RAPPORT TECHNIQUE

EPM/RT-88/1

HIGH-RATE PUNCTURED CONVOLUTIONAL CODES*

David Haccoun, Eng. Ph.D.
Professor of Electrical Engineering
École Polytechnique de Montréal
Montréal (Québec) Canada

Guy Bégin, Eng. M.Sc.A.
Doctoral Student
Department of Electrical Engineering
École Polytechnique de Montréal
Montréal (Québec) Canada

Janvier 1988

* This research has been supported in part by the National Sciences and Engineering Research Council of Canada and by the Fonds FCAR of Quebec.

Dépôt légal, 1er trimestre 1988
Bibliothèque nationale du Québec
Bibliothèque nationale du Canada

Pour se procurer une copie de ce document, s'adresser au:

Éditions de l'École Polytechnique de Montréal
École Polytechnique de Montréal
Case postale 6079, Succursale A
Montréal (Québec) H3C 3A7
(514) 340-4000

Compter 0,10$ par page (arrondir au dollar le plus près) et ajouter 3,00$ (Canada) pour la couverture, les frais de poste et la manutention. Régler en dollars canadiens par chèque ou mandat-poste au nom de l'École Polytechnique de Montréal. Nous n'honorerons que les commandes accompagnées d'un paiement, sauf s'il y a eu entente préalable dans le cas d'établissements d'enseignement, de sociétés ou d'organismes canadiens.

HIGH-RATE PUNCTURED CONVOLUTIONAL CODES

David Haccoun and Guy Bégin
Department of Electrical Engineering
Ecole Polytechnique de Montréal

ABSTRACT

This paper investigates high rate punctured convolutional codes suitable for Viterbi and sequential decoding. Results on known short constraint length codes ($K<10$) discovered by others are extended. Weight spectra and upper bounds on the bit error probability of the best known punctured codes having constraint lengths $3\leq K\leq 9$, and coding rates $2/3\leq R\leq 7/8$ are provided. Newly discovered rates 2/3 and 3/4 long constraint length punctured convolutional codes with $10\leq K\leq 23$, are provided together with the leading terms of their weight spectra and their bit error performance bounds. Some results of simulation with sequential decoding are given.

HIGH-RATE PUNCTURED CONVOLUTIONAL CODES

David Haccoun and Guy Bégin

1. INTRODUCTION

Over the last two decades a very substantial research effort has been devoted to the theoretical investigation, performance analysis and applications of coding and decoding techniques. A huge literature exists on the subject [1]-[6],[7]. In Forward Error Correction (FEC) systems the main difficulties usually reside at the decoder, and one of the problems standing in the way of the widespread use of FEC techniques is the material realization of powerful decoders that can operate at high data rates, deliver low error probabilities while being practical and not too complex to implement.

For discrete memoryless channels where the noise is essentially white (such as the space and satellite channels), systems using convolutional encoding at the transmitting end of the link and probabilistic decoding at the receiving end are among the most attractive means of approaching the reliability of communication predicted by the Shannon theory; these systems provide substantial coding gains while being readily implementable. Probabilistic decoding refers to techniques where the decoded message is obtained by probabilistic considerations and utilization of the channel statistics, rather than by a fixed set of code-dependent algebraic operations. Moreover no particular algebraic structure is imposed on the code which may even be chosen at random.

The two principal probabilistic decoding for convolutional codes are Viterbi decoding [8] and sequential decoding [9]. In both of these techniques the decoder attempts to determine the most likely path $\underline{U} = (\underline{u}_1, \underline{u}_2, \underline{u}_3.)$ through a graph (tree or trellis) in which the branches $\{\underline{u}_y\}$ of the paths are assigned metric values $\{\gamma_j\}$, that is likelihood values of having been transmitted. The objective of the decoder is thus to find, with the highest reliability and the minimum computational effort, the tree of trellis path that has the largest cumulative metric $\Gamma = {}_{j}\Sigma\gamma_j$ over all possible transmitted paths.

Viterbi and sequential decoding are quite different techniques with different error performances, different inherent problems and somewhat different domains of applications. They have developed independently and appear to be the opposite for determining the most likely information sequence given the received sequence. The Viterbi algorithm exploits the path remergers of the trellis structure of the code and exhaustively examines all distinct paths at every trellis level, whereas a sequential decoder operates on the tree structure of the code and follows only the single path that is currently the most likely, without searching the entire tree. As a consequence, the computational effort is constant but large for Viterbi decoding, whereas it is on the average typically very small, but highly variable for sequential decoding [10].

This computational variability constitutes one of the principal drawbacks of sequential decoding and several methods have been proposed to alleviate the problem [11], [12]. However an interesting property of sequential decoding is that the average number of computations to decode one information bit, C_{AV}, is practically independent of the encoder memory length M, which may thus be chosen at will. This property is not shared by Viterbi decoding where the computational effort and the complexity of the decoder both grow exponentially with the memory of the code. Now since the error probability of either decoder decreases exponentially with the memory of the code, any further improvement on the error performance by increasing the memory of the code becomes quickly expensive (even impossible) for Viterbi decoding. Consequently, Viterbi decoding is practically limited to small memory length codes ($M \leq 6$), whereas sequential decoders can operate with M in the order of 40 or more [13].

Soft-decision Viterbi decoding with rate $R = 1/2$ and memory $M = 6$ codes can provide a coding gain of 5.2 dB in required Eb/No at a bit error rate $P_B = 10^{-5}$, whereas soft-decisions. Sequential decoding can provide more than 7 dB gain. In practice a given coding gain, say 5 dB, can be translated as either a 5 dB reduction of the transmitting power for the same data rate and error performance as a noncoded system, or as an increase of the data rate by a factor of $10^{0.5} = 3.16$. Depending on the application, each (or a combination) of these alternatives may be quite attractive in improving the overall system design, especially in satellite communications where the transmitting power is at a premium.

By far, error control techniques using convolutional codes have been dominated by low rate R=1/V codes. Optimal low rate codes, providing large coding gains are available in the literature [1]-[6], and practical implementations of decoders (Viterbi or Sequential) exist for decoding rates in the range of 10 to 40 Mbits/s [14][15][16].

However, as the trend for ever increasing data transmission rates and high error performance continue while conserving bandwidth, the needs arise for good high rate R=b/V convolutional codes as well as practical encoding and decoding techniques for these codes. Unfortunately a straight forward application of Viterbi and sequential decoding to high rate codes becomes very rapidly impractical as the coding rate increases. Furthermore a conspicuous absence prevails in the literature for good non systematic long constraint length (K>10) convolutional codes with rates R larger than 2/3.

A significant breakthrough occured with the advent of high rate punctured convolutional codes [18], [19], [20], [22], [7], [23], [24], [25] where the inherent difficulties of coding and decoding of high rate codes can be almost entirely circumvented.

In this paper we present Viterbi and sequential decoding of high rate punctured convolutional codes. Extending results on known short constraint length codes (K<10) discovered by others [18], [19], [20], we provide the weight spectra and upper bounds on the bit error probability of the best known punctured codes having constraint lengths $3 \leq K \leq 9$, and coding rates $2/3 \leq R \leq 7/8$. Newly discovered rates 2/3 and 3/4 long constraint length punctured convolutional codes with $10 \leq K \leq 23$, are provided together with the leading terms of their weight spectra and their bit error performance bounds.

The paper is structured as follows: Section 2 introduces the encoding of punctured codes and section 3 Viterbi decoding of these codes. Extensions to sequential decoding is presented in section 4. The search for good punctured code is the object of section 5. This section contains the principal new results of the paper: weight spectra of short constraint length codes, new good rate 2/3 and 3/4 long constraint length punctured codes, together with their weight spectra and bit error performance bounds. Finaly some simulation results for sequential decoding are given in section 6, demonstrating the advantages of using high rate punctured convolutional codes over usual codes of the same rate.

2. BASIC CONCEPTS OF PUNCTURED CONVOLUTIONAL CODES

We recall that the decoding complexity of either Viterbi or sequential decoding increases rapidly with the coding rates R=b/V, b>1, b<V, so that a straightforward application of these techniques becomes rather combersome at rates larger that 1/2 [26], [27]. However, by using punctured high rate codes, the difficulty can be entirely circumvented. Viterbi or sequential decoding of rates b/V convolutional codes is hardly more complex that for rates 1/V codes and, furthermore, either technique may be easily applicable to adaptive and variable rate decoding [18], [19], [20], [21], [22], [7], [23], [24].

2.1 Encoding of punctured codes

A punctured convolutional code is a high rate code obtained by the periodic elimination of specific code symbols from the output of a low rate encoder. The resulting high rate code depends on both the low rate code called original or mother code, and on both the number and specific positions of the punctured symbols. The pattern of punctured symbols is called the perforation pattern of the punctured code, and is conveniently described in matrix form.

Consider constructing a high rate R=b/V convolutional punctured code from a given original code of any low rate $R=1/V_0$. From every V_0b code symbols corresponding to the encoding of b information bits by the original encoder, a number $S=(V_0b-V)$ symbols are deleted according to some chosen perforation pattern. The resulting rate is then $R=b/(V_0b-S)$ which is equal to the desired target rate R=b/V. By a judicious choice of the original low rate code and perforation pattern any rate code may be thus obtained [24], [25].

For example Fig. 1 shows the trellis diagram of a rate 1/2, constraint length K=3 code where every third symbol is punctured (indicated by X on every second branch on the diagram). Reading this trellis two branches at a time and redrawing it as in figure 2, we see that it corresponds to a rate 2/3 constraint length 3 code. A punctured rate 2/3 code has therefore been obtained from an original rate 1/2 encoder.

Obviously puncturing a code reduces its free distance, and hence a punctured code cannot achieve the free distance of its original code. Although the free distance of a code increases as its rate decreases, using original codes whith rates $1/V_0$ lower than 1/2 does not always warrant punctured codes with larger free distances since for a given b and V the proportion of deleted symbols, $S/V_0b=1-(V/V_0b)$ also increases with V_0. Consequently good results and ease of implementation tend to favor the use of rate 1/2 original codes for generating good punctured codes with coding rates of the form R=(V-1)/V suitable for Viterbi decoding [18], [19], [20], [23], or sequential decoding [21], [21], [7], [24], [25]. Further results on both short (K<10) and long memory codes (K>10) are provided in this paper.

An encoder for high-rate b/V punctured codes is shown shematically in Fig. 3. It consists of an original low-rate convolutional encoder of rate $R=1/V_0$ followed by a sampler. Given the V_0 generators $\underline{G}_j$ of the low rate encoder of memory M, where M=(K-1), and where

$$\underline{G}_j = (g_{0j}, g_{1j}, g_{2j}, \ldots, g_{Mj}), \quad j = 1, 2, \ldots, V_0 \qquad (1)$$

an information bit u_i is encoded as a sequence (or branch) of length V_0 symbols,

$$\underline{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iv0}) \qquad (2)$$

where the code symbols xi_j are given by

$$x_{ij} = \mathop{}_{n=0}^{M} u_{i-n}\, g_{nj}, \qquad j = 1,2,\ldots V_0 \tag{3}$$

An information sequence $\underline{U}=(u_1, u_2, u_3,\ldots)$ is therefore encoded by the low rate encoder as the encoded sequence $\underline{X}=(\underline{x}_1, \underline{x}_2, \underline{x}_3,\ldots)$ which is then properly modulated and transmitted over a noisy channel.

PERFORATION PATTERNS

As shown in Fig. 3, following the low rate $1/V_0$ encoder a symbol selector or sampler is used to periodically delete from each consecutive b branches a number $S=(V_0b-V)$ symbols, according to the perforation pattern, thus yielding a punctured code of rate $R=b/V$. The matrix [P] expressing the perforation pattern has V_0 rows and b columns, and its elements are only 0's and 1's, corresponding to the deleting or keeping of the corresponding code symbol of the original encoder. Clearly both the punctured code and its rate can be varied by suitably modifying the elements of the perforation matrix. For example, starting from an original rate 1/2 code, the perforation matrix of the rate 2/3 punctured code of Fig. 1 is given by

$$[P_1] = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \tag{4}$$

whereas a rate 4/5 code could be obtained using the perforation matrix

$$[P_2] = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \end{bmatrix} \tag{5}$$

Matrix $[P_2]$ above indicates that in deriving the rate 4/5 punctured code, periodically, out of every four branches on any path of the original code, no code symbol is deleted from the first branch, the second symbol on the second branch is deleted, and on both the third and fourth branch the first symbol is deleted. Clearly, by manipulating the perforation matrix a wide range of possibilities exists in deriving punctured codes.

Variable rate coding is readily obtained if all punctured rates of interest are obtained from the same low rate encoder. Only the perforation matrices need to be modified accordingly as illustrated by (4) and (5).

Variable rate coding may be further specialized by adding the restriction that all the code symbols of the high rate punctured codes are required by the lower rate codes. This restriction implies minimal modifications of the perforation matrix as the code rates vary. Punctured codes satisfying this restriction are said to be <u>rate compatible</u>. For example starting from an original code of rate $1/V_0=1/2$, the perforation matrices $[P_1]$ through $[P_4]$ given below generate a family of rate compatible punctured codes of rates 5/6 through 5/9 respectively.

$$[P_1]=\begin{bmatrix}1&0&1&0&1\\1&1&0&1&0\end{bmatrix}\;[P_2]=\begin{bmatrix}1&1&1&0&1\\1&1&0&1&0\end{bmatrix}\;[P_3]=\begin{bmatrix}1&1&1&1&1\\1&1&0&1&0\end{bmatrix}\;[P_4]=\begin{bmatrix}1&1&1&1&1\\1&1&1&1&0\end{bmatrix} \tag{6}$$

To illustrate the concept, let

$$\underline{X}_0 = 11 \quad 01 \quad 10 \quad 00 \quad 11 \quad 01 \quad 10 \quad 11 \quad 01 \quad 10 \tag{7}$$

be a code sequence of length 10 information bits delivered by the original rate 1/2 encoder. Using the perforation matrices (6), the corresponding encoded sequences $\underline{X}_1$ to $\underline{X}_4$ of coding rates 5/6 to 5/9 respectively are obtained as given below, where an x indicates a punctured symbol:

$$\underline{X}_1 = 11 \quad x1 \quad 1x \quad x0 \quad 1x \quad 01 \quad x0 \quad 1x \quad x1 \quad 1x \tag{8}$$

$$\underline{X}_2 = 11 \quad 01 \quad 1x \quad x0 \quad 1x \quad 01 \quad 10 \quad 1x \quad x1 \quad 1x \tag{9}$$

$\underline{X}_3$ = 11 01 1x 00 1x 01 10 1x 01 1x (11)

$\underline{X}_4$ = 11 01 10 00 1x 01 10 11 01 1x (10)

Rate compatible punctured codes are especially useful in some rate adaptive ARQ/FEC applications since, as illustrated in the example above, only the incremental redundancy needs to be transmitted as the coding rate is decreased. Families of good non catastrophic short memory rate compatible punctured codes with rates varying from 8/9 to 1/4 have been found by Hagenauer [23].

Finaly another class of perforation patterns called orthogonal perforation patterns plays an important part in the search of specific punctured codes [25]. An orthogonal perforation pattern is a pattern in which any code symbol that is not punctured on one of the b branches is punctured on every other (b-1) branches of the resulting rate b/V punctured code. In an orthogonal perforation pattern the perforation matrix has V_0=V rows and b columns, with each row containing only one element 1. For example, the following orthogonal perforation pattern yields a rate 3/4 code from an original rate 1/4 code.

$$[P_0] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (12)$$

Orthogonal perforation patterns ensure that all the different generators of the original low rate $1/V_0$ code are used in deriving the desired punctured rate b/V code. In particular it can be shown that any punctured code can be obtained by means of an orthogonal perforation pattern [25]. Using this concept punctured codes identical to the best known usual rate 2/3 and 3/4 codes have been obtained by Bégin and Haccoun [25].

The basic notions of encoding punctured codes having been established, the problem of their decoding by both the Viterbi algorithm and sequential decoding is examined in the next two sections.

3. VITERBI DECODING OF PUNCTURED CODES

Given the received sequence from the channel Viterbi decoding consists essentially of computing the likelihood (that is the metric) that a particular sequence has been transmitted for every possible encoder state. For rate R=b/V codes, there are 2^b paths merging at every state and only the path with the largest metric is selected at each state. The process is repeated for each of the encoder states, so that clearly, as b increases the operations of the decoder become rapidly involved.

Now for punctured high rate b/V codes Viterbi decoding is hardly more complex than for the original low rate $1/V_0$ code from which the punctured codes are derived. The decoding is performed on the trellis of the original low rate code where the only modification consists of discarding the metric increments corresponding to the punctured code symbols. Given the perforation pattern of the code, this can be readily performed by inserting dummy data into the positions corresponding to the deleted code symbols. In the decoding process this dummy data is discarded by assigning them the same metric value (usually zero) regardless of the code symbol, 0 or 1. For either hard or soft-quantized channels, this procedure in effect inhibits the conventional metric calculation.

