Logarithms, Syndrome Functions, and the Information Rates of Greedy Loop Transversal Codes

F.-L. Hsu, F.A. Hummer and J.D.H. Smith
Department of Mathematics
Iowa State University
Ames, IA 50011, U.S.A.

ABSTRACT. The paper studies linear block codes and syndrome functions built by the greedy loop transversal algorithm. The syndrome functions in the binary white-noise case are generalizations of the logarithm, with curious fractal properties. The codes in the binary white-noise case coincide with lexicodes: their dimensions are listed for channel lengths up to the sixties, and up to the three hundreds for double errors. In the ternary double-error case, record-breaking codes of lengths 43 to 68 are constructed.

1 Introduction

The general loop transversal approach to the construction of linear block codes was introduced in [Sm]. A companion paper [HS] gives further details, concentrating on the greedy loop transversal algorithm in the binary case. In particular, it is shown there [HS, Theorem 6.1] that the greedy loop transversal algorithm provides an alternative method for building binary lexicodes, especially suitable for good channels. The current paper has two aims. The first is to present data on the dimensions of the codes of various lengths constructed by the greedy loop transversal algorithm. (The phrase "loop transversal code" is abbreviated here to "LT code".) For binary channels, double, triple and quadruple white-noise error patterns are treated, corresponding to minimum distances 5, 6 (Table 2) and 7–10 (Table 3) in metric language. The data may be read as giving the dimensions of lexicodes, for channel lengths beyond 300 in the double-error case. For ternary channels, greedy loop transversal codes and lexicodes differ, since the former are linear while the latter are not. Table 4 gives the dimensions of the ternary Hamming double-error correcting greedy loop

JCMCC 22 (1996), pp. 33-49

transversal codes, for channel lengths up to 68. They include the perfect ternary Golay code, and new record-breaking codes for lengths above 42.

The second aim of the paper is to draw attention to the syndrome functions constructed by the greedy loop transversal algorithm for binary white-noise error patterns. For single errors, the syndrome (3.1) is essentially the logarithm function, so for other white-noise error patterns the syndromes may be considered as generalizations of the logarithm. Their graphs (Figures 1–3) display curious fractal properties that warrant further investigation. Another mysterious feature of the syndromes is the apparent convergence of the "efficiencies" defined below (3.3) and recorded in Table 1.

The presentation of the graphical and tabular data in Sections 3 and 4 is prefaced by a description of the greedy loop transversal algorithm in Section 2, for arbitrary error patterns in channels over alphabets that are prime fields.

2 The greedy loop transversal algorithm

Fix a prime p. Each natural number n (including 0) has a unique expansion

$$n = \sum_{i=0}^{\infty} n(i)p^i \tag{2.1}$$

with $0 \leq n(i) < p$ for each i. Moreover, $n(i) = 0$ for $i > \lfloor \log_p n \rfloor$. For $d > \lfloor \log_p n \rfloor$, the natural number n may be identified with the vector $(n(d-1), \ldots, n(1), n(0))$ in the d-dimensional vector space $V_d = GF(p)^d$ over the Galois field $GF(p)$ of order p. Thus the set $\mathbb{N}$ of natural numbers is identified with the nested union $\bigcup_{d>0} V_d$ of vector spaces. The induced addition and subtraction operations on integers are written as $+_p$ and $-_p$ to avoid confusion with the usual addition and subtraction. For example, both $+_2$ and $-_2$ are the "nim sum" of [Co, p. 51]. Besides the usual (well-)ordering $\leq$, the set of natural numbers carries a partial ordering $\subseteq_p$, known as the *Hamming order*, defined by

$$m \subseteq_p n \Leftrightarrow \exists F \subset \mathbb{N}. \quad m = n - \sum_{i \in F} n(i)p^i. \tag{2.2}$$

In other words, m is bounded above by n in the Hamming order if and only if the expansion (2.1) for m is obtained from the expansion for n by replacing certain digits $n(i)$ – namely those with i in F – by the digit 0. Note that 0 is the bottom element of $(N, \subseteq_p)$, and that the Hamming distance between m and n is the rank of $m -_p n$ in the poset $(\mathbb{N}, \subseteq_p)$.

