*Submitted to Electronics Letters*
Draft of 20 Dec 1995 11:04 a.m.

# The Effective Free Distance of Turbo Codes*

Dariush Divsalar and Robert J. McEliece

Jet Propulsion Laboratory and Department of Electrical Engineering
California Institute of Technology
Pasadena, California 91125, USA

*Indexing terms: Turbo codes, Convolutional codes*

*In this paper we will define and study the effective free distance of a turbo-code. If a turbo code is constructed from a number of component codes, we will argue that the effective free distance can be maximized by choosing the component codes to be IIR convolutional code fragments with maximal input-weight 2 free distance. We then present some theoretical bounds for, and some numerical tables of, IIR code fragments with maximal input-weight 2 free distance.*

*1. The Effective Free Distance of a Turbo-Code.* Turbo codes were introduced in [1], and have proved to have remarkably good performance in many applications ([2],[3],[4]). The view we take here is that a turbo code is a long block code with the structure shown in Figure 1. There are $L$ input bits, and each of these bits is encoded $q$ times. In the $j$th encoding, the $L$ bits are sent through a "permutation box" $P_j$ (often called a "interleaver" in the literature), and then encoded via an $(N_j, L)$ block encoder $G_j$, which can be thought of as an $L \times N_j$ matrix. Since in practice, $N_j$ may be less than or equal to $L$ for some or all values of $j$, we call the encoders $G_j$ "code fragment" encoders. The overall turbo code generated by the encoder depicted in Figure 1 is then a $(N_1 + \cdots + N_q, L)$ linear block code.

In current practice ([2],[3]), each of the $q$ code fragments is usually taken to be the $L$th truncation of either a systematic IIR (iunfinite impulse response) convolutional code, or an IIR convolutional code fragment. For this class of turbo-codes, several previous authors ([2], [3], [4]) have argued that on an AWGN channel, at the low signal to noise ratios where the codes are most effective, code performance is determined largely by the *weight-2 input minimum distance* ($d_2$) of the code, i.e., the minimum weight among codewords corresponding to input words of weight 2. The argument for this is roughly that $d_1$ is very large because of the IIR structure of the code fragments, and so can be ignored, and for

* The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. McEliece's contribution was also partially supported by NSF grant no. NCR-9505975 and a grant from Pacific Bell.

$i \geq 3$, the so-called "interleaver gain" makes the number of words of weight $d_i$ negligibly small. We thus define the *effective free distance* of the turbo code to be the value of $d_2$.

It is easy to deduce from Figure 1 that the input weight-$i$ minimum distance of the turbo code satisfies the following inequality:

$$d_i \geq d_i^{(1)} + \cdots + d_i^{(q)}, \tag{1.1}$$

where $d_i^{(j)}$ is the input-weight $i$ minimum distance of the $j$th code fragment. In particular, if we want the turbo-code to have a large value of $d_2$, in choosing the convolutional code fragments in Figure 1, it is desirable to choose them to be IIR, and to have the largest possible individual values of $d_2$. In the next section, we will state (without proofs) some theoretical results about the maximum possible value of $d_2$ for IIR convolutional code fragments, and then in Section 3 we will give some tables of IIR code fragments for which $d_2$ has been maximized. We hope that these tables will prove to be useful to engineers responsible for designing turbo-coded telecommunication systems..

*2. Bounds on $d_2$ for convolutional code fragments.* We define an encoder $G(D)$ for an $(r, k, m)$ *convolutional code fragment* to be a $k \times r$ matrix whose entries are causal rational functions in the indeterminate $D$. This definition differs from the usual definiton of a convolutional encoder in that we allow $r \leq k$.