Therefore Viterbi codecs for high-rate punctured codes involve none of the complexity of the strightforward decoding of rate b/V codes. They can be implemented by adding relatively simple circuitry to the codecs of the original low rate $1/V_0$ code. Furthermore since a given low rate $1/V_0$ code can give rise to a large number of high rate punctured codes, the punctured approach leads to a very attractive realization of variable-rate

Viterbi decoding. In addition to the metric inhibition mentionned above, the only coding rate-dependent modification in a variable rate codec is the truncation path length which must be increased with the coding rate. All other operations remain essentially unchanged [19], [20].

BIT ERROR PERFORMANCE

For discrete memoryless channels an upperbound on the bit error probability of a convolutional code can be obtained. The derivation of the bound is based on an union bound argument on the transfer function T(D,N) of the code which describes the weight distribution, or weight spectrum, of the incorrect codewords and the number of bit errors on these paths [8], [3], [4]. The entire transfer function of the code is rarely known but an upper bound can still be calculated using only the first few terms of two series expansion related to the transfer function T(D,N), that is

$$T(D,N)\Big|_{N=1} = \sum_{j=d_{free}}^{\infty} a_j D^j \qquad (13)$$

and

$$\frac{dT(D,N)}{dN}\Big|_{N=1} = \sum_{j=d_{free}}^{\infty} c_j D^j \qquad (14)$$

In these expressions d_{free} is the free distance of the code, a_j is the number of incorrect paths or adversaries of Hamming weight j, $j \geq dfree$, that diverge from the correct path and remerge with it sometime later. As for c_j it is simply the total number of bit errors in all the adversaries having a given Hamming weight j.

Using the weight spectrum an upper bound on the bit error probability bound P_B of a code of rate R=b/V is given by

$$P_B \leq \frac{1}{b} \sum_{j=d_{free}}^{\infty} c_j P_j \qquad (15)$$

The evaluation of this bound depends on the actual expression of the pairwise error probability P_j, which in turn depends on the type of modulation and channel parameters used [8], [3], [4].

For coherent PSK modulation and unquantized additive white gaussian noise channels, the pairwise error probability between two codewords that differ over j symbols is bounded by

$$P_j \leq \exp(-jREb/N_0) \qquad (16)$$

where Eb/N_0 is the energy per-bit to noise ratio.

A good evaluation of the bound on P_b requires knowledge of the transfer functions (13) or (14). However for the vast majority of codes only the first few terms of either functions are known, and very often only the leading coefficients a_{dfree} and c_{dfree} are available. But for channels with large Eb/N_0 values such as those usually used with high rate codes, the bound on P_B is dominated by the first term c_{dfree}

Naturally bound (15) is also applicable for punctured codes. Therefore in deriving the bit error performance of punctured codes the free distances and at least the first few terms of the weight spectra of these codes must be obtained. These informations are provided in this paper for the short constraint length codes given by others [19], [20] and for the long constraint length codes discovered by the authors.

4. SEQUENTIAL DECODING FOR PUNCTURED CODES

In sequential decoding, the decoding of the received message is performed one branch at a time without searching the entire tree. The exploration of the most likely fraction of the tree is performed along the path having the current largest likelihood function or Fano metric [9]. Starting from the root note of the tree, the path selected to be extended one step further in the tree is the path whose metric is the largest among all previously examined paths. For practical reasons the tree is limited to a depth of a few hundreds to a few thousands branches, and the path that first reaches the end of the tree is accepted as the decoded path.

For binary convolutional codes of rate R=b/V, 2^b branches each with V code symbols emerge from each node of the encoding tree. Since decoding is an exploration of tree paths then clearly, a straightforward application of the decoding algorithm to codes of high rates b/V may become very quickly unacceptable. Several methods using discarding thresholds to eliminate unlikely paths have been proposed to alleviate this difficulty [26], [27].

However even with discarding thresholds, the use of sequential decoding for high rate codes is somewhat limited by the lack of suitable long constraint length codes [28], [29], [30]. By considering punctured codes instead, both difficulties of computational complexity and finding good high rate codes can be circumvented [7], [25].

Just like for Viterbi decoding, the punctured approach to high rate codes can, as easily, be applied to long memory codes and sequential decoding. Again decoding is performed on the tree of the original low rate code rather than on the high rate code [21], [22], [7], [24], and in principle any sequential decoding algorithm could be used. However only the Zigangirov-Jelinek or stack algorithm will be considered here [31], [32]. Since it is very simple and readily amenable to variants and generalisations [11], [33], this algorithm is a good candidate for punctured codes decoding.

In the stack algorithm the decoder consists of a stack or list of the already searched paths, ordered in decreasing order of their metric values. The path having the current largest accumulated metric is at the top of the stack, and will be searched further, i.e., extended one level further along all the branch extensions emerging from its end node. The algorithm consists of the following 3 steps:

1. Compute the metrics of all successors of the top node and enter them in their proper place in the stack.

2. Remove from the stack the top node that was just extended

3. Find the new top node. If it is the final node, stop. Otherwise go to 1.

For the decoding of high-rate punctured codes the stack algorithm requires only minimal modifications. Given the perforation pattern the decoder proceeds one branch at a time just like for the decoding of the original low rate $1/V_0$ code. For a punctured rate b/V code, the algorithm is repeated through (b-1) elementary intermediate steps involving the original low rate code. At each intermediate step the top node of the stack is extended into its two V_0-symbol branches from which specific symbols are discarded. Again this is accomplished by inhibiting the metric evaluation of these punctured symbols, that is, by assigning them the same metric increment (usually zero).

The advantages of the procedure is that the likelihood that the decoder is following the correct path is evaluated on a bit-by-bit basis rather than on the basis of blocks of b bits like in a brute force decoding. Hence the decision to retreat in the tree and follow another path may be taken earlier, at a possible reduction of the computational variability and required stack storage. In addition, since each elementary extension involves only two branches, for a rate b/V code the forward motion of the decoder involves 2b entries in the stack rather than the 2^b entries required of straithforward decoding.

This is illustrated in Fig. 4 where the paths explored by the stack algorithm for both an usual and punctured rate 2/3 code are compared. Since each decoding cycle of the algorithm involves $2^2=4$ stack entries for the usual code, but only 2 for the punctured code, then clearly the slightless metric dip of the correct path is far more costly (in terms of computations and stack storage) for the usual than for the punctured code. For example, Fig. 4 shows that in reaching the same decoded path, a substantially larger number of nodes must be explored with the usual code than with the punctured code.

An additional advantage of the punctured approach is that the flag procedure usually used to save stack entries and speed-up the decoding of rate 1/V codes can be readily implemented here at no additional cost. The use of rejection thresholds to discard very unlikely branch extensions can even be added up at hardly any cost [26], [27]. Therefore the stack algorithm can be very easily adapted for the decoding of punctured codes, and the method can lead to substantial simplifications and stack storage savings over the direct high coding rate approach.

Finally just like variable-rate Viterbi decoding, variable-rate sequential decoding can be easily implemented. If all the punctured codes of interest are derived from the same low rate $1/V_0$ original code, then changing the coding rate involves only changing the perforation pattern.

PERFORMANCE CHARACTERISTICS OF SEQUENTIAL DECODING

The performance analysis of sequential decoding involves both the error performance and computational effort.

It is well known that regardless of the algorithm, the computational effort of sequential decoding is on the average very small, but may be also highly variable. The number of computations C to decode one branch has an asymptotically Pareto distribution given by:

$$P(C \geq N) \approx AN^{-\alpha}, \quad N >> 1 \tag{17}$$

where a computation is defined as the execution of step 1 of the algorithm. In (17) A is a constant and α, the Pareto exponent is given approximately by:

$$\alpha \simeq \frac{R_{comp}}{R} \tag{18}$$

where R_{comp} depends on the channel only [9], [10].

Provided the coding rate R is smaller than R_{comp}, typically the average number of computations per decoded bit, C_{AV}, is very small, much smaller than the constant number of computations 2^M required for Viterbi decoding of codes of memory M. Although a computation is more complex for sequential than for Viterbi decoding, in sequential decoding C_{AV} is practically independent of M, which may thus be chosen to be quite large, much larger than with Viterbi decoding. However, whenever the coding rate R exceeds R_{comp}, then C_{AV} becomes theoretically unbounded. Decoding becomes erratic and the decoder may never succeed in reaching the end of the tree. R_{comp} is called the computational cut-off rate of sequential decoding, and the operating point of sequential decoders is often given by the ratio R/R_{comp}. This ratio is usually chosen to be close to but strictly smaller than 1 [9], [10], [1], [3], [4].

Even though sequential decoding is a suboptimal procedure, just like for Viterbi decoding its error probability decreases exponentially with the memory M of the code and can therefore be made arbitrarily small, provided again that $R/R_{comp}<1$. Over an ensemble of convolutional codes of rate R and memory length M, the error probability can be upper bounded by

$$P_B < B\ 2^{-M\ Rcomp/R}, \qquad R/Rcomp < 1 \qquad (19)$$

where B is a constant [9].

In Fig. 5 the Eb/N_0 values that are required for operation at a given R/R_{comp} are plotted as a function of the rate R, $1/2 \leq R < 1$, for hard quantized white gaussian noise channels. For 3 bit soft quantized channels these E_b/N_0 values are improved by approximately 2dB. However, from Fig. 5 it is apparent that the 2dB gain provided by soft quantization yields an improvement in R/R_{comp} that decreases very rapidly as the coding rate increases.

Long memory codes present no problem in sequential decoding. The achievement of low error probabilities, fast decoding speeds and low overflow probabilities require the selection of long memory codes having a large free distance and a good distance profile. Extensive lists of such codes exist for low rate 1/V codes [34], [3], [4]. However with few exceptions usual memory codes suitable for sequential decoding and with rates higher than 3/4 are not known [30], [29]. By taking the punctured approach, the search for good high rate codes becomes much simplified. The search problem for punctured codes is examined next

5. SEARCH FOR GOOD PUNCTURED CODES

Since punctured coding was originally devised for Viterbi decoding, the criterion of goodness for these codes was the free distance, and the maximal free distance punctured codes that first appeared in the literature were all short constraint length codes [18], [19], [20]. For sequential decoding good long constraint length punctured codes should have both a large free distance and a good distance profile.

In searching for good punctured codes of rate b/V and memory M, one is confronted with the problem of finding both an original code of rate $R=1/V_0$, $V_0 \leq V$, and its accompanying perforation pattern. Not unlike the search for usual convolutional codes, the search for punctured codes is often based on intuition and trial and error rather than on a strict mathematical construction [25].

An approach that yielded good results is based on the intuition that "good codes generate good codes". Consequently one could choose a known good large memory code of rate $R=1/V_0$, (e.g. R=1/2, 1/3, 1/4...) and exhaustively try out all possible perforation patterns to generate good punctured codes of rates R=b/V. Naturally if families of variable-rate codes are desired then all the perforation patterns must be applied to the same low rate code. Furthermore if the codes are to be rate compatible, then the perforation patterns must be selected accordingly.

Although one could select the punctured code on the basis of its free distance only, a finer method consists of determining the weight spectrum of the punctured code according to (13) and (14) and then plotting the bit error probability bound (15). The code yielding the best error performance may be thus selected as the best punctured code, provided it is not catastrophic. Therefore a check for the catastroph conditions must also be applied to each suitable candidate.

Clearly then, starting from a known optimal low rate code a successful search for good punctured codes hinges on the ability of determining the weight spectrum corresponding to each possible perforation pattern. Although seemingly simple, determinating the weight spectrum of punctured codes turned out to be a very difficult, and sometimes, a formidable task. This is because even if the spectrum of the low rate original code were available, the spectrum of the punctured code cannot be derived from it. One has to go back exploring the tree or trellis of the low rate original code and apply to each path of interest the perforation pattern.

For the well known short memory codes the procedure is at best a rediscovery of their weight spectra, whereas for long memory codes where often only the free distance is known, it is a novel determination of their spectra. The problem is further compounded by the fact that since puncturing a path reduces its Hamming weight, then in obtaining a given number of spectral terms a larger number of paths must be explored for a punctured than for a usual code.

In our search we have used recently developed algorithms for the spectral determination of convolutional codes [35] [36]. These algorithms which use a stack, explore astutely the tree structure of the codes and offer substqantial advantages over all other published methods for determining the codes spectra. Each algorithm has been designe to be especially efficient within a given range of constraint lengths, allowing substantial extentions of known spectra of the best convolutional codes.

SHORT CONSTRAINT LENGTH PUNCTURED CODES

A number of short constraint length (K<9) punctured codes of rates R=(V-1)/V varying from 2/3 to 16/17 have been proposed by Cain et al [18], Yasuda et al [19], [20], and more recently by Hagenauer [23] for rate compatible codes. In particular in [19] all the memory M=6 punctured codes of rates varying from 2/3 to 16/17 have been derived from the same original constraint length 7, rate 1/2, best known convolutional code due to Odenwalder [37]. A more complete list of rate 2/3 to 13/14 punctured codes has been derived from the best known rate 1/2 codes with constraint length K varying from 3 to 9 and compiled by Yasuda et al [20]. In this list, for each code the perforation matrix is provided but the weight spectrum is limited to the first term only, that is, the term corresponding to d_{free}.

Using the given perforation patterns we have extended the results in [20] by determining the first seven spectral terms of all the codes having constraint length $3 \leq K \leq 9$ and coding rates 2/3, 3/4, 4/5, 5/6, 6/7 and 7/8. these results are given in Table 1 to Table 6 respectively. Each Table lists the generators of the original low rate code, the perforation matrix, the free distance of the resulting punctured code and the coefficients a_n and $c_{n)}$ of the series expansions (13) and (14) of the corresponding weight spectra. These spectra determinations have been conducted on SUN/3 and APPOLO microcomputers and turned out to be quite time consuming, with a tripling of the required CPU time for each additional spectral coefficient. Beyond seven spectral coefficients, the required computer time becomes prohibitive, reaching several hundred hours with these microcomputers.

The bit error probability upper bound P_B over the binary symmetric channel has been evaluated for all the codes listed in Tables 1 to 6 and are shown in Fig. 6 to 12. These bounds have been calculated according to (15) using all the weight spectra terms listed in the Tables. As for the pairwise error probability P_j, the exact expression based on the binominal expansions

$$P_j = \begin{cases} \sum_{i=\frac{j+1}{2}}^{j} \binom{j}{i} p^i (1-p)^{j-i} , & j \text{ odd} \\ \sum_{i=\frac{j}{2}+1}^{j} \binom{j}{i} p^i (1-p)^{j-i} + \frac{1}{2} \binom{j}{j/2} [p(1-p)]^{j/2} , & j \text{ even} \end{cases} \qquad (20)$$

has been used, where p is the channel transition probability [3]. Double precision arithmetic has been used throughout.

For all the codes above the error performance improves as the coding rate decreases, indicating well chosen perforation patterns. A notable exception may be seen on Fig. 7 for the constraint length 4 code where the error performance is slightly better at rate 4/5 than it is at rate 3/4. This anomaly may be explained by an examination of the spectra of these two codes. As shown in Tables 2 and 3, the free distance of the rate 3/4 and 4/5 codes are $d_f=4$ and $d_f=3$ respectively. However the number of bit errors on the various spectral terms becomes far larger on the rate 3/4 code than on the rate 4/5 code. This anomaly illustrates the fact that selecting a code according to the free distance only is good in general but may sometimes be insufficient. Knowledge of further terms of the spectrum will always provide more insight on the code performance. Accordingly, seeking to improve on that K=4, R=3/4 code provided in Table 2, new K=4, R=3/4 codes have been discovered. These new punctured codes deliver a better error performance than the one given by Yasuda et al [19] even though their free distance are $d_f=3$ whereas Yasuda's code has $d_f=4$.

Figure 13 compares the bit error probability bounds for the constraint length 7 punctured codes of rates 2/3, 3/4, 7/8 and 15/16, of Yasuda et al [19]. For comparison purposes Fig.13

includes the theoretical bound for the the original rate 1/2 code as well as the performance curve of the uncoded coherent PSK modulation. It shows that the performance degradation from the original rate 1/2 code is rather gentle as the coding rate increases from 1/2 to 15/16. At $P_B=10^{-5}$ the coding gains for the punctured rate 2/3 and 3/4 codes are 4.9 dB and 4.6 dB respectively. These results indicate that these codes are indeed very good, even though their free distances, which are equal to 6 and 5 respectively, are slightly smaller than the free distances of the best known usual rate 2/3 and 3/4 codes which are equal to 7 and 6 respectively.

The error performance of these codes has been verified using an actual punctured Viterbi codec [19], and independently, using computer simulation [21]. Both evaluations have been performed using 8-level soft decision Viterbi decoding with truncation path lengths equal to 50, 56, 96 and 240 bits for the coding rates 2/3, 3/4, 7/8 and 15/16 respectively. Both hardware and software evaluations have yielded identical error performances which, interestingly, match closely the theoretical upper bound.