A subset X of a poset $(Y, \sqsubseteq)$ is said to be *self-subordinate* if $y \sqsubseteq x \in X$ implies $y \in X$. A self-subordinate subset E of $(\mathbb{N}, \subseteq_p)$ is called an *error pattern* if it contains the set $p^{\mathbb{N}} = \{p^i \mid i \in \mathbb{N}\}$. Error patterns model sets of

errors to be corrected in the various channels V_d. For example, $\{\alpha p^i \mid \alpha \in GF(p); i \in \mathbb{N}\}$ comprises the errors of Hamming weight at most 1. White noise double errors are modeled by $\{\alpha p^i +_p \beta p^j \mid \alpha, \beta \in GF(p); i, j \in \mathbb{N}\}$. Burst double errors are modeled by $\{(\alpha p^i +_p \beta p^j \mid \alpha, \beta \in GF(p); i, j \in \mathbb{N}, |i-j| \leq 1\}$. Error patterns form partial algebras under the operations of the vector space $(\mathbb{N}, +_p, GF(p))$. For example, the sum $p^i +_p p^j$ is defined in the burst double error pattern if and only if $|i-j| \leq 1$.

Suppose that an error pattern E is given. Then an *E-syndrome*, or just *syndrome*, is a partial function $s\colon E \to \mathbb{N}$ which:

(a) injects;

(b) is a partial vector space homomorphism; (2.3)

(c) has domain self-subordinate in $(E, \leq)$, and

(d) satisfies: $\forall n \in \mathbb{N}, \exists r \in \mathbb{N}.\ p^{\mathbb{N}} \cap s(V_n \cap E)$ spans V_r.

The syndrome is said to be *proper* if s is a properly partial function. In view of (c), this is equivalent to finiteness of the domain of s. For a proper syndrome, the *length* is defined to be

$$n = \max\{1 + \lfloor \log_p m \rfloor \mid m \in \operatorname{dom} s\}. \tag{2.4}$$

The *redundancy* is defined to be

$$r = \max\{1 \dot{+} \lfloor \log_p(ms) \rfloor \mid m \in \operatorname{dom} s\}. \tag{2.5}$$

A proper syndrome s defines a *parity map*

$$\varepsilon_s\colon V_n \to V_r \tag{2.6}$$

by linearity and $p^i\varepsilon_s = p^i s$ for $i < n$. By (c), these values $p^i s$ are defined. Condition (b) guarantees that s agrees with ε_s on $V_n \cap E$. Condition (d) yields that ε_s surjects. Condition (a) guarantees that dom s embeds into V_r under ε_s. A code C_n in the channel V_n correcting the set $V_n \cap E$ of errors, and having dimension $n-r$, is then given as the kernel of ε_s.

The greedy loop transversal algorithm determines an E-syndrome s by the partial linearity (2.3) (b) and the greedy choice of $p^n s$ given that $s\colon (V_n \cap E) \to \mathbb{N}$ has already been defined. In other words, $p^n s$ is the minimal element of the set of integers m satisfying the requirement

$$\begin{aligned} &\forall e \neq f \in V_{n+1} \cap E, \\ &e(n)m +_p (e -_p (e(n)p^n))s \neq f(n)m +_p (f -_p f(n)p^n)s. \end{aligned} \tag{2.7}$$

Then for $e \in (V_{n+1} - (V_n \cup \{p^n\})) \cap E$,

$$es = e(n)(p^n s) +_p (e -_p e(n)p^n)s. \tag{2.8}$$

In (2.7) and (2.8), juxtaposition of an element of $GF(p)$ and an integer denotes the scalar multiple of that integer by the element of $GF(p)$. Note that the partial linearity requirement (2.3) (b) initializes the algorithm with $0s = 0$. If E is closed under scalar multiplication, then (2.7) may be simplified. The greedy algorithm picks $p^n s$ to be the least integer not in *anathema*, the set

$$\{es +_p fs | e, f \in V_n \cap E; p^n +_p e \in E\} \tag{2.9}$$

(cf. [Sm, (5.1)]).