**2.1 Definition.** *For $i = 1, 2, \ldots$, the input weight $i$ free distance of the convolutional encoder fragment $G(D)$ is*

$$d_i^{(G(D))} = \min\{|uG| : |u| = i\}, \tag{2.1}$$

*where in (2.1), $|x|$ denotes the weight of $x$.*

An $(r, k, m)$ convolutional encoder fragment $G(D)$ may be converted to a conventional *systematic* $(n, k, m)$ encoder $G'(D)$, with $n = r + k$, by defining

$$G'(D) = (\, I_k \quad G(D)\,), \tag{2.2}$$

where $I_k$ denotes the $k \times k$ identity matrix. A corollary to Definition 2.1 is that for the systematic code with generator matrix defined in (2.2), we have

$$d_i^{(G'(D))} = i + d_i^{(G(D))}. \tag{2.3}$$

If a turbo code is designed using truncated convolutional code fragments, then provided $d_i$ is finite, and that $L$ is sufficiently large, the value of $d_i$ for the $L$th truncation will be the same as the corresponding $d_i$ for the untruncated fragment. Thus by (1.1), the value of $d_2$ for the turbo code built from truncated convolutional code fragments with weight-2 free distances $d_2^{(1)}, \ldots, d_2^{(q)}$, will have effective free distance at least $d_2^{(1)} + \cdots + d_2^{(q)}$, if $L$ is sufficiently large.

We say that the encoder fragment $G(D)$ is *IIR* (infinite inpulse response), if every weight-1 input sequence produces an infinite weight output sequence. Equivalently, no row of $G(D)$ consists entirely of polynomials. If $G(D)$ is not IIR, then it is easy to see that $d_1 \le r(m+1)$, and $d_2 \le r(m+2)$. However, for IIR fragments, we have $d_1 = \infty$, and the following theorems.

**2.2 Theorem.** *If $G(D)$ is an IIR encoder for a $(r, k, m)$ convolutional code fragment, then its input weight-2 minimum distance $d_2$ must satisfy*

$$d_2 \le \min\left(\left\lceil \frac{2^m}{k} \right\rceil r,\ 2r + \left\lfloor \frac{2^{m-1} r}{k} \right\rfloor\right). \tag{2.4}$$

**2.3 Theorem.** *If $G(D)$ is an IIR encoder for an $(r, 1, m)$ convolutional code fragment with $m \ge 2$*, then*

$$d_2 \le (2 + 2^{m-1})r. \tag{2.5}$$

*Equality holds in (2.5) if and only if $G(D)$ is of the form*

$$G(D) = \left(\frac{P_1(D)}{Q(D)}, \ldots, \frac{P_r(D)}{Q(D)}\right), \tag{2.6}$$

*where $Q(D)$ is a primitive polynomial of degree $m$, and $P_1(D), \ldots, P_r(D)$ are each polynomials of degree $m$ with constant term 1, none of which are equal to $Q(D)$.*

*3. Tables.* In Tables 1–6, we give the generator matrices (in octal notation) for a number of convolutional code fragments with the largest possible values of $d_2$. In some cases, the maximum value of $d_2$ can be obtained only by code fragments with repeated columns in $G(D)$. Experiment shows that using such code fragments as components of a turbo code yields poor results. Thus when the optimal code fragment has repeated columns, we list it in *italics*, and immediately below we give the best code fragment with the same parameters, but which does not have repeated columns. When we found two or more code fragments tied for the largest value of $d_2$, we chose one with the largest value of $d_3$. If the code fragment is used to build a systematic encoder as in (2.2), the free distance of the resulting code is, by (2.3), $d_{\text{free}} = \min_i\{i + d_i\}$. In the tables, we denote the optimizing value of $i$ by $i^*$, and and corresponding $d_i$ by $d^*$. Thus for example, the systematic code corresponding to the $m = 3$ entry in Table 2 has $G(D) = \begin{pmatrix} 1 & 0 & (D+1)/(D^2+D+1) \\ 0 & 1 & (D^2+1)/(D^2+D+1) \end{pmatrix}$, $d_1 = \infty$, and by (2.3), we have $d_2 = 2 + 3 = 5$, $d_3 = 3 + 1 = 4$, and $d_{\text{free}} = d^* + i^* = 1 + 3 = 4$.

*Acknowledgement.* The authors thank Professor Solomon W. Golomb for his help in the derivation of Theorem 2.3.