Fig.14 shows the bit error performance bound curves of both the constraint length 7 punctured and Maximal Free Distance codes of rates 2/3 and 3/4. These bounds have been computed using only the term at d_{free} for the MFD codes, and using both terms at d_{free} and $d_{(free+1)}$ for the punctured codes. Based on these terms only the two MFD codes appear to be only very slightly better than punctured codes. Therefore it may be concluded that although not optimal, the error performances of the rates 2/3 and 3/4 punctured codes of constraint length 7 closely match those of the MFD codes of the same rates and constraint lengths. The same general conclusions may be made for the other punctured codes with different constraint lengths.

At rate 15/16 Fig. 13 shows that the coding gain reaches a substantial 3dB at $P_B=10^{-6}$. The fact that such a coding gain can be achieved with only a 7% redundancy and a Viterbi decoder that is hardly more complex than for a rate 1/2 code makes the punctured coding technique very attractive for short constraint length codes. For longer codes and larger coding gains, Viterbi decoding becomes impractical and sequential decoding should be considered instead. Newly discovered long constraint length punctured codes and their performance with sequential decoding are presented next.

LONG CONSTRAINT LENGTH PUNCTURED CODES

Following the same approach as for the short constraint length codes, one could choose a known optimal long constraint code of rate 1/2 and exhaustively try out all possible perforation patterns to generate all punctured codes of high rate R=b/V. The selection of the punctured code is again based on its bit error performance which is calculated from the series expansion of its transfer function. Here one of the difficulties is that for the original low rate and long memory codes of interest, only very partial knowledge of their weight spectra is available [39]. In fact beyond constraint length K=16, very often only the free distances of these codes are available in the literature [34].

Extending previous work in this area [21], [22], [7], a computer search form the best rates 2/3 and 3/4 punctured codes of constraint length extending from 10 to 23 that are derived from the best known low rate 1/2 codes has been conducted. For each code the first few terms of the weight spectrum have been obtained for each possible distinct perforation pattern. The obviously bad codes are discarded and the codes to be investigated further are those that provide the largest free distance and the smallest number of bit errors in their weight spectra. However the final selection of the best punctured codes is based on the evaluation of the upper bound on the bit error probability. Naturally the codes obtained with this approach are suitable for variable rate decoding with sequential decoding.

The search has been initially limited to these two rates only since a fine comparison of the punctured codes with the best known non systematic high rate codes of the same constraint length is limited to the rate 2/3 and 3/4 codes. This is due to the fact that with very few exceptions optimal long codes for sequential decoding are known for rates 2/3 and 3/4 only.

Table 7 and Table 8 list the characteristics of the best punctured codes of rate 2/3 and 3/4 respectively, with constraint lengths varying from 10 to 23, derived from the best non systematic rate 1/2 codes for sequential decoding [34]. In both Tables, for each constraint length the generators of the original code and its perforation matrix are given, together with the free distance of the resulting punctured code. As with short constraint length codes the first few terms a_n and c_n, $n=d_{free}$, d_{free+1}, d_{free+2},... of the series expansions of the weight spectra are also given for each punctured code. In deriving these spectral coefficients up to 24 terms of the original code weight spectra have been used.

In the search for the best punctured codes the perforation patterns were chosen as to yield both a maximal free distance and a good distance profile. Although all perforation patterns were exhaustively examined, the search was somewhat reduced by exploiting equivalence of the perforation patterns under cyclical shifts of their rows [25]. Among all the codes that were found, Tables 7 and 8 list only those having the smallest number of bit errors c_{dfree} at the largest free distance d_{free}, and obviously all, the codes listed are non catastrophic.

Fig. 15 plots the free distances of the original rate 1/2 codes and the punctured rate 2/3 and 3/4 derived from them as a function of the constraint length. As expected the free distance of the punctured codes of a given rate is non decreasing with the constraint length, and at a given constraint length the free distance decreases with increasing coding rates.

When the punctured codes of rate b/V are determined from the best original low rate 1/V code, an upper bound on the free distance of the punctured code can be derived [25]. This derivation which is based on an analysis of the effect of the different perforation patterns on the spectrum of the original code yields the bound

$$d_{free(p)} \leq (1/b)\, d_{free(0)} \qquad (21)$$

where $d_{free(p)}$ and $d_{free(0)}$ are the free distances of the punctured and original codes respectively. This bound which agrees with the results of the usual high rate codes indicates that the best punctured codes may have a free distance not necessarily lower than that of the best known maximal free distance codes of the same rate.

The upper bounds on the bit error probability have been evaluated for all the punctured codes listed in Tables 7 and 8 and are shown in Fig. 16 to 19. Just like for the short constraint length codes the evaluation has been performed according to (20) over a binary symmetric channel. These bounds indicate a normal behaviour for all the punctured codes listed in Tables 7 and 8. The bit error performances all improve as the coding rate decreases and/or as the constraint length increases, with approximately 0.5 dB improvement for each unit increase of the constraint length.

The selection of the best punctured codes listed Tables 7 and 8 has been based on both the maximal free distance and the calculated bit error probability bound. However the choice of the best punctured code was often not a very clear cut one. It was often observed that different perforation patterns yielded only marginally different error performances. This is illustrated typically in Fig. 20 for the rate 3/4 constraint length 20 punctured codes obtained under different perforation patterns. It can be seen that the two perforation patterns $\begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}$ and $\begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$ yield codes with error performance bounds quite close to that of the selected code with pattern $\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$

In some cases the performance curves were undistinguishable and hence several "best" punctured codes having the same rate and constraint length may be obtained. For example Table 7 lists two K=18 and two K=20 rate 2/3 "best" punctured codes, and Table 8 lists two choices for the K=11 and the K=20 rate 3/4 codes. However since these codes are for sequential decoding applications then clearly, the final selection must also be based on the distance profile and computational performance. Short of analysing the computational behaviour, when in doubt, the codes finally selected had the fastest growing column distance function.

In the search for punctured codes the above approach allows to find good but not optimal codes since the original low rate code is imposed at the ouset. However given an optimal usual high rate code of rate R=b/V and constraint length K, one could attempt to determine the low rate 1/V code which, after perforation will yield a punctured code that is equivalent to that optimal code. This approach which is the converse of the usual code searching method allows to find the punctured code equivalent to any known usual high rate code. Based on this approach and using the notion of orthogonal perforation patterns a systematic construction technique has been developed by the authors [25]. Using this technique punctured codes equivalent to the best known nonsystematic rate 2/3 codes with constraint lengths up to K=24 have been found. Likewise, punctured codes equivalent to the best rate 3/4 codes with constraint lengths up to K=10 have been tabulated [25]. Longer R=3/4 punctured codes have not been searched since beyond K=10 optimal usual R=3/4 codes are not available in the literature.

Furthermore, using the same approach the punctured codes equivalent to all the best known very long memory systematic codes of rates 2/3, 3/4, 4/5, 5/6 and 7/8 discovered by Hagenauer [30] have been determined [25]. Finally it must be pointed out that the punctured codes generated by this second approach are not suitable for variable rate applications since each punctured code has its own distinct low rate original code.

6. COMPUTER SIMULATION RESULTS

The stack algorithm for sequential decoding of punctured rates 2/3 and 3/4 codes has been simulated on a computer. In these simulations the results of interest concern both the computational effort and bit error performance.

For R/R_{comp} values ranging between 0.85 and 0.99, meaningful error events could be observed with short constraint length codes only. In these cases the resulting error probabilities were always found to satisfy the upper bound computed from the first terms of the code spectrum. As the constraint length of the codes increase no errors were collected within a reasonable simulation time (approximately 200,000 bits). Therefore, as usually expected with sequential decoding of long constraint length codes, the simulation results will concern mainly the computational effort.

Figure 21 shows a typical distribution of computation obtained for the constraint length 24 rate 2/3 punctured code equivalent to the best known usual code of the same constraint length and rate discovered by Johannesson and Paaske [29]. For comparison purposes the computational distribution for the usual code is also plotted. Both punctured and normal codes were simulated over identical additive white gaussian noise binary symmetric channels with Eb/N_0=5.76 dB corresponding to R/R_{comp}=0.94. In either case no decoding errors were observed.

The distributions of the computational efforts appear to be very nearly identical for both the punctured and usual codes. However an examination of the decoding parameters demonstrate the advantages of using the punctured approach for the decoding of high rate codes. The average number of stack entries S_{AV} is equal to 1057 for the usual code whereas it is only 656 for the punctured code. Counting a computation as the extension of the top node of the stack into all its successors, then clearly the decoding of punctured codes requires on the average more computations than the decoding of usual high rate codes, but these computations are far simpler. For our example, as indicated in Fig. 21 the average number of computations C_{AV} is equal 1.32 and 1.12 for the punctured and usual codes respectively. A fairer comparison of the average computational effort using punctured and usual codes may be based on the overall simulation times, since using both the punctured and normal codes the decoding algorithms are identical with the exception of the branch extensions and metric inhibition of the punctured symbols. For example, for rate 2/3 codes, the decoding time of the punctured codes was observed to be approximately 16% of that required for the usual codes whereas for rate 7/8 codes this proportion of decoding time drops to 5 percent. Based on extensive simulation results involving a large number of codes having different coding rates over a wide range of Eb/N_0 values, it was observed that these decoding advantages of the punctured codes over the normal codes increase as both the coding rates and Eb/N_0 increase [38].

7. CONCLUSIONS

In this paper we have presented the encoding as well as Viterbi and sequential encoding decoding of high rate punctured convolutional codes. These codes are derived from well known optimal low rate convolutional, and depending on the choice of the perforation patterns may yield easy implementations of variable rate and rate compatible coding. Using computer search we have extended results on previously discovered short constraint length punctured codes by providing up to seven terms of their weight spectra.

We have shown that sequential decoding can be easily adapted for the decoding of punctured codes and have provided new rates 2/3 and 3/4 long constraint length codes with $10 \leq K \leq 23$. Upper bounds on the bit error probabilities of all the codes examined have been computed. The substantial advantages of using high rate punctured codes over the usual high rate open the way for powerful yet practical implementations of varable rate codecs, extending from very low to very high coding rates.

ACKNOWLEDGEMENTS

We wish to thank Miss Chantal Paquin, a graduate student at Ecole Polytechnique de Montréal, for her help and devotion in running the computer programs for the search of punctured codes.

REFERENCES

[1] I.M. JACOBS, "Practical Applications of Coding", IEEE Trans. on Inf. Theory, IT-20, pp. 305-310, May 1974.

[2] E.R., BERLEKAMP, "The Technology of Error-Correcting Codes", Proc. IEEE, vol. 68, pp. 564-593, May 1980.

[3] V.K. BHARGAVA, D. HACCOUN, R. MATYAS, P. NUSPL, "Digital Communications by Satellite", J. Wiley and Sons, N.Y., 1981.

[4] S. LIN and D. J. COSTELLO Jr., "Error Control Coding", Prentice-Hall, Englewood Cliffs, N.J. 1983.

[5] W. W. WU, "Elements of Digital Satellite Communications, Volume II", Computer Science Press, Rockville, Md, 1985.

[6] A. MICHELSON and A. LEVESQUE, "Error-Control Techniques for Digital Communication", J. Wiley and Sons, N.Y., 1985.

[7] W. WU, D. HACCOUN, R. PEILE, Y. HIRATA and "Coding for Satellite Communication", IEEE Journal on Selected Areas in Communications, vol. SAC-5, No. 4, pp. 724-748, May 1987.

[8] A.J. VITERBI, "Convolutional Codes and Their Performance in Communications Systems", IEEE Trans. on Comm. Tech., Vol. COM-19, Oct. 1971.

[9] R.M.FANO,"A Heuristic Discussion of Probabilistic Decoding", IEEE Trans. on Inform. Theory, Vol. IT-9, April 1962.

[10] I. M. JACOBS and E. R. BERLEKAMP, "A Lower Bound to the Distribution of Computation for Sequential Decoding", IEEE Trans. on Inf. Theory, Vol. IT-13, pp. 167-174, April 1967.

[11] D. HACCOUN and M.J. FERGUSON, "Generalized Stack Algorithms for Decoding Convolutional Codes", IEEE Trans. on Inf. th., vol. IT-21, pp. 638-651, Nov. 1975.

[12] P.R. CHEVILLAT and D.J. COSTELLO Jr., "A Multiple Stack Algorithm for Erasurefree Decoding of Convolutional Codes", IEEE Trans. Commun., COM-25, pp. 1460-1470, Dec. 1977.

[13] J.A. HELLER and I.M. JACOBS, "Viterbi Decoding for Satellite and Space Communications", IEEE Trans. on Com. Techn., COM-19, Oct. 1971.

[14] A.S. ACAMPORA and R. GILMORE, "Analog Viterbi Decoding for High Speed Digital Satellite Channels", NTC 77 Conference Record, Los Angeles, CA, 34.6.1-34.6.5, Dec. 1977.

[15] R.T. CLARK and R.D. McCALLISTER, "Development of an LSI Maximum-Lilelihood Convolutional Decoder for Advanced Forward Error Correction Capability on the NASA 30/20 GHz Program", Proceedings AIAA 9th Communications Satellite System Conference, San Diego, CA, pp. 142-144, March 1982.

[16] R.M. ORNDOFF et al., "Viterbi Decoder VLSI Integrated Circuit for Bit Error Correction", Proceedings National Telecommunications Conference, New Orleans, LA, pp. E1.7.1-E1.7.4, Dec. 1981.

[17] J.S. SNYDER and T. MURATANI, "Forward Error Correction for Satellite TDMA in the Intelsat V Era", AIAA 8th Communications Satellite Systems Conference, Orlando, FL., pp. 674-683, April 1980.

[18] J.B. CAIN, G.C. CLARK and J. GEIST, "Punctured Convolutional Codes of Rate (n-1)/n and Simplified Maximum Likelihood Decoding", IEEE Trans. Inform. Theory, vol. IT-25, pp. 97-100, Jan. 1979.

[19] Y. YASUDA, Y. HIRATA, K. NAKAMURA, S. OTANI, "Development of a Variable-Rate Viterbi Decoder and its Performance Characteristics", 6th International Conference on Digital Satellite Communications, Phoenix, Sept. 1983.

[20] Y. YASUDA, K. KASHIKI, Y. HIRATA, "High-Rate Punctured Convolutional Codes for Soft Decision Viterbi Decoding", IEEE Trans. Commun., vol. COM-32, pp. 315-319, March 1984.

[21] G. BEGIN and D. HACCOUN, "Sequential Decoding of Punctured Convolutional Codes", Proceedings of the 13th Biennial Symposium on Communications, Kingston, Ont., Canada, June 1986, pp. A.3.5-A3.8.

[22] G. BEGIN and D. HACCOUN,"Decoding of Punctured Convolutional Codes by the Stack Algorithm", Abstracts of Papers, 1986 IEEE International Symposium on Information Theory, Ann Arbor, Michigan, Oct. 1986, p. 159.

[23] J. HAGENAUER "Rate Compatible Punctured Convolutional Codes and their Applications", to appear in IEEE Trans. on Com., Dec. 1987.

[24] D. HACCOUN, G. BEGIN, "Codage et décodage séquentiel de codes convolutionnels perforés", Proceedings 11th GRETSI Conf., Nice, France, June 1987, pp. 221-224.

[25] G. BEGIN, D. HACCOUN, "High Rate Punctured Convolutional Codes: Structure Properties and Construction Technique", submitted for publication in IEEE Trans. on Com., May 1987.

[26] D. HACCOUN and C. NAIYUN, "Variants of the Stack Algorithm for the Decoding of High-Rate codes by Sequential Decoding", in Proc. Int. Satellite Commun. Conf., Ottawa, Canada, June 1983, pp. 21.4.1-21.4.4.

[27] D. HACCOUN, "Décodage séquentiel des codes convolutionals de taux élevés", Traitement du Signal, Paris, France, Vol. 5 No. 1, Jan. 1988.

[28] E. PAASKE, "Short Binary Convolutional Codes with Maximal Free Distance for rates 2/3 and 3/4", IEEE Trans. on Inf. Theory, Vol. IT-20, Sept. 1974, pp. 683-686.

[29] R. JOHANNESSON and E. PAASKE, "Further Results on Binary Convolutional Codes with an Optimum Distance Profile", IEEE Trans. on Inf. Theory, IT-24, pp. 264-268, March 1978.

[30] J. HAGENAUER, "High Rate Convolutional Codes with Good Profiles", IEEE Trans. Inf. Theory, Vol. IT-23, pp. 615-618, Sept. 1977.

[31] F. JELINEK, "A Fast Sequential Decoding Algorithm Using a Stack", IBM Jour. Res. and Develop., Vol. 13, pp 675-685, Nov. 1980.

[32] K. ZIGANGIROV, "Some Sequential Decoding Procedures", Problemii Peredachi Informatsii, Vol. 2, pp. 13-15, 1966.

[33] D. HACCOUN, "Variabilité de calcul et débordements de décodeurs séquentiels à pile", Traitement du signal (Paris, France), Vol. 3, No. 3, pp. 127-143, Déc. 1986.

[34] R.JOHANNESSON, "Some Long Rate One-Half Binary Convolutional Codes with an Optimal Distance Profile", IEEE Trans. on Inf. Theory, IT-22, pp. 629-631, Sept. 1976.

[35] P. MONTREUIL, "Algorithmes de détermination de spectres des codes convolutionnels", M.Sc.A. Thesis, Dept. Electr. Eng. Ecole Polytechnique de Montréal, 1987.

