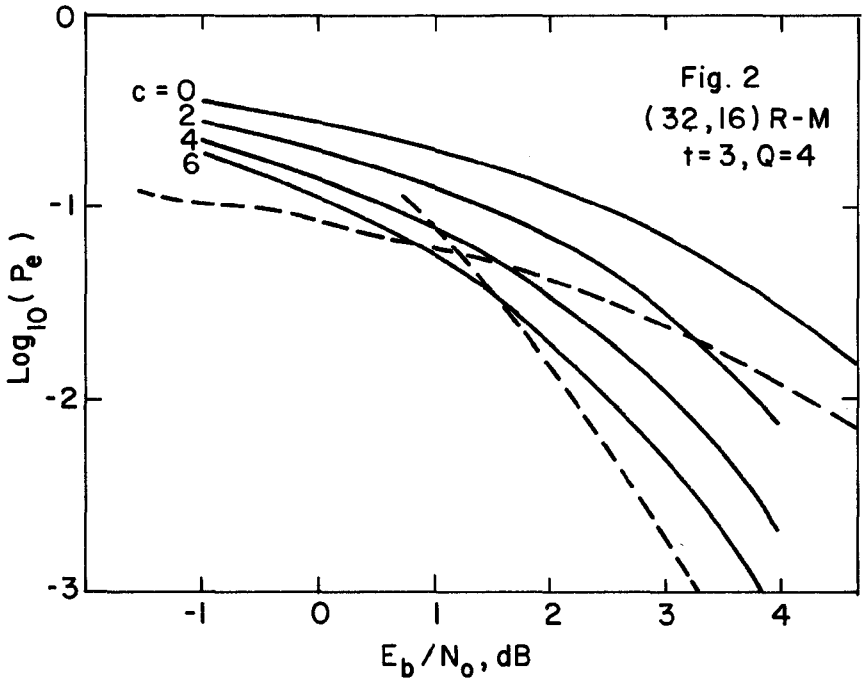


Fig. 2
(32,16) R-M
t=3, Q=4
c = 0
2
4
6
0
-1
-2
-3
-1
0
1
2
3
4
$Log_{10}(P_e)$
$E_b/N_o$, dB

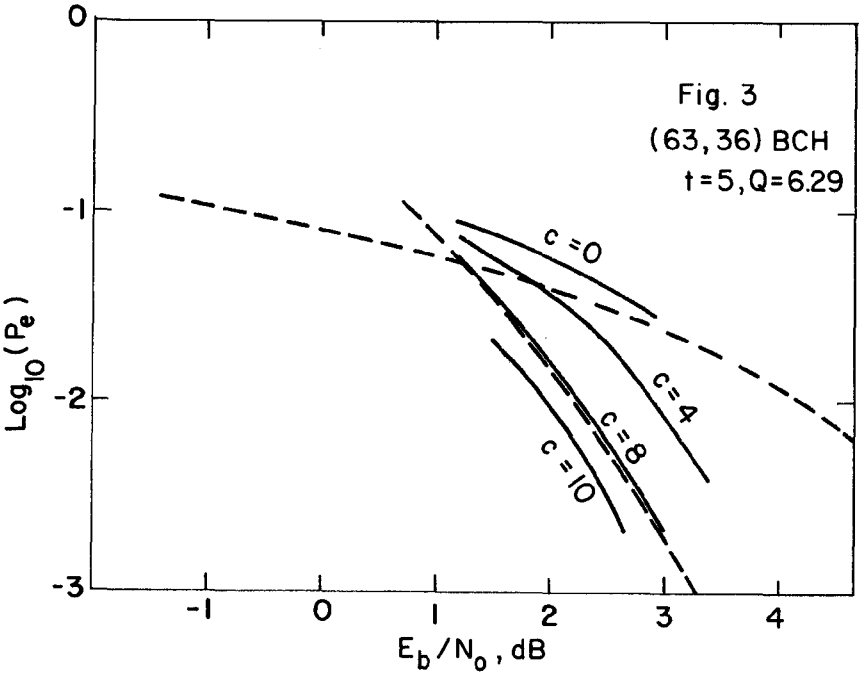


Fig. 3
(63,36) BCH
t=5, Q=6.29
c=0
c=4
c=8
c=10
0
-1
-2
-3
-1
0
1
2
3
4
$Log_{10}(P_e)$
$E_b/N_o$, dB

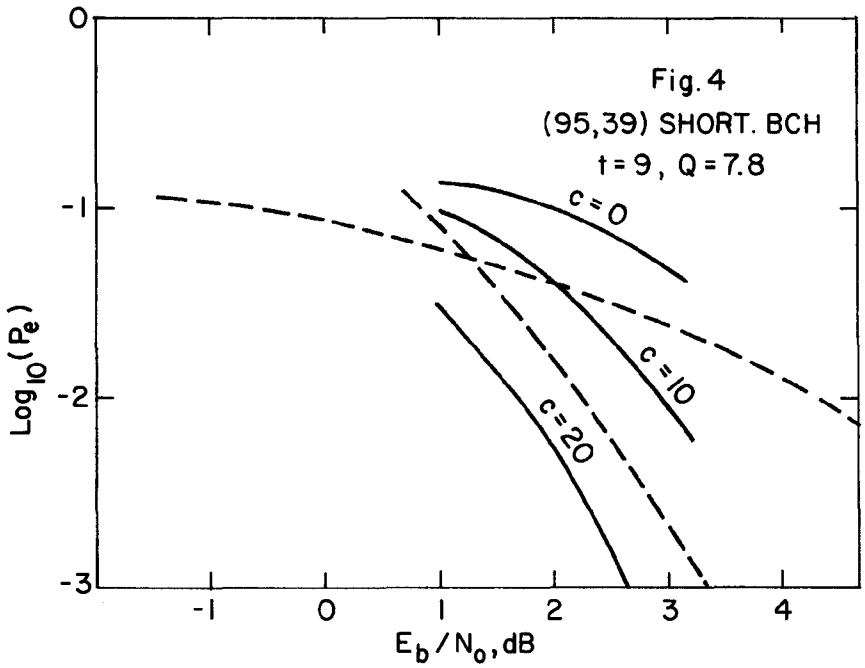


Fig. 4
(95,39) SHORT. BCH
t = 9, Q = 7.8
c = 0
c = 10
c = 20
0
-1
-2
-3
-1
0
1
2
3
4
$Log_{10}(P_e)$
$E_b / N_0$, dB