* If $m = 1$, the bound in (2.5) can be improved to $d_2 \le r$, with equality if and only if $G(D) = (P_1(D)/(D+1), \ldots, P_r(D)/(D+1))$, where each $P_i(D)$ equals either 1 or $D$.

**References.**


1. G. Berrou, A. Glavieux, P. Thitimajshima, "Near Shannon limit error-correcting coding: Turbo codes," *Proc. 1993 International Conf. Comm.*, (Geneva, May 1993), pp. 1064–1070.

2. S. Benedetto and G. Montorsi, "Design of parallel concatenated convolutional codes," *IEEE Trans. Comm.*, in press.

3. D. Divsalar and F. Pollara, "On the design of turbo codes," *TDA Progress Report* vol. 42-123 (November 15, 1995), pp. 99–121.

4. S. Dolinar and D. Divsalar, "Weight distributions for turbo codes using random and non-random interleaving." *TDA Progress Report* vol. 42-122 (August 15, 1995), pp. 56–65.


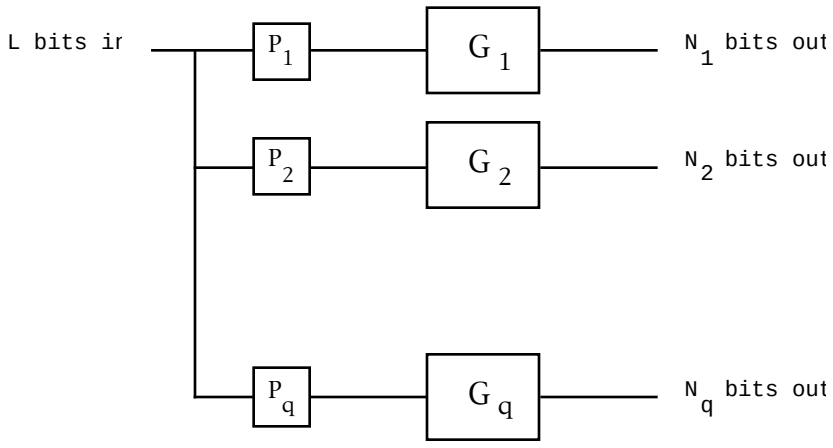


**Figure 1.** A General Turbo Code.

**Table 1. $d_2$-optimal $(1, 1, m)$ code fragments.**

$$G = (\, h_1/h_0 \,)$$

| $m$ | $h_0$ | $h_1$ | $d_2$ | $d_2^b$ | $d_3$ | $(d^*, i^*)$ |
|---|---|---|---|---|---|---|
| 1 | 3 | 2 | 1 | 1 | $\infty$ | $(1, 2)$ |
| 2 | 7 | 5 | 4 | 4 | 2 | $(2, 3)$ |
| 3 | 15 | 17 | 6 | 6 | 4 | $(2, 4)$ |
| 4 | 31 | 37 | 10 | 10 | 5 | $(2, 4)$ |
| 5 | 75 | 57 | 18 | 18 | 7 | $(4, 4)$ |
| 6 | 147 | 115 | 34 | 34 | 10 | $(4, 5)$ |

**Table 2. $d_2$-optimal $(1, 2, m)$ code fragments.**

$$G = (\, h_1/h_0 \quad h_2/h_0 \,)^T$$

| $m$ | $h_0$ | $h_1$ | $h_2$ | $d_2$ | $d_2^b$ | $d_3$ | $(d^*, i^*)$ |
|---|---|---|---|---|---|---|---|
| 1 | 3 | 1 | 1 | 0 | 1 | $\infty$ | $(0, 2)$ |
| 2 | 7 | 3 | 5 | 2 | 2 | 0 | $(0, 3)$ |
| 3 | 13 | 15 | 17 | 3 | 4 | 1 | $(1, 3)$ |
| 4 | 23 | 35 | 27 | 6 | 6 | 2 | $(2, 3)$ |
| 5 | 45 | 43 | 61 | 10 | 10 | 3 | $(3, 3)$ |