[36] D. HACCOUN, P. MONTREUIL, "Weight Spectrum Determination of Convolutional Codes", to be submitted to IEEE Trans. on Communications.

[37] J.P. ODENWALDER, "Optimal Decoding of Convolutional Codes", Ph.D. dissertation, Dept. of Electr. Eng., U.C.L.A., Los Angeles, 1970.

[38] G. BEGIN and D. HACCOUN, "Simulation Results of Sequential Decoding of High Rate Punctured Convolutional Codes", to be submitted to IEEE Trans. on Communications.

[39] J. CONAN, "On the Distance Properties of Paaske's Class of Rate 2/3 and 3/4 Convolutional Codes", IEEE Trans. on Inform. Theory, Vol. IT-30, pp. 100-104, Jan. 1984.

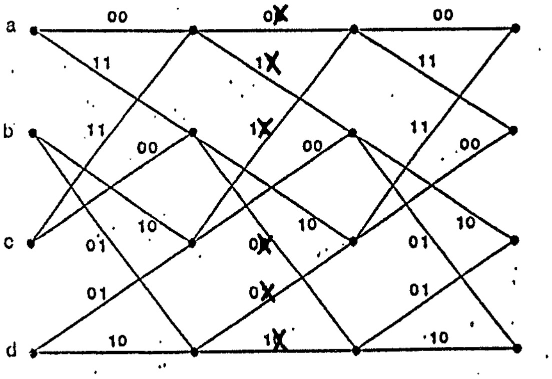

Figure 1

Treillis for K= 3, R=1/2
Convolutional Code

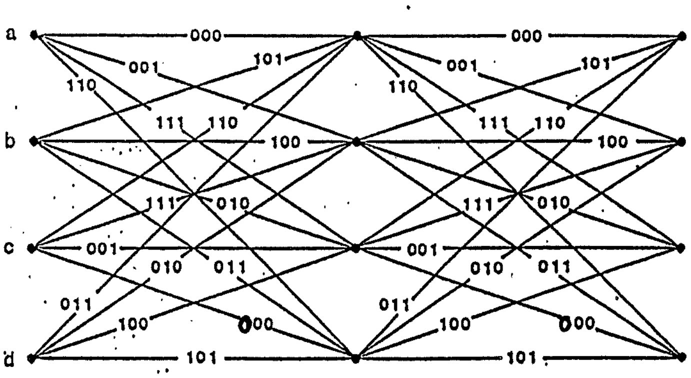

Figure 2

Treillis for Punctured
K= 3, R= 2/3 Code

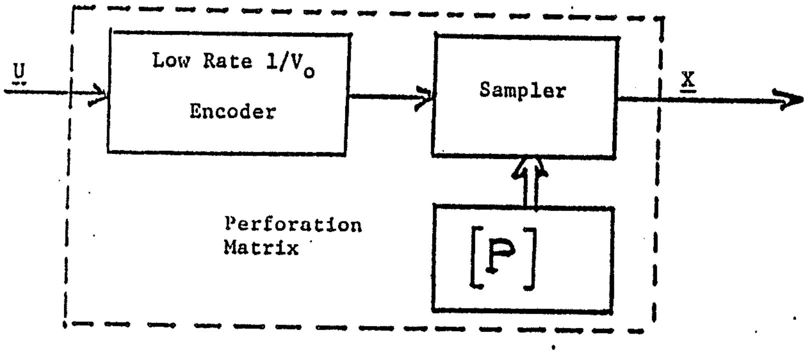

Figure 3

Encoder for Punctured
Convolutional Code

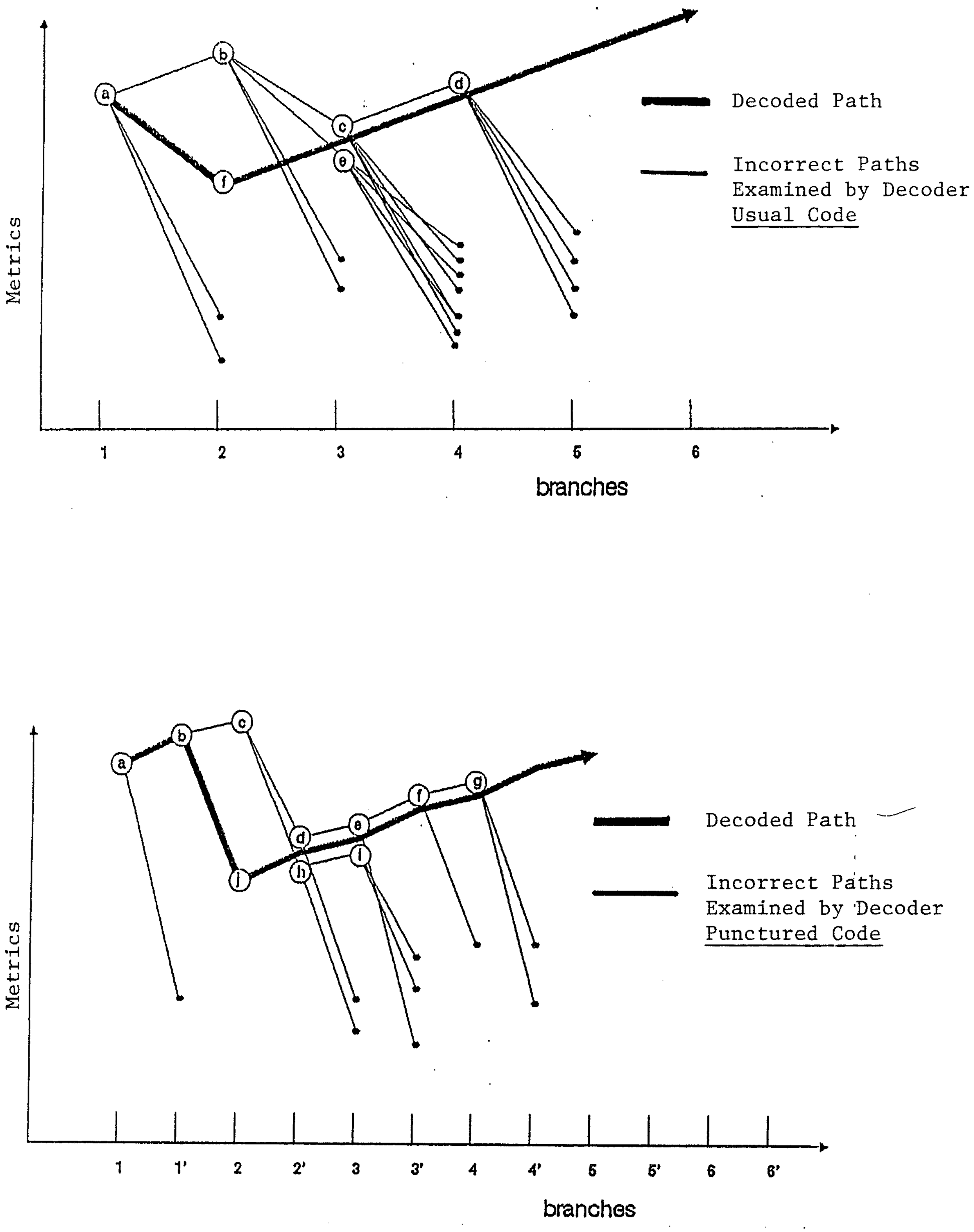

Figure 4 Paths Explored by the Stack Algorithm for an Usual and Punctured R= 2/3 Code

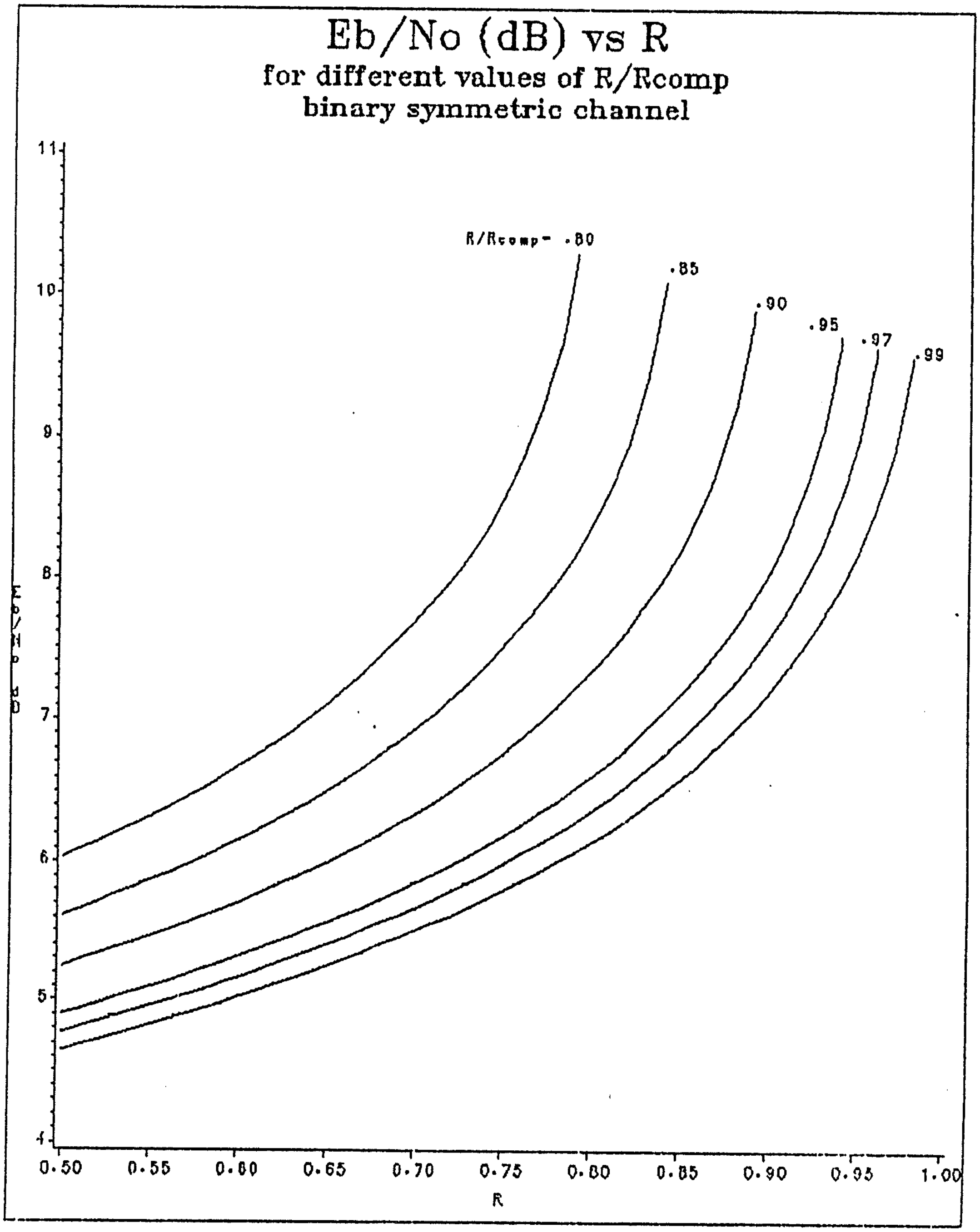

Figure 5 Eb/No Required for Given R/R_{comp} as a Function of the Coding Rate R

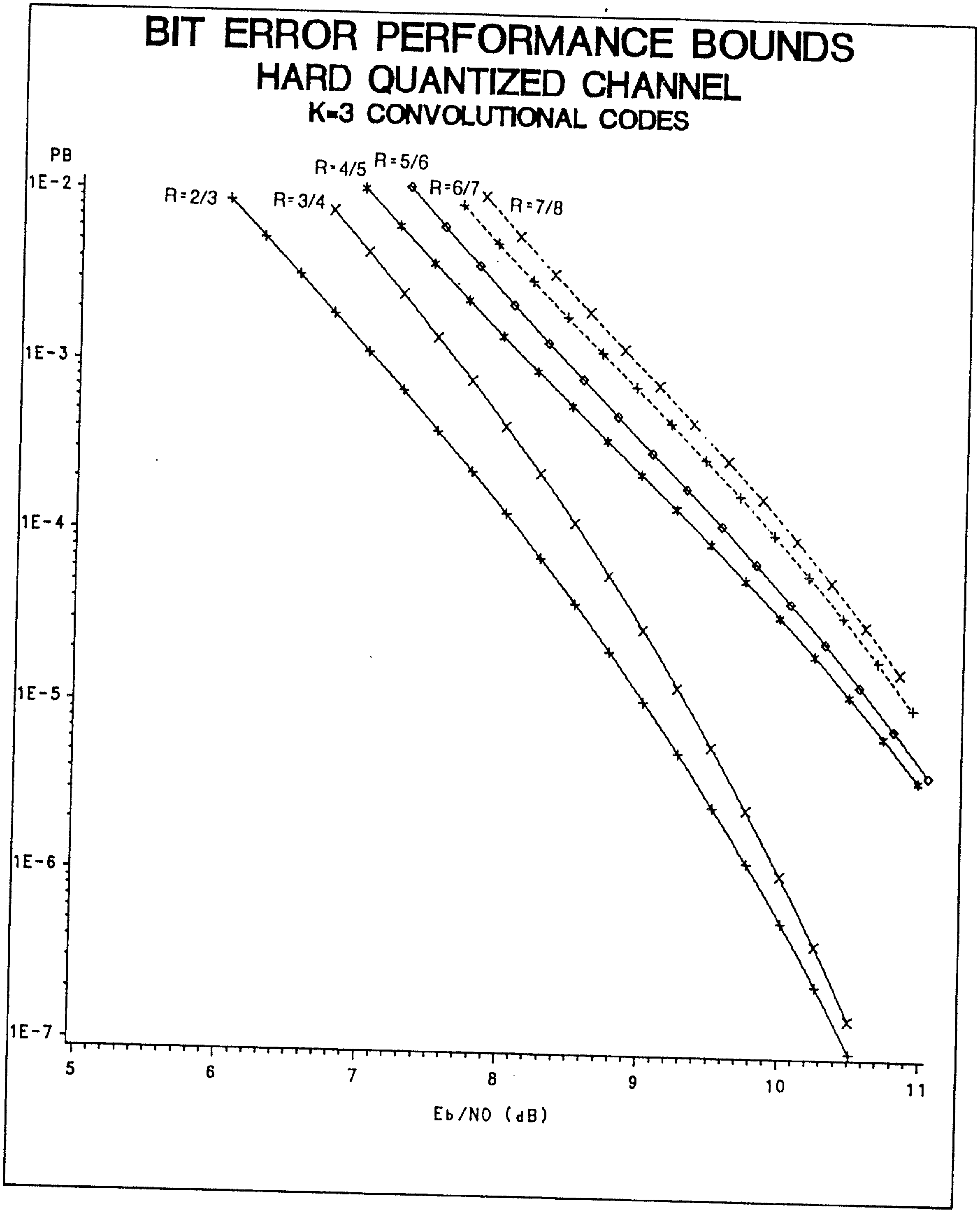

Figure 6 Upper Bounds on the Bit Error Probability for K = 3 Punctured Codes with 2/3 ⩽ R ⩽ 7/8 Derived from the K = 3, R = 1/2 Maximal Free Distance Code