3 Binary white-noise syndrome functions

If E is the binary white-noise single error pattern $\{0\} \cup 2^{\mathbb{N}}$, the greedy loop transversal algorithm builds the improper syndrome function s_1 with $0s_1 = 0$ and

$$s_1 \colon 2^{\mathbb{N}} \to \mathbb{N}; x \mapsto 1 + \log_2 x. \tag{3.1}$$

In this sense, the syndrome function for single errors is essentially the logarithm function. One may then regard the syndrome function s_t built by the greedy loop transversal algorithm for the binary white-noise t-error pattern E_t as a generalization of the logarithm function. To facilitate comparison with the logarithm function, the syndrome function

$$s_t \colon E_t \to \mathbb{N}; x \mapsto y \tag{3.2}$$

is graphed with $\log_2 x$ on the ordinate and y on the abscissa in Figures $(t-1)$ (a), for $t = 2, 3, 4$. With a similar convention, the graph of s_1 would appear as a straight line of slope 1. Figures $(t-1)$ (b) "plot the graphs of Figures $(t-1)$ (a) on logarithmic paper," i.e. they graph $s_t \colon x \mapsto y$ by plotting $\log_2 \log_2 x$ against $\log_2 y$. The fractal form of Figures 1–3 is very striking, and clearly warrants further investigation. As an initial step in such an investigation, define a *nodal point* of the syndrome function s_t to be a point on its graph of the form $(2^{n-1}, 2^k)$ for integral n, k. (The analysis given in [HS] shows that the graph includes such points.) The proper syndrome given by the restriction of s_t to the channel V_n then yields a t-error correcting code C_n with redundancy $k+1$. By the sphere-packing bound,

$$2^{k+1} \geq \binom{n}{t} + \binom{n}{t-1} + \cdots + \binom{n}{1} + \binom{n}{0}. \tag{3.3}$$

The *efficiency* of the code C_n of redundancy $k+1$ is the ratio of $\log_2[\binom{n}{t} + \cdots + \binom{n}{0}]$ to $(k+1)$, usually expressed as a percentage. Table 1 lists these efficiencies for $0 < k < 19$ and $1 < t < 5$. Their apparent convergence to about 80% is rather curious.

$k+1$	2-Error	3-Error	4-Error
19	79.61	79.67	79.76
18	79.73	82.44	79.71
17	79.99	78.74	78.90
16	80.31	77.64	80.51
15	80.63	78.98	79.89
14	80.57	82.34	80.70
13	80.64	87.37	84.02
12	80.08	93.19	84.12
11	80.58	85.86	87.57
10	79.89	82.24	91.34
9	82.51	82.88	95.47
8	78.80	87.78	91.86
7	78.91	93.42	94.71
6	80.97	89.87	97.21
5	89.19	94.01	99.08
4	86.49	97.67	100.00
3	93.58	100.00	100.00
2	100.00	100.00	100.00

Table 1.
Efficiency of binary 2-, 3-, and 4-error greedy LT codes

4 Dimensions of greedy LT codes

The final three tables record the dimensions of the codes C_n of length n constructed by the greedy loop transversal algorithm. For each n, the second column of Table 2 records the dimension of the greedy LT code correcting binary white-noise double errors, i.e. of minimum Hamming distance $d = 5$. The third column records the dimension of the corresponding codes of minimum Hamming distance $d = 6$ obtained by adjoining a parity check (thus increasing the length by 1). For lengths less than 2^7, the entries in parentheses are the dimensions of the best known linear code of length n and minimum distance d, as recorded in [Ve]. Table 3 records analogous data for binary white-noise triple and quadruple errors, i.e. minimum distances $d = 7, 8, 9, 10$. Since binary greedy LT codes are the same as lexicodes [HS, Theorem 6.1], Tables 2 and 3 may also be read as giving the dimensions of lexicodes. Table 4 lists the dimensions of the ternary greedy LT codes correcting white-noise Hamming double errors, i.e. with minimum Hamming distance $d = 5$. It is interesting to note that the code of length 11 has dimension 6, so that it coincides with the perfect ternary Golay code. The numbers in parentheses list the dimensions of the best known ternary linear codes of minimum distance 5, as recorded in [KP]. The greedy LT

codes of lengths 43 to 50 are better than the best known to Kschischang and Pasupathy. The data in [KP] stopped at length 50, but the greedy LT codes of lengths 51 onwards are also likely to be world records.

References

[Co] J.H. Conway, *On Numbers and Games*, Camb. Univ. Press, Cambridge, 1975.