**Table 3. $d_2$-optimal $(2, 1, m)$ code fragments.**

$$G = (\, h_1/h_0 \quad h_2/h_0 \,)$$

| $m$ | $h_0$ | $h_1$ | $h_2$ | $d_2$ | $d_2^b$ | $d_3$ | $(d^*, i^*)$ |
|---|---|---|---|---|---|---|---|
| 1 | 3 | 2 | 1 | 2 | 2 | $\infty$ | $(2, 2)$ |
| *2* | *7* | *5* | *5* | *8* | *8* | *4* | $(4, 3)$ |
| 2 | 7 | 5 | 3 | 6 | 8 | 4 | $(4, 3)$ |
| 3 | 13 | 17 | 15 | 12 | 12 | 7 | $(7, 3)$ |
| 4 | 23 | 33 | 37 | 20 | 20 | 9 | $(6, 4)$ |
| 5 | 73 | 45 | 51 | 36 | 36 | 14 | $(6, 5)$ |
| 6 | 147 | 115 | 101 | 68 | 68 | 20 | $(6, 5)$ |

**Table 4.** $d_2$**-optimal** $(1,3,m)$ **code fragments.**

$$G = (\, h_1/h_0 \quad h_2/h_0 \quad h_3/h_0 \,)^T$$

| $m$ | $h_0$ | $h_1$ | $h_2$ | $h_3$ | $d_2$ | $d_2^b$ | $d_3$ | $(d^*, i^*)$ |
|---|---|---|---|---|---|---|---|---|
| 2 | 7 | 5 | 3 | 1 | 1 | 2 | 0 | $(1,2)$ |
| 3 | 13 | 15 | 17 | 11 | 2 | 3 | 1 | $(2,2)$ |
| 4 | 23 | 35 | 33 | 25 | 3 | 4 | 1 | $(1,3)$ |
| 5 | 51 | 47 | 45 | 63 | 7 | 7 | 2 | $(2,3)$ |

**Table 5.** $d_2$**-optimal** $(2,2,m)$ **code fragments.**

$$G = \begin{pmatrix} h_1/h_0 & h_2/h_0 \\ h_3/h_0 & h_4/h_0 \end{pmatrix}$$

| $m$ | $h_0$ | $h_1$ | $h_2$ | $h_3$ | $h_4$ | $d_2$ | $d_2^b$ | $d_3$ | $(d^*, i^*)$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 3 | 1 | 2 | 2 | 1 | 2 | 2 | $\infty$ | $(2,2)$ |
| *2* | *7* | *3* | *3* | *5* | *5* | *4* | *4* | *0* | $(0,3)$ |
| 2 | 7 | 1 | 5 | 5 | 3 | 3 | 4 | 1 | $(1,3)$ |
| 3 | 15 | 3 | 11 | 11 | 13 | 7 | 8 | 3 | $(1,4)$ |
| 4 | 31 | 35 | 23 | 23 | 21 | 12 | 12 | 3 | $(3,3)$ |

**Table 6.** $d_2$**-optimal** $(3,1,m)$ **code fragments.**

$$G = (\, h_1/h_0 \quad h_2/h_0 \quad h_3/h_0 \,)$$

| $m$ | $h_0$ | $h_1$ | $h_2$ | $h_3$ | $d_2$ | $d_2^b$ | $d_3$ | $(d^*, i^*)$ |
|---|---|---|---|---|---|---|---|---|
| *1* | *3* | *2* | *1* | *1* | *3* | *3* | $\infty$ | $(3,2)$ |
| *2* | *7* | *5* | *5* | *5* | *12* | *12* | *6* | $(6,3)$ |
| 2 | 7 | 5 | 3 | 6 | 8 | 12 | 6 | $(6,3)$ |
| 3 | 13 | 17 | 15 | 11 | 18 | 18 | 9 | $(9,3)$ |
| 4 | 23 | 35 | 27 | 37 | 30 | 30 | 13 | $(10,4)$ |
| 5 | 73 | 45 | 51 | 47 | 54 | 54 | 20 | $(10,5)$ |
| 6 | 147 | 115 | 101 | 135 | 102 | 102 | 29 | $(11,5)$ |