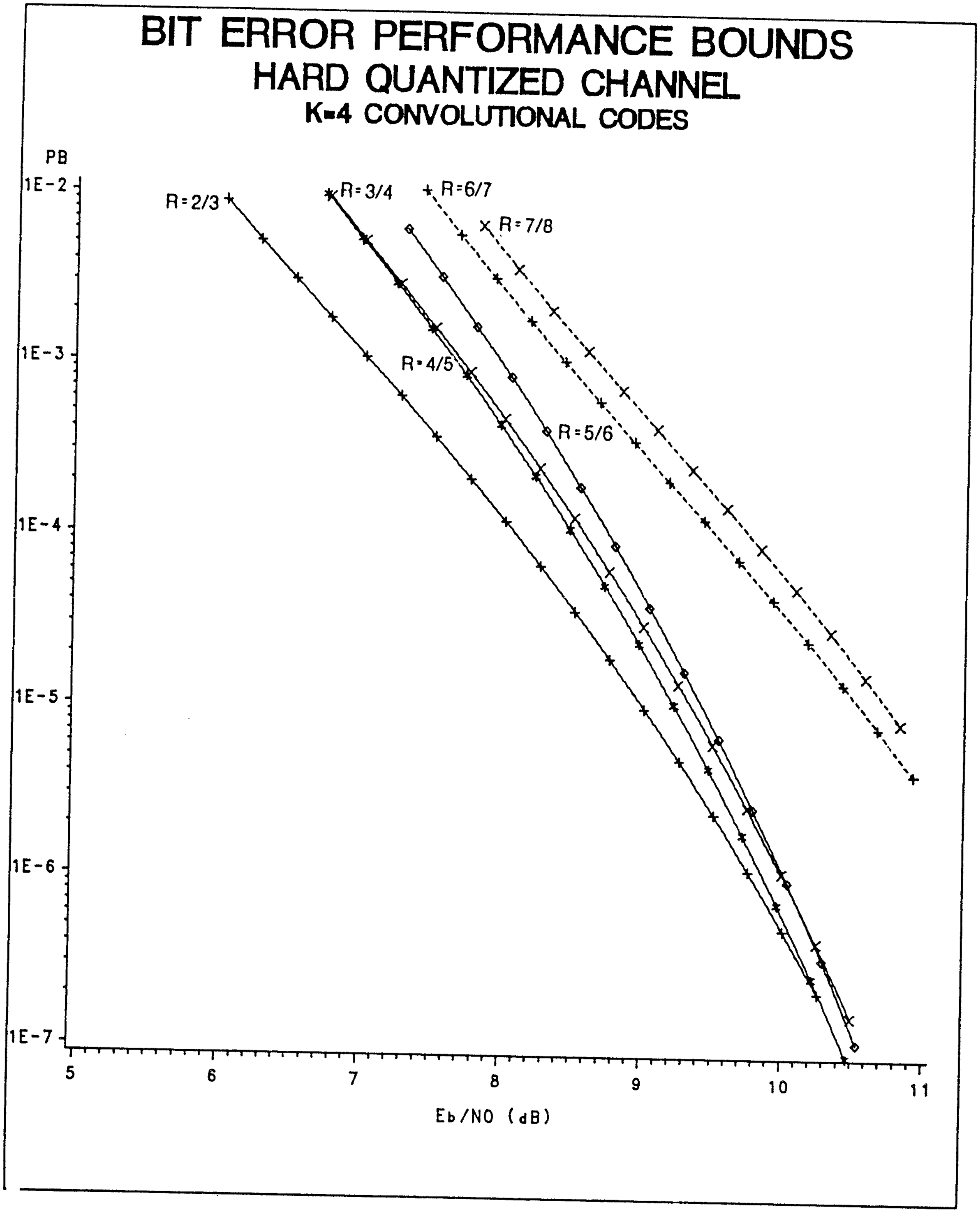

Figure 7 Upper Bounds on the Bit Error Probability for K = 4 Punctured Codes with 2/3 ⩽ R ⩽ 7/8 Derived from the K = 4, R = 1/2 Maximal Free Distance Code

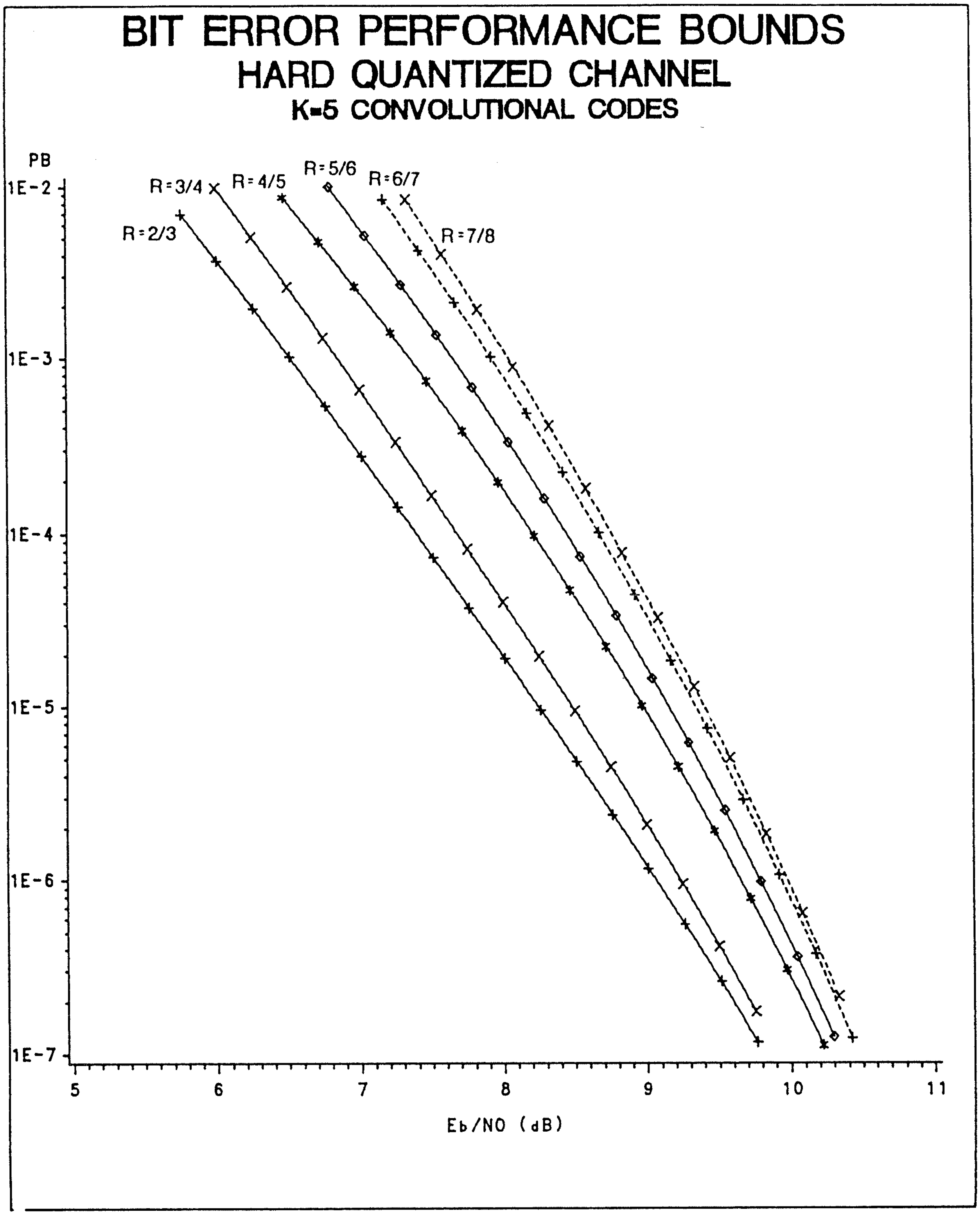

Figure 8 Upper Bounds on the Bit Error Probability for K = 5 Punctured Codes with 2/3 ⩽ R ⩽ 7/8 Derived from the K = 5, R = 1/2 Maximal Free Distance Code

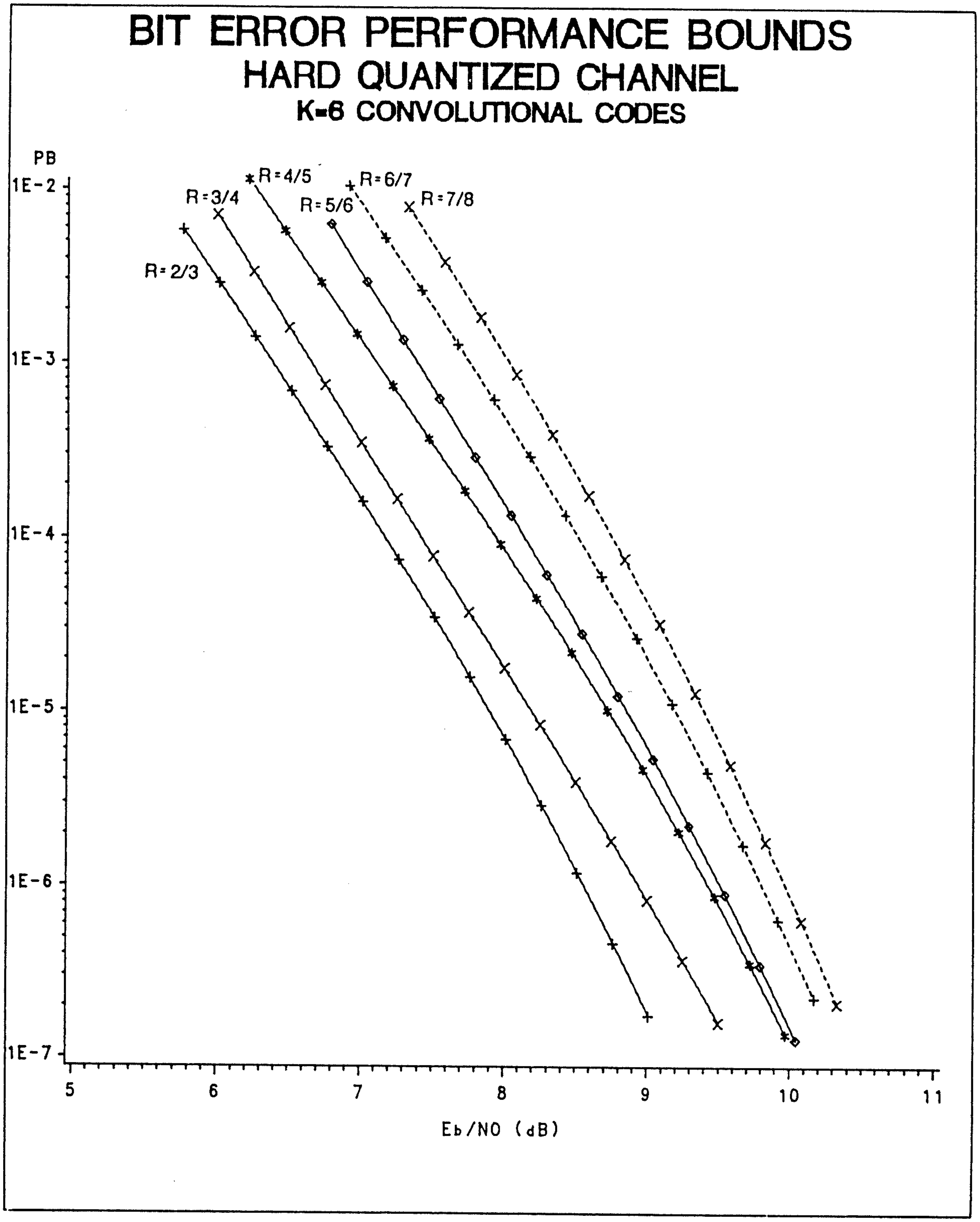

Figure 9 Upper Bounds on the Bit Error Probability for K = 6 Punctured Codes with 2/3 ⩽ R ⩽ 7/8 Derived from the K = 6, R = 1/2 Maximal Free Distance Code

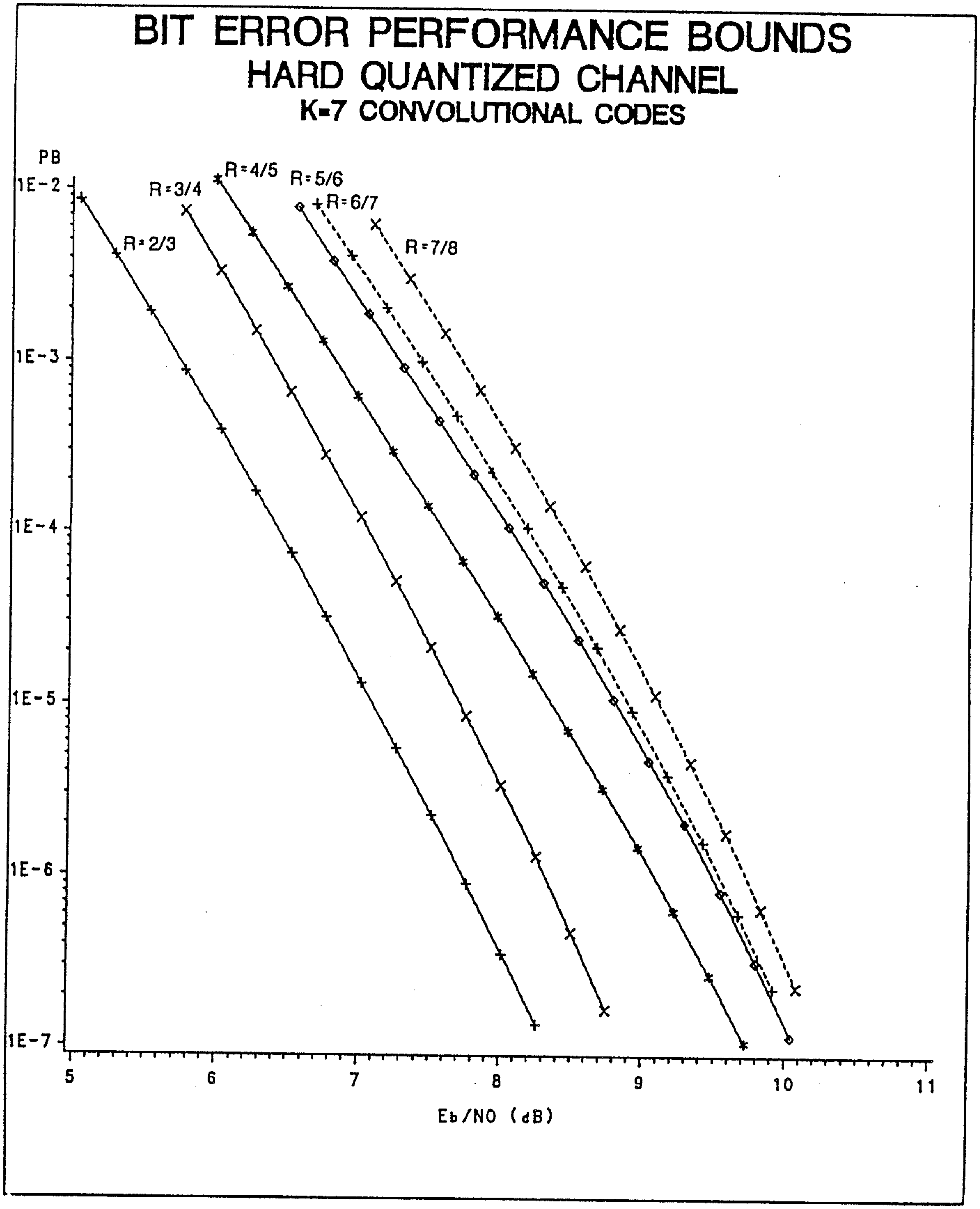

Figure 10 Upper Bounds on the Bit Error Probability for K = 7 Punctured Codes with 2/3 ⩽ R ⩽ 7/8 Derived from the K = 7, R = 1/2 Maximal Free Distance Code

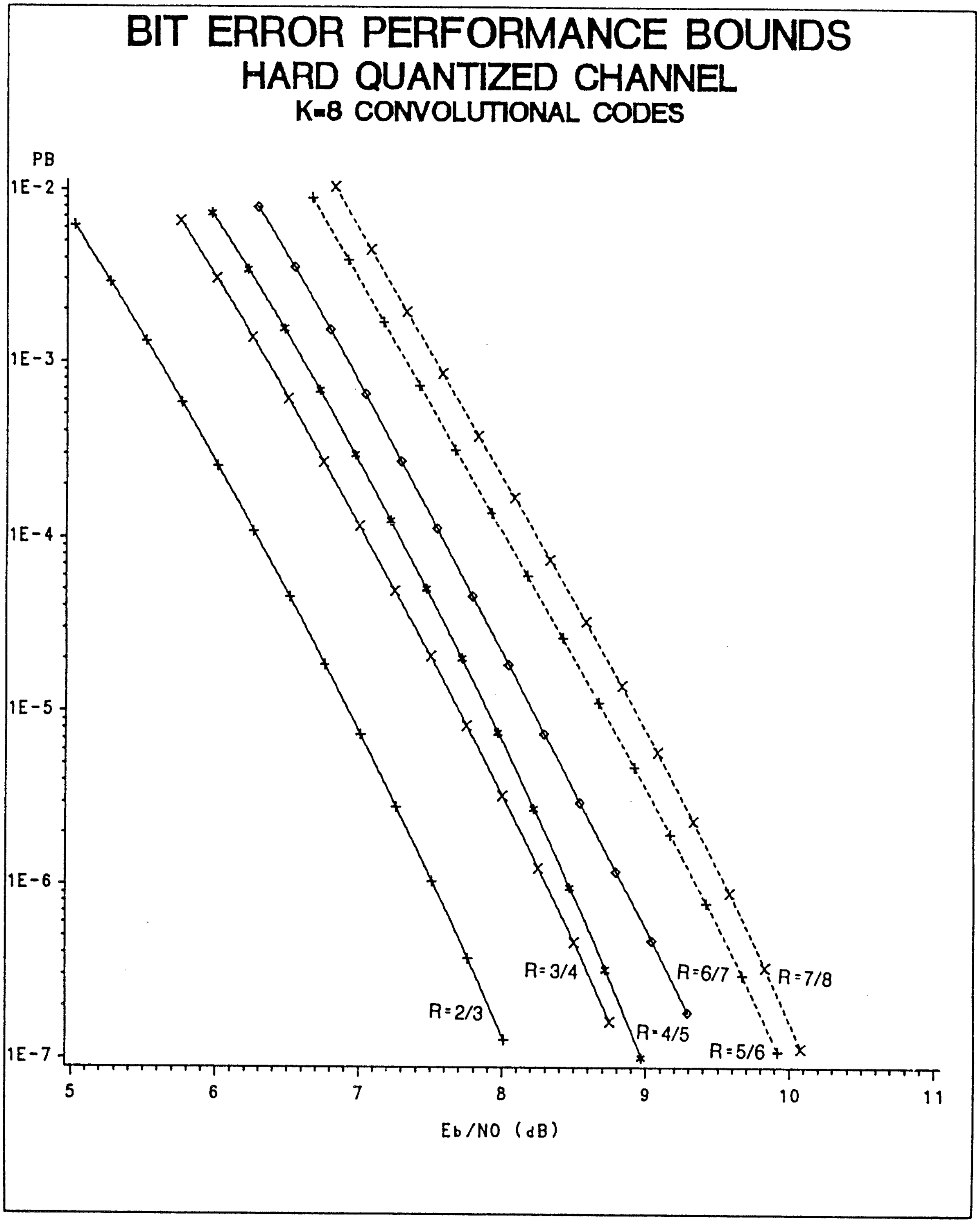

Figure 11 Upper Bounds on the Bit Error Probability for K = 8 Punctured Codes with 2/3 ⩽ R ⩽ 7/8 Derived from the K = 8, R = 1/2 Maximal Free Distance Code