[HS] F.A. Hummer and J.D.H. Smith, Greedy loop transversal codes, metrics, and lexicodes, this volume.

[KP] F.R. Kschischang and S. Pasupathy, Some ternary and quaternary codes and associated sphere packings, *I.E.E.E. Trans.. Info. Th.* **IT-38** (1992), 227–246.

[Sm] J.D.H. Smith, Loop transversals to linear codes, *J. Comb., Info. and Syst. Sci.* **17** (1992), 1–8.

[Ve] T. Verhoeff, An updated table of minimum-distance bounds for binary linear codes, *I.E.E.E. Trans.. Info. Th.* **IT-33** (1987), 665–680.

n	d = 5	d = 6	n	d = 5	d = 6	n	d = 5	d = 6
12	4(4)	4(4)	48	36(26)	35(35)	84	70(70)	69(69)
13	5(5)	4(4)	49	37(37)	36(36)	85	71(71)	70(70)
14	6(6)	5(5)	50	38(38)	37(37)	86	72(72)	71(71)
15	7(7)	6(6)	51	39(39)	38(38)	87	73(73)	72(72)
16	8(8)	7(7)	52	40(40)	39(39)	88	74(74)	73(73)
17	9(0	8(8)	53	41(41)	40(40)	89	75(75)	74(74)
18	9(9)	9(9)	54	41(42)	40(41)	90	76(76)	75(75)
19	10(10)	9(9)	55	42(43)	41(42)	91	77(77)	76(76)
20	11(11)	10(10)	56	43(44)	42(43)	92	78(78)	77(77)
21	12(12)	11(11)	57	44(45)	43(44)	93	78(79)	78(78)
22	12(13)	12(12)	58	45(46)	44(45)	94	79(80)	78(79)
23	13(14)	12(13)	59	46(47)	45(46)	95	80(81)	79(80)
24	14(14)	13(14)	60	47(48)	46(47)	96	81(82)	80(81)
25	15(15)	14(14)	61	48(49)	47(48)	97	82(83)	81(82)
26	16(16)	15(15)	62	49(50)	48(49)	98	83(84)	82(83)
27	17(17)	16(16)	63	50(51)	49(50)	99	84(85)	83(84)
28	18(18)	17(17)	64	51(52)	50(51)	100	85(86)	84(85)
29	19(19)	18(18)	65	52(53)	51(52)	101	86(87)	85(86)
30	19(20)	19(19)	66	52(52)	52(53)	102	87(88)	86(87)
31	20(21)	19(20)	67	54(54)	53(53)	103	88(89)	87(88)
32	21(22)	20(21)	68	55(55)	54(54)	104	89(90)	88(89)
33	22(22)	21(22)	69	56(56)	55(55)	105	90(91)	89(90)
34	23(23)	22(22)	70	56(57)	56(56)	106	91(92)	90(91)
35	24(24)	23(23)	71	57(58)	56(57)	107	92(93)	91(92)
36	25(25)	24(24)	72	58(59)	57(58)	108	93(94)	92(93)
37	26(26)	25(25)	73	59(60)	58(59)	109	94(95)	93(94)
38	27(27)	26(26)	74	60(61)	59(60)	110	95(96)	94(95)
39	27(28)	27(27)	75	61(61)	60(61)	111	96(97)	95(96)
40	28(29)	27(28)	76	62(62)	61(61)	112	97(98)	96(97)
41	29(30)	28(29)	77	63(63)	62(62)	113	98(99)	97(98)
42	30(30)	29(30)	78	64(64)	63(63)	114	99(100)	98(99)
43	31(31)	30(30)	79	65(65)	64(64)	115	100(101)	99(100)
44	32(32)	31(31)	80	66(66)	65(65)	116	101(102)	100(101)
45	33(33)	32(32)	81	67(67)	66(66)	117	102(103)	101(102)
46	34(34)	33(33)	82	68(68)	67(67)	118	103(104)	102(103)
47	35(35)	34(34)	83	69(69)	68(68)	119	104(105)	103(104)