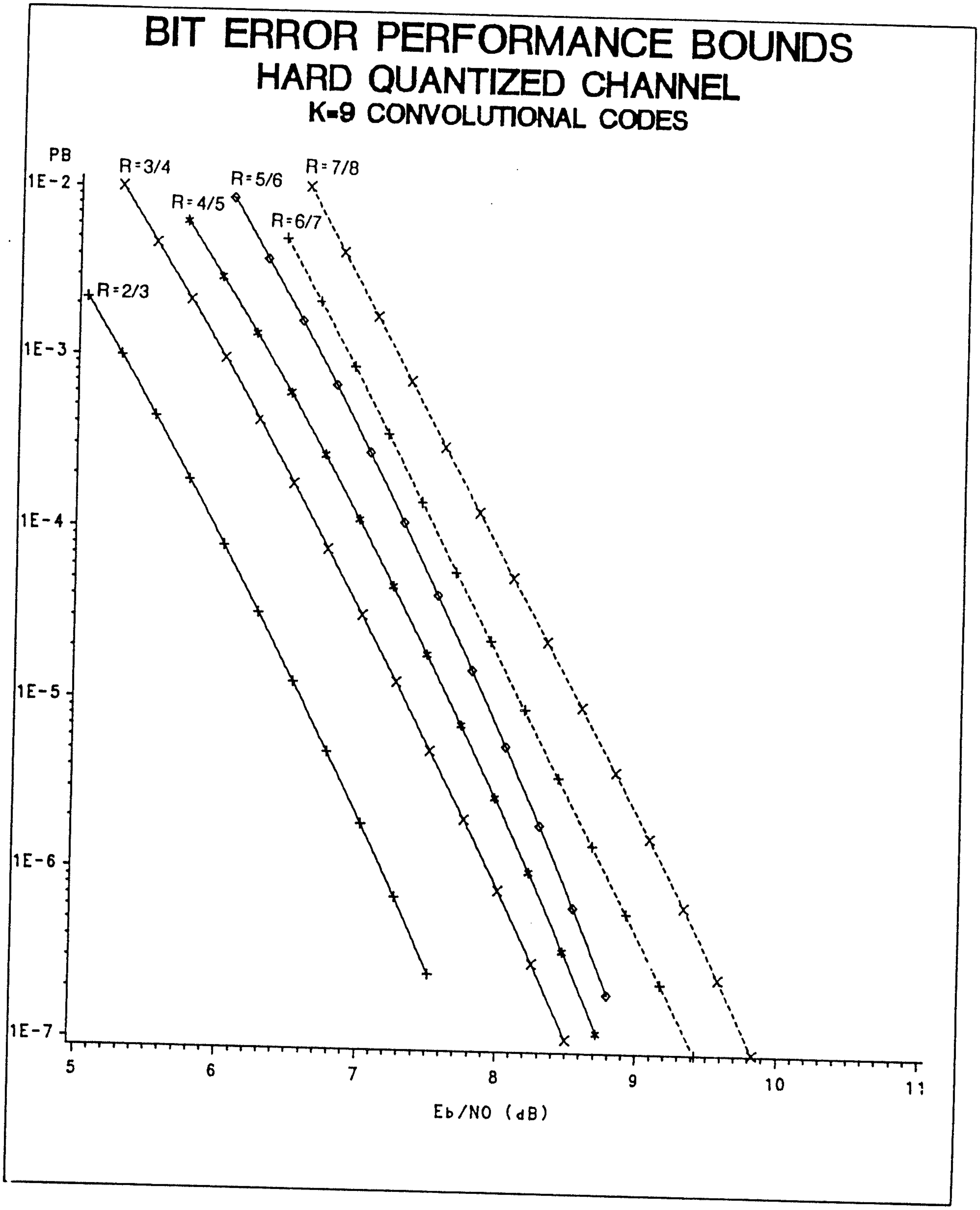

Figure 12 Upper Bou-ds on the Bit Error Probability for K = 9 Punctured Codes with 2/3 ⩽ R ⩽ 7/8 Derived from the K = 9, R = 1/2 Maximal Free Distance Code

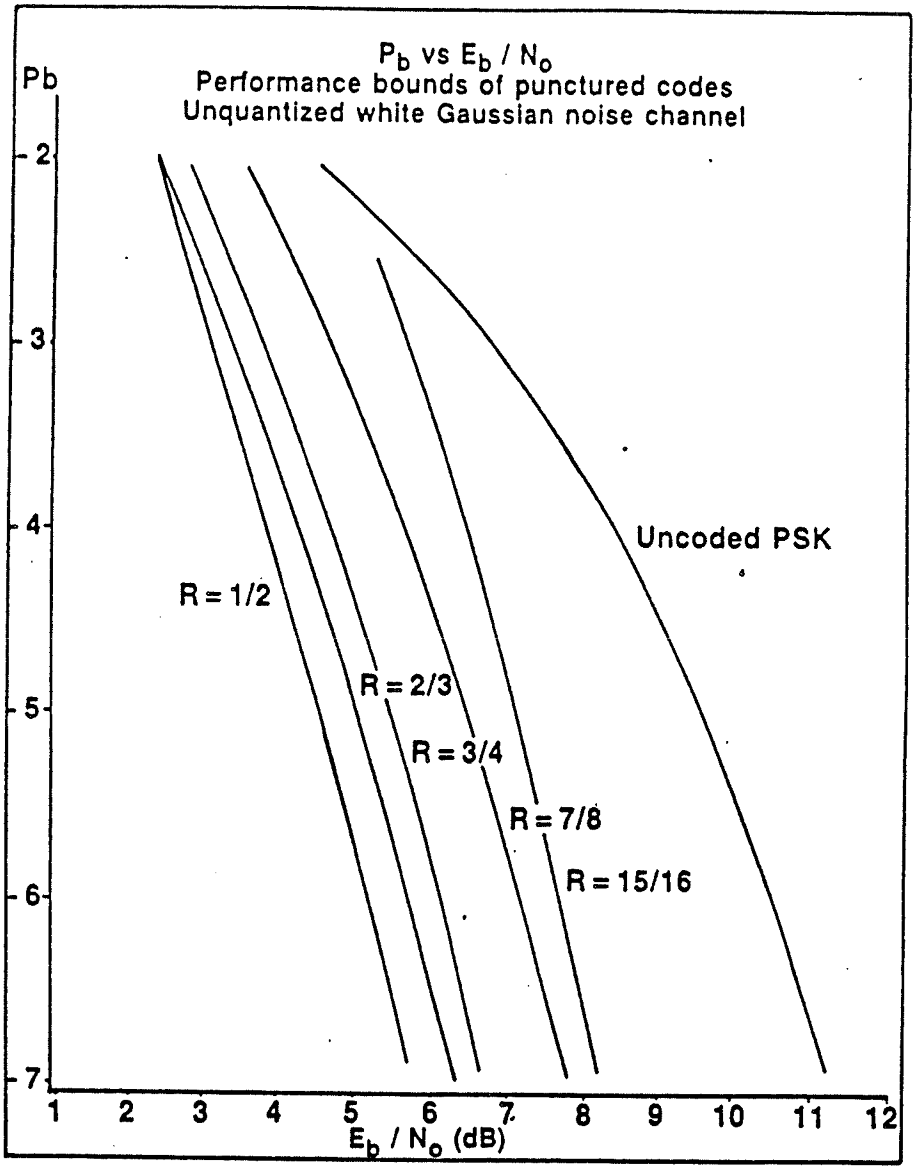

Figure 13 Bit Error Performance Bounds for the K = 7, R = 1/2 Original Code and the Punctured Rates 2/3, 3/4, 7/8 and 15/16 Derived fron it

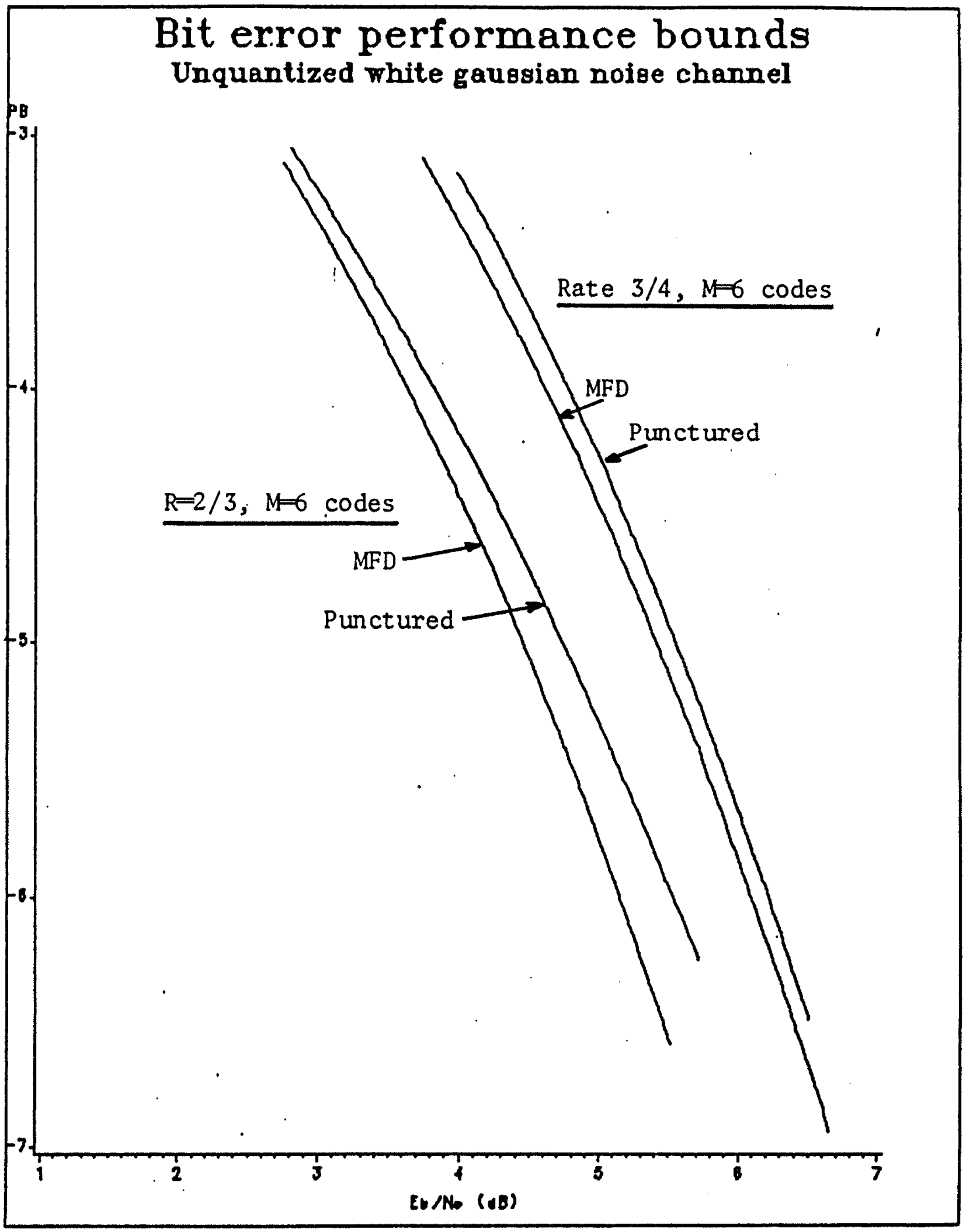

Figure 14 Bit error performance bounds for the maximal free distance (MFD) code and punctured code rates 2/3 and 3/4 with memory M = 6.

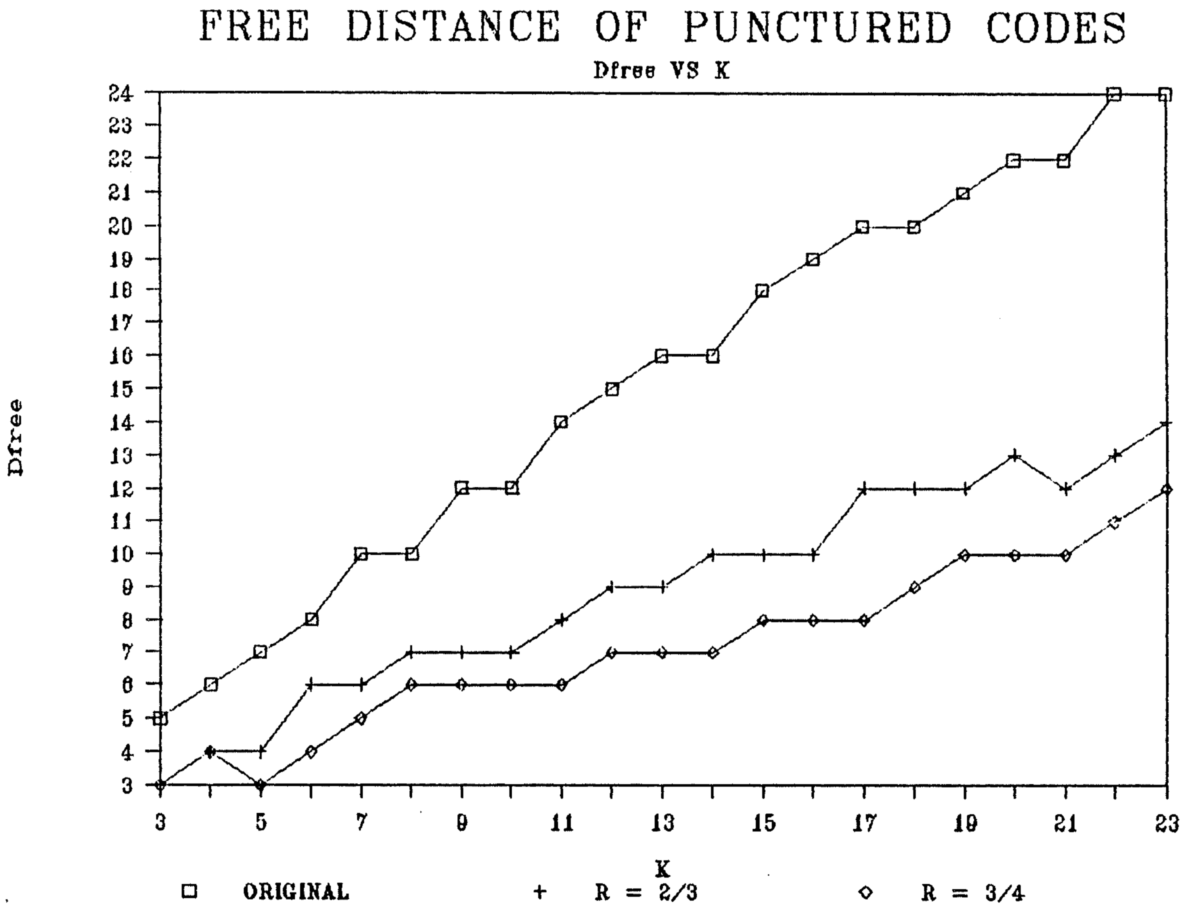

Figure 15 Free Distance of Original Rate 1/2 Codes and Punctured R = 2/3 and 3/4 Derived from them as a Function of K, 3 ⩽ K ⩽ 23

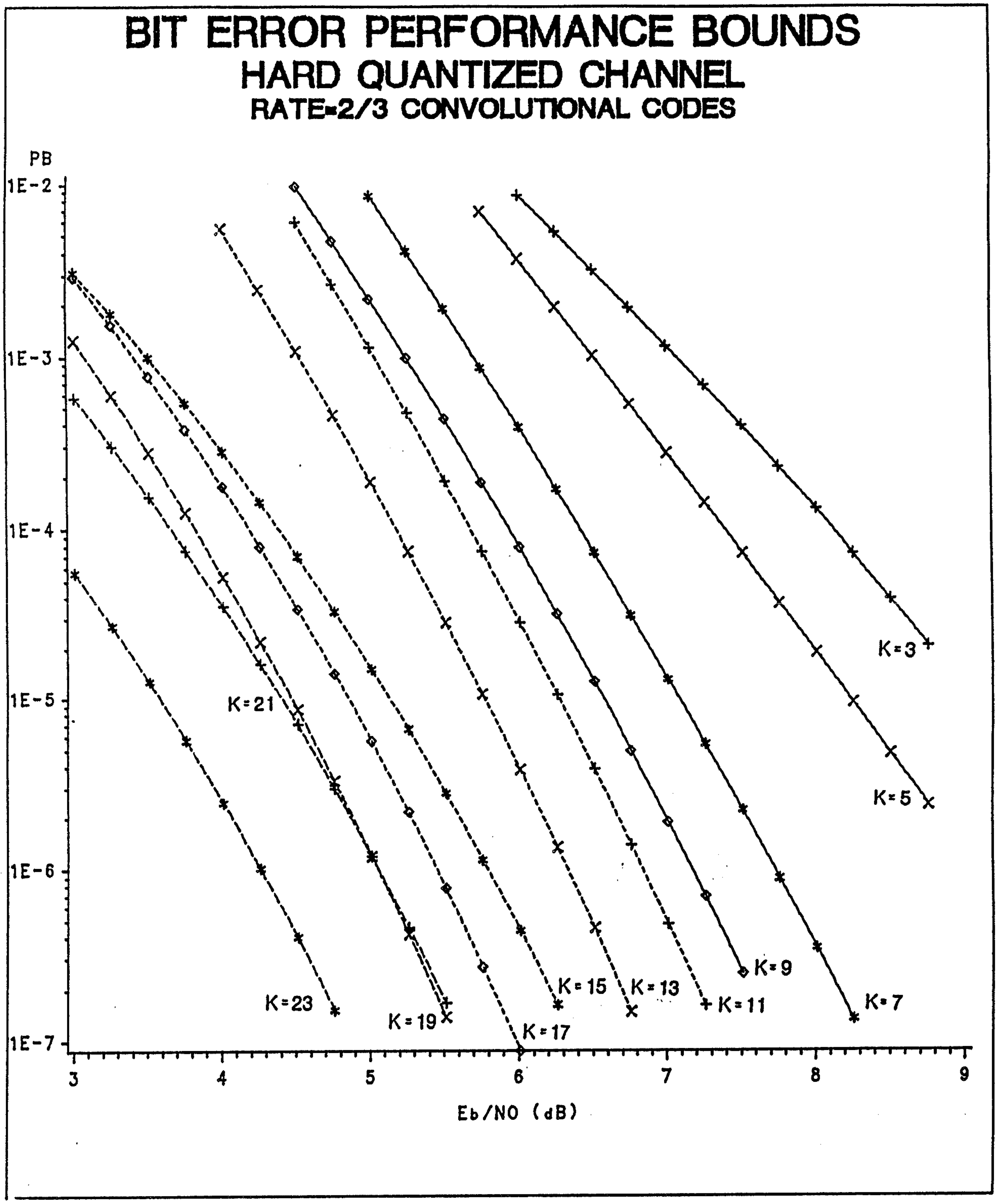

Figure 16 Upper Bounds on the Bit Error Probability over Hard Quantized Channel for R = 2/3 Punctured Codes with 23 ⩽ K ⩽ 3, K odd

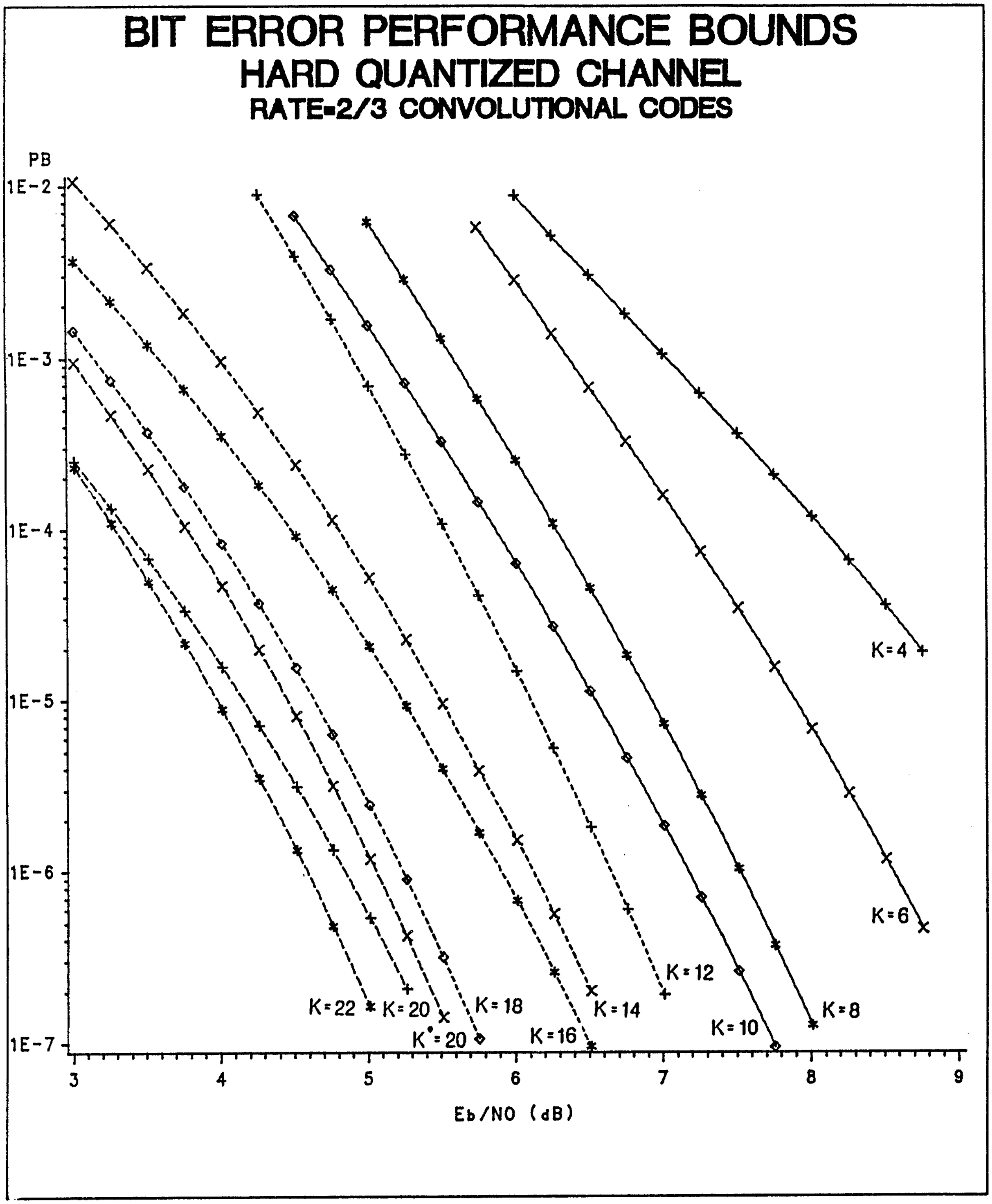

Figure 17 Upper Bounds on the Bit Error Probability over Hard Quantized Channel for R = 2/3 Punctured Codes with 22 ⩽ K ⩽ 4, K even

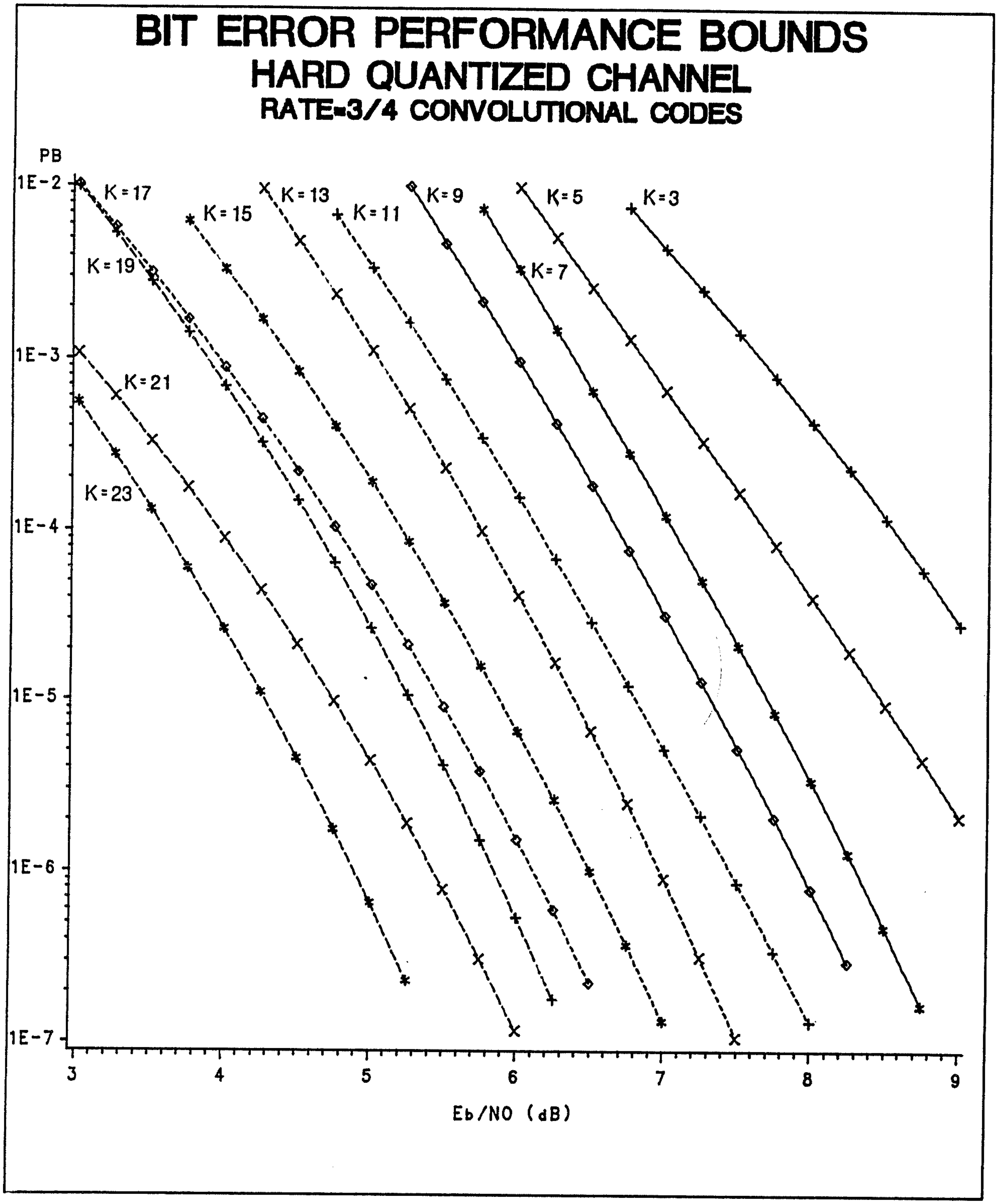

Figure 18 Upper Bounds on the Bit Error Probability over Hard Quantized Channel for R = 3/4 Punctured Codes with 23 ⩽ K ⩽ 3, K odd

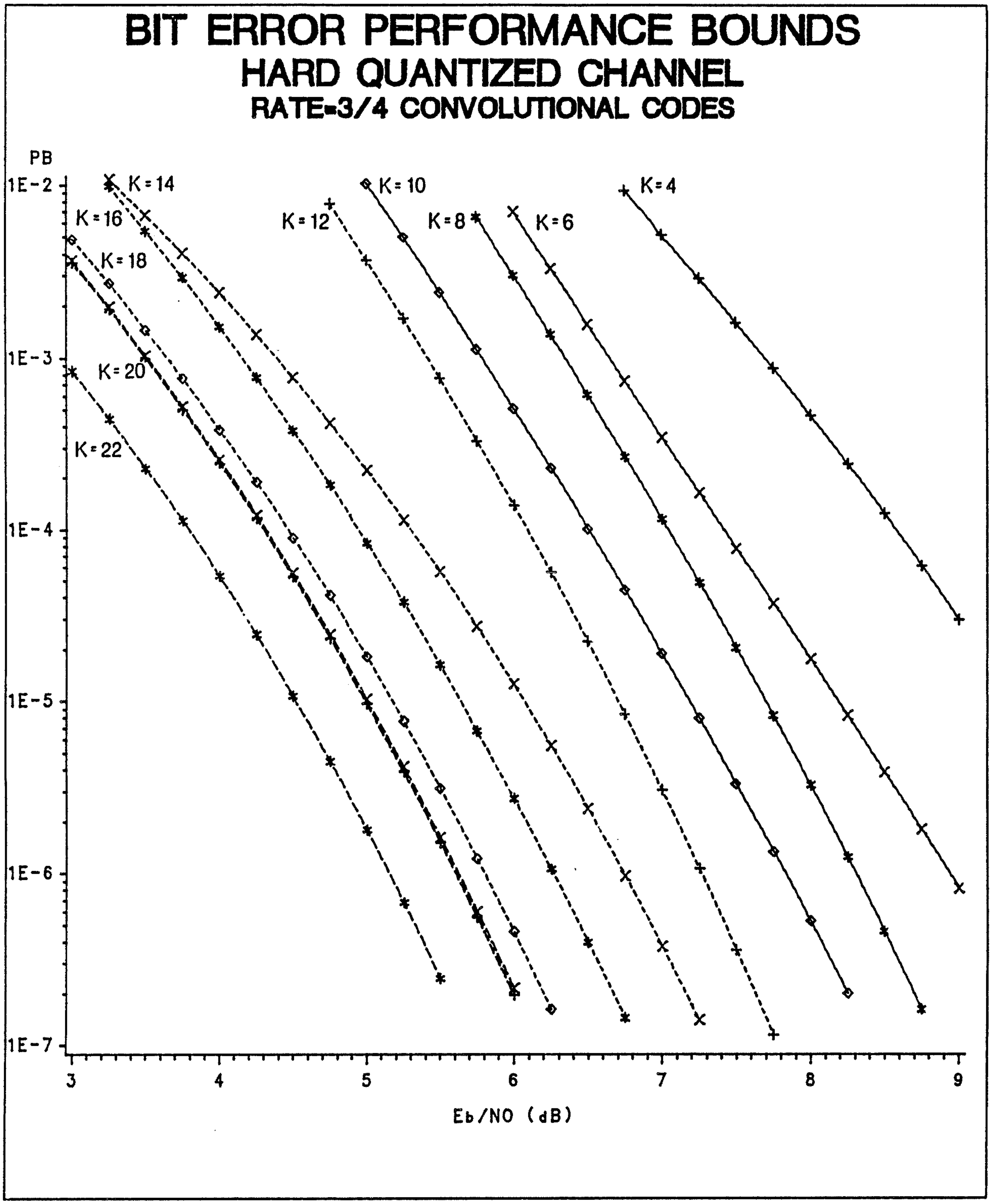

Figure 19 Upper Bounds on the Bit Error Probability over Hard Quantized Channel for R = 3/4 Punctured Codes with 22 ⩽ K ⩽ 4, K even

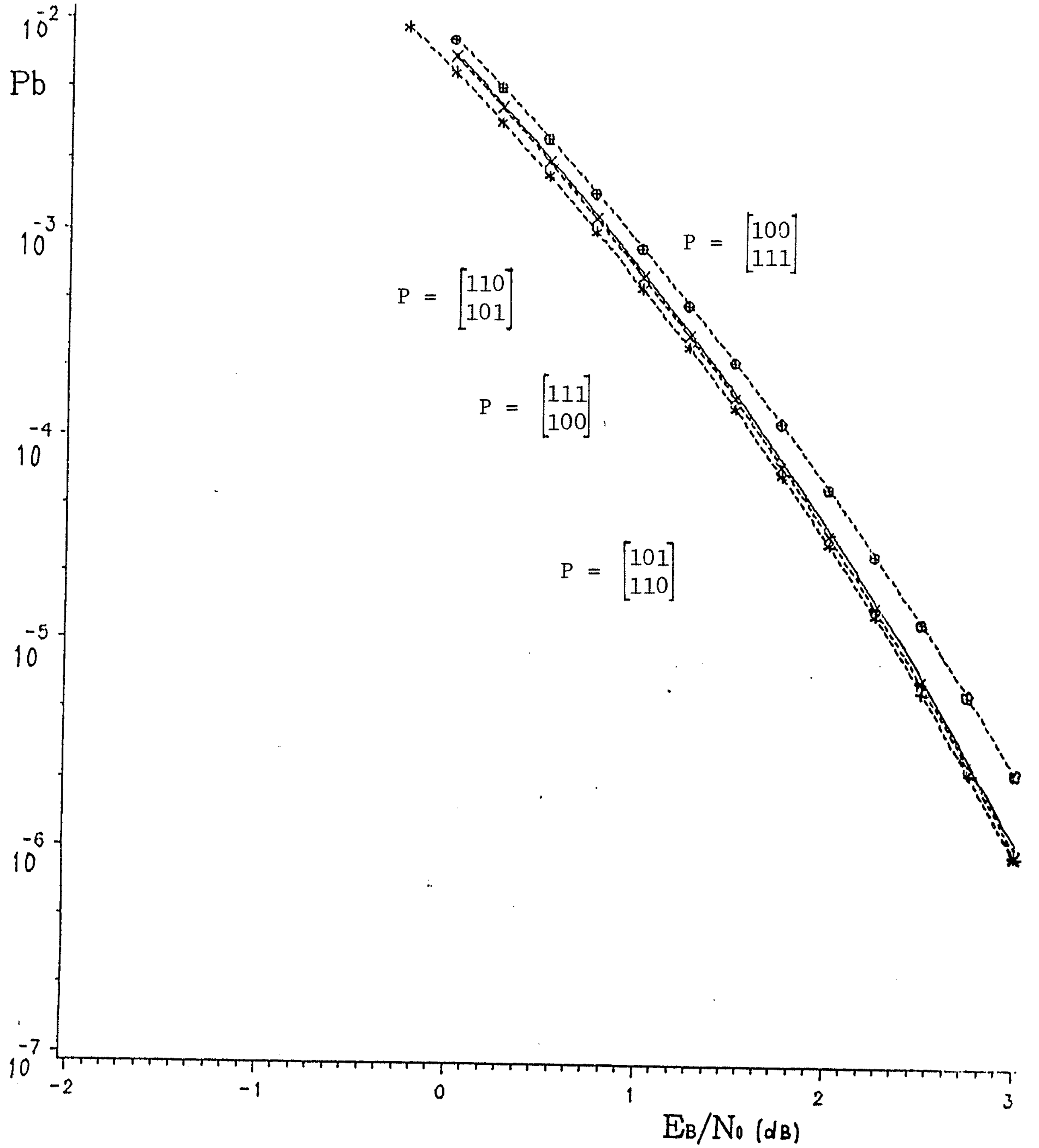

Figure 20 Upper Bounds on the Bit Error Probability over Unquantized Channel for the R = 3/4, K = 20 Punctured Code with Different Perforation Patterns