Note: numbers inside ( ) are data from Tom Verhoeff in IEEE 1987

Table 2: dimensions of binary greedy LT codes

n	d = 5	d = 6	n	d = 5	d = 6	n	d = 5	d = 6
120	105(106)	104(105)	**156**	14)	139	**192**	175	174
121	105(107)	105(106)	**157**	140	140	**193**	176	175
122	106(108)	105(107)	**158**	141	140	**194**	177	176
123	107(109)	106(108)	**159**	142	141	**195**	178	177
124	108(110)	107(109)	**160**	143	142	**196**	179	178
125	109(111)	108(110)	**161**	144	143	**197**	180	179
126	110(112)	109(111)	**162**	145	144	**198**	181	180
127	111(113)	110(112)	**163**	146	145	**199**	182	181
128	112	111	**164**	147	146	**200**	183	182
129	113	112	**165**	148	147	**201**	184	183
130	114	113	**166**	149	148	**202**	185	184
131	115	114	**167**	150	149	**203**	186	185
132	116	115	**168**	151	150	**204**	186	186
133	117	116	**169**	152	151	**205**	187	186
134	118	117	**170**	153	152	**206**	188	187
135	119	118	**171**	154	153	**207**	189	188
136	120	119	**172**	155	154	**208**	190	189
137	121	120	**173**	156	155	**209**	191	190
138	122	121	**174**	157	156	**210**	192	191
139	123	122	**175**	158	157	**211**	193	192
140	124	123	**176**	159	158	**212**	194	193
141	125	124	**177**	160	159	**213**	195	194
142	126	125	**178**	161	160	**214**	196	195
143	127	126	**179**	162	161	**215**	197	196
144	128	127	**180**	163	162	**216**	198	197
145	129	128	**181**	164	163	**217**	199	198
146	130	129	**182**	165	164	**218**	200	199
147	131	130	**183**	166	165	**219**	201	200
148	132	131	**184**	167	166	**220**	202	201
149	133	132	**185**	168	167	**221**	203	202
150	134	133	**186**	169	168	**222**	204	203
151	135	134	**187**	170	169	**223**	205	204
152	136	135	**188**	171	170	**224**	206	205
153	137	136	**189**	172	171	**225**	207	206
154	138	137	**190**	173	172	**226**	208	207
155	139	138	**191**	174	173	**227**	209	208

Table 2 (cont): dimensions of binary greedy LT codes

n	d = 5	d = 6	n	d = 5	d = 6	n	d = 5	d = 6
228	210	209	**264**	246	245	**300**	281	280
229	211	210	**265**	247	246	**301**	282	281
230	212	211	**266**	248	247	**302**	283	282
231	213	212	**267**	248	248	**303**	284	283
232	214	213	**268**	249	248	**304**	285	284
233	215	214	**269**	250	249	**305**	286	285
234	216	215	**270**	251	250	**306**	287	286
235	217	216	**271**	252	251	**307**	288	287
236	218	217	**272**	253	252	**308**	289	288
237	219	218	**273**	254	253	**309**	290	289
238	220	219	**274**	255	254	**310**	291	290
239	221	220	**275**	256	255	**311**	292	291
240	222	221	**276**	257	256	**312**	293	292
241	223	222	**277**	258	257	**313**	294	293
242	224	223	**278**	259	258	**314**	295	294
243	225	224	**279**	260	259	**315**	296	295
244	226	225	**280**	261	260	**316**	297	296
245	227	226	**281**	262	261	**317**	298	297
246	228	227	**282**	263	262	**318**	299	298
247	229	228	**283**	264	263	**319**	300	299
248	230	229	**284**	265	264	**320**	301	300
249	231	230	**285**	266	265	**321**	302	301
250	232	231	**286**	267	266	**322**	303	302
251	233	232	**287**	268	267	**323**	304	303
252	234	233	**288**	269	268	**324**	305	304
253	235	234	**289**	270	269	**325**	306	305
254	236	235	**290**	271	270	**326**	307	306
255	237	236	**291**	272	271	**327**	308	307
256	238	237	**292**	273	272	**328**	309	308
257	239	238	**293**	274	273	**329**	310	309
258	240	239	**294**	275	274	**330**	311	310
259	241	240	**295**	276	275	**331**	312	311
260	242	241	**296**	277	276	**332**	313	312
261	243	242	**297**	278	277	**333**	314	313
262	244	243	**298**	279	278	**334**	315	314
263	245	244	**299**	280	279	**335**	316	315