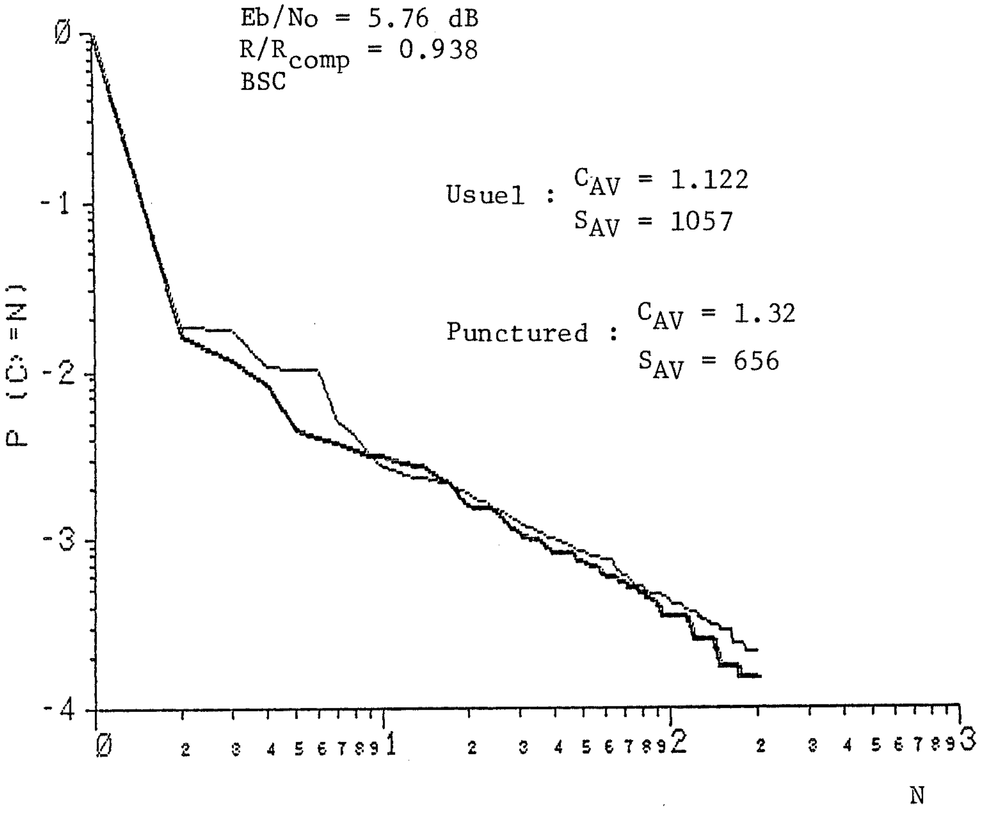

Figure 21 Empirical Distribution of the Number of Computations per Decoded Branch for Equivalent Usual and Punctured Codes having R = 2/3 and K = 24 over Hard Quantized Channel at Eb/No = 5.76 dB

original code				punctured code		
K	G_1	G_2	d_f	[P]	d_f	$(a_n,\ n = d_f,\ d_f+1,\ d_f+2,\ \ldots)$ $[c_n,\ n = d_f,\ d_f+1,\ d_f+2,\ \ldots]$
3	5	7	5	1 0 1 1	3	(1,4,14,40,116,339,991) [1,10,54,226,856,3072,10647]
4	15	17	6	1 1 1 0	4	(3,11,35,114,381,1276,4257) [10,43,200,826,3336,13032,49836]
5	23	35	7	1 1 1 0	4	(1,0,27,0,345,0,4528) [1,0,124,0,2721,0,50738]
6	53	75	8	1 0 1 1	6	(19,0,220,0,3089,0,42790) [96,0,1904,0,35936,0,638393]
7	133	171	10	1 1 1 0	6	(1,16,48,158,642,2435,9174) [3,70,285,1276,6160,27128,117019]
8	247	371	10	1 0 1 1	7	(9,35,104,372,1552,5905,22148) [47,237,835,3637,17770,76162,322120]
9	561	753	12	1 1 1 0	7	(3,9,50,190,641,2507,9745) [11,46,324,1594,6425,29069,127923]

Table 1 Weight Spectra of Yasuda's et al Punctured Codes with $R = 2/3$, $3 \leqslant K \leqslant 9$

original code				punctured code		
K	G_1	G_2	d_f	[P]	d_f	$(a_n, n = d_f, d_f+1, d_f+2, \ldots)$ $[c_n, n = d_f, d_f+1, d_f+2, \ldots]$
3	5	7	5	1 0 1 1 1 0	3	(6,23,80,290,1050,3804,13782) [15,104,540,2557,11441,49340,207335]
4	15	17	6	1 1 0 1 0 1	4	(29,0,532,0,10059,0,190112,0) [124,0,4504,0,126049,0,3156062,0]
5	23	35	7	1 0 1 1 1 0	3	(1,2,23,124,576,2852,14192) [1,7,125,936,5915,36608,216972]
6	53	75	8	1 0 0 1 1 1	4	(1,15,65,321,1661,8396,42626) [3,85,490,3198,20557,123384,725389]
7	133	171	10	1 1 0 1 0 1	5	(8,31,160,892,4512,23307,121077) [42,201,1492,10469,62935,379644,2253373]
8	247	371	10	1 1 0 1 0 1	6	(36,0,990,0,26668,0) [239,0,11165,0,422030,0]
9	561	753	12	1 1 1 1 0 0	6	(10,77,303,1599,8565) [52,659,3265,21442,133697]

Table 2 Weight Spectra of Yasuda's et al Punctured Codes with $R = 3/4$, $3 \leqslant K \leqslant 9$

original code				punctured code		
K	G_1	G_2	d_f	[P]	d_f	$(a_n, n = d_f, d_f+1, d_f+2, \ldots)$ $[c_n, n = d_f, d_f+1, d_f+2, \ldots]$
3	5	7	5	1 0 1 1 1 1 0 0	2	(1,12,53,238,1091,4947) [1,36,309,2060,12320,69343]
4	15	17	6	1 0 1 1 1 1 0 0	3	(5,36,200,1070,5919) [14,194,1579,11313,77947]
5	23	35	7	1 0 1 0 1 1 0 1	3	(3,16,103,675,3969) (11,78,753,6901,51737]
6	53	75	8	1 0 0 0 1 1 1 1	4	(7,54,307,2005,12970) [40,381,3251,27123,213451]
7	133	171	10	1 1 1 1 1 0 0 0	4	(3,24,172,1158,7409) [12,188,1732,15256,121372]
8	247	371	10	1 0 1 0 1 1 0 1	5	(20,115,694,4816,32027) [168,1232,9120,78715,626483]
9	561	753	12	1 1 0 1 1 0 1 0	5	(7,49,351,2259,14749) [31,469,4205,34011,268650]

Table 3 Weight Spectra of Yasuda's et al Punctured Codes with $R = 4/5$, $3 \leqslant K \leqslant 9$

original code				punctured code		
K	G_1	G_2	d_f	[P]	d_f	$(a_n,\ n = d_f,\ d_f+1,\ d_f+2,\ \ldots)$ $[c_n,\ n = d_f,\ d_f+1,\ d_f+2,\ \ldots]$
3	5	7	5	1 0 1 1 1 1 1 0 0 0	2	(2,26,129,633,3316,17194) [2,111,974,6857,45555,288020]
4	15	17	6	1 0 1 0 0 1 1 0 1 1	3	(15,96,601,3918,25391) [63,697,6367,53574,426471]
5	23	35	7	1 0 1 1 1 1 1 0 0 0	3	(5,37,309,2282,16614) [20,265,3248,32328,297825]
6	53	75	8	1 0 0 0 0 1 1 1 1 1	4	(19,171,1251,9573,75167) [100,1592,17441,166331,1591841]
7	133	171	10	1 1 0 1 0 1 0 1 0 1	4	(14,69,654,4996,39699) [92,528,8694,79453,792114]
8	247	371	10	1 1 1 0 0 1 0 0 1 1	4	(2,51,415,3044,25530) [7,426,5244,49920,514857]
9	561	753	12	1 0 1 1 0 1 1 0 0 1	5	(19,187,1499,11809,95407) [168,2469,25174,242850,2320429]

Table 4 Weight Spectra of Yasuda's et al Punctured Codes with $R = 5/6,\ 3 \leqslant K \leqslant 9$

original code				punctured code		
K	G_1	G_2	d_f	[P]	d_f	$(a_n,\ n = d_f,\ d_f+1,\ d_f+2,\ \ldots)$ $[c_n,\ n = d_f,\ d_f+1,\ d_f+2,\ \ldots]$
3	5	7	5	1 0 1 1 1 1 1 1 0 0 0 0	2	(4,39,221,1330,8190,49754) [5,186,1942,16642,131415,981578]
4	15	17	6	1 0 0 0 1 1 1 1 1 1 0 0	2	(1,25,188,1416,10757) [2,134,1696,18284,179989]
5	23	35	7	1 0 1 0 1 0 1 1 0 1 0 1	3	(14,100,828,7198,60847) [69,779,9770,113537,1203746]
6	53	75	8	1 1 0 1 1 0 1 0 1 0 0 1	3	(5,55,517,4523,40476) [25,475,6302,73704,823440]
7	133	171	10	1 1 1 0 1 0 1 0 0 1 0 1	3	(1,20,223,1961,18093) [5,169,2725,32233,370861]
8	247	371	10	1 0 1 0 0 1 1 1 0 1 1 0	4	(11,155,1399,13018,122560) [85,1979,24038,282998,3224456]
9	561	753	12	1 1 0 1 1 0 1 0 1 0 0 1	4	(2,48,427,4153,39645) [9,447,5954,76660,912140]

Table 5 Weight Spectra of Yasuda's et al Punctured Codes with R = 6/7, $3 \leqslant K \leqslant 9$

original code				punctured code		
K	G_1	G_2	d_f	[P]	d_f	$(a_n, n = d_f, d_f+1, d_f+2, \ldots)$ $[c_n, n = d_f, d_f+1, d_f+2, \ldots]$
3	5	7	5	1 0 1 1 1 1 1 1 1 0 0 0 0 0	2	(6,66,408,2636,17844,119144) [8,393,4248,38142,325739,2647528]
4	15	17	6	1 0 0 0 0 1 0 1 1 1 1 1 0 1	2	(2,38,346,2772,23958) [4,219,3456,38973,437072]
5	23	35	7	1 0 1 0 0 1 1 1 1 0 1 1 0 0	3	(13,145,1471,14473,143110) [49,1414,21358,284324,3544716]
6	53	75	8	1 0 1 1 1 0 1 1 1 0 0 0 1 0	3	(9,122,1195,12139,123889) [60,1360,18971,252751,3165885]
7	133	171	10	1 1 1 1 0 1 0 1 0 0 0 1 0 1	3	(2,46,499,5291,56179) [9,500,7437,105707,1402743]
8	247	371	10	1 0 1 0 1 0 0 1 1 0 1 0 1 1	4	(26,264,2732,30389,328927) [258,3652,52824,746564,9825110]
9	561	753	12	1 1 0 1 0 1 1 1 0 1 0 1 0 0	4	(6,132,1289,13986,154839) [70,1842,24096,337514,4548454]

Table 6 Weight Spectra Yasuda's et al Punctured Codes with $R = 7/8$, $3 \leqslant K \leqslant 9$

original code				punctured code		
K	G_1	G_0	d_f	[P]	d_f	$(a_n,\ n = d_f,\ d_f+1,\ d_f+2,\ \ldots)$ $[c^n_n,\ n = d^f_f,\ d^f_f+1,\ d^f_f+2,\ \ldots]$
10	1167	1545	12	1 1 1 0	7	(1,10,29,94,415,1589,5956) [3,70,207,836,4411,19580,82154]
11	2335	3661	14	1 0 1 1	8	(1,21,65,226,907,3397,13223) [8,165,560,2321,10932,46921,204372]
12	4335	5723	15	1 1 1 0	9	(10,38,137,518,1990,7495,28907) [86,326,1379,6350,27194,114590,492275]
13	10533	17661	16	1 1 1 0	9	(4,8,45,193,604,2383,9412) [25,65,413,1991,6925,31304,139555]
14	21675	27123	16	1 1 1 0	10	(5,30,104,380,1486) [46,268,1066,4344,19992]
15	55367	63121	18	1 1 1 0	10	(2,6,37,153,582) (13,62,334,1606,7321]
16	111653	145665	19	1 1 1 0	10	(3,0,46,0,683,0) [28,0,397,0,7735,0]
17	347241	246277	20	1 1 1 0	12	(8,45,145) [68,495,1569]
18	506477	673711	20	1 0 1 1	12	(2,24,79) [11,253,889]
19	1352755	1771563	21	1 1 1 0	12	(2,11,27) [18,105,276]
20	2451321	3546713	22	1 1 1 0	12	(1,3,14) [9,21,139]
20	2142513	3276177	22	1 1 1 0	13	(10,34,101) [99,425,1425]
21	6567413	5322305	22	1 1 1 0	12	(1,0,18,0,333,0) [8,0,210,0,4290,0]
22	15724153	12076311	24	1 1 1 0	13	(1,1,20,62) [11,5,231,736]
23	33455341	24247063	24	1 0 1 1	14	(1,12) [17,163]

Table 7 Best Rate 3/4 10 ⩽ K ⩽ 23 Punctured Codes with their Weight Spectra, Perforation Matrix and original R = 1/2 Codes

original code				punctured code		
K	G_1	G_2	d_f	[P]	d_f	$(a_n, n = d_f, d_f+1, d_f+2, ...)$ $[c^n, n = d^f, d^f+1, d^f+2, ...]$
10	1167	1545	12	1 1 1 1 0 0	6	(1,3,24,150,852,4328) [6,16,202,1678,11538,69517]
11	2335	3661	14	1 0 1 1 1 0	6	(2,7,59,338,1646) [9,40,517,3731,22869]
12	4335	5723	15	1 0 0 1 1 1	7	(12,55,236,1271,6853) [107,628,3365,20655,126960]
13	10533	17661	16	1 1 0 1 0 1	7	(4,18,90,476,2466) [34,182,965,6294,38461]
14	21675	27123	16	1 1 0 1 0 1	7	(1,11,41,202,1334) [12,109,387,2711,20403]
15	55367	63121	18	1 0 1 1 1 0	8	(3,19,95,529) [28,159,1186,7461]
16	111653	145665	19	1 0 0 1 1 1	8	(1,14,47,259) [9,143,512,3571]
17	347241	246277	20	1 1 0 1 0 1	8	(1,5,28,167) [5,49,311,2266]
18	506477	673711	20	1 0 0 1 1 1	9	(1,13,101) [5,142,1375]
19	1352755	1771563	21	1 1 1 1 0 0	10	(6,51,217) [104,735,3368]
20	2451321	3546713	22	1 1 0 1 0 1	10	(4,18,81) [40,240,1219]
20	2142513	3276177	22	1 1 0 1 0 1	10	(4,18,89) [48,202,1248]
21	6567413	5322305	22	1 1 1 1 0 0	10	(4,19,82) [40,249,1510]
22	15724153	12076311	24	1 1 1 1 0 0	11	(8,19) [143,266]
23	33455341	24247063	24	1 0 0 1 1 1	12	(7,68) [79,1275]

Table 8 Best Rate 3/4 $10 \leqslant K \leqslant 23$ Punctured Codes with their Weight Spectra, Perforation Matrix and Original Codes