Table 2 (cont): dimensions of binary greedy LT codes

n	d = 5	d = 6	n	d = 5	d = 6	n	d = 5	d = 6
336	317	316	**337**	318	317	**338**	319	318
339	320	319	**340**	321	320	**341**	322	321
342	323	322	**343**	323	323	**344**	324	323
345	325	324	**346**	326	325	**347**	327	326
348	328	327	**349**	329	328	**350**	330	329
351	331	330	**352**	332	331	**353**	333	332
354	334	333	**355**	335	334	**356**	336	335
357	337	336	**358**	338	337	**359**	339	338
360	340	339						

Table 2 (concl): dimensions of binary greedy LT codes

n	d = 7	d = 8	d = 9	d = 10	n	d = 7	d = 8
12	2(2)	2(2)	1(1)	1(1)	**39**	23(23)	22(22)
13	3(3)	2(2)	1(1)	1(1)	**40**	23(24)	23(23)
14	4(4)	3(3)	2(2)	1(1)	**41**	24(25)	23(24)
15	5(5)	4(4)	2(2)	2(2)	**42**	24(26)	24(25)
16	5(5)	5(5)	2(2)	2(2)	**43**	26(27)	25(26)
17	6(6)	5(5)	3(3)	2(2)	**44**	27(28)	26(27)
18	7(7)	6(6)	3(3)	3(3)	**45**	28(29)	27(28)
19	8(8)	7(7)	4(4)	3(3)	**46**	29(30)	28(29)
20	9(9)	8(8)	5(5)	4(4)	**47**	30(31)	29(30)
21	10(10)	9(9)	5(5)	5(5)	**48**	31(31)	30(31)
22	11(11)	10(10)	6(6)	5(5)	**49**	32(32)	31(31)
23	12(12)	11(11)	6(7)	6(6)	**50**	33(33)	32(32)
24	12(12)	12(12)	7(7)	6(7)	**51**	34(34)	33(33)
25	12(12)	12(12)	8(8)	7(7)	**52**	35(35)	34(34)
26	12(13)	12(12)	9(9)	8(8)	**53**	36(36)	35(35)
27	13(14)	12(13)	9(10)	9(9)	**54**	37(37)	36(36)
28	13(14)	13(14)	10(10)	9(10)	**55**	38(38)	37(37)
29	14(15)	13(14)	11(11)	10(10)	**56**	38(39)	38(38)
30	15(16)	14(15)	12(12)	11(11)	**57**	39(40	38(39)
31	16(17)	15(16)	12(12)	12(12)	**58**	40(41)	39(40)
32	16(17)	16(17)	13(13)	12(12)	**59**	41(42)	40(41)
33	17(18)	16(17)	14(14)	13(13)	**60**	41(43)	41(42)
34	18(19)	17(18)		14(14)	**61**	42(44)	41(43)
35	19(20)	18(19)			**62**	43(45)	42(44)
36	20(20)	19(20)			**63**	44(46)	43(45)
37	21(21)	20(20			**64**		44(46)
38	22(22)	21(21)					

Table 3: dimensions of binary greedy LT codes

n	d = 5	n	d = 5	n	d = 5
5	1(1)	27	18(10)	49	39(38)
6	1(1)	28	19(20)	50	40(39)
7	2(2)	29	20(21)	51	41
8	3(3)	30	21(22)	52	42
9	4(4)	31	22(23)	53	43
10	5(5)	32	23(24)	54	44
11	6(6)	33	24(25)	55	45
12	6(6)	34	25(26)	56	46
13	6(7)	35	26(27)	57	47
14	7(8)	36	27(28)	58	48
15	8(8)	37	28(20)	59	49
16	8(9)	38	29(30)	60	50
17	9(10)	39	30(31)	61	51
18	19(11)	40	31(32)	62	52
19	11(12)	41	32(33)	63	53
20	12(13)	42	33(33)	64	54
21	13(14)	43	34(33)	65	55
22	14(15)	44	35(33)	66	56
23	15(16)	45	36(34)	67	57
24	16(17)	46	37(35)	68	58
25	16(18)	47	37(36)	69	
26	17(19)	48	38(37)	70	

Note: numbers inside ( ) are data from Kschischang and Pasupathy in IEEE 1992

Table 4: dimensions of ternary greedy LT 2-Error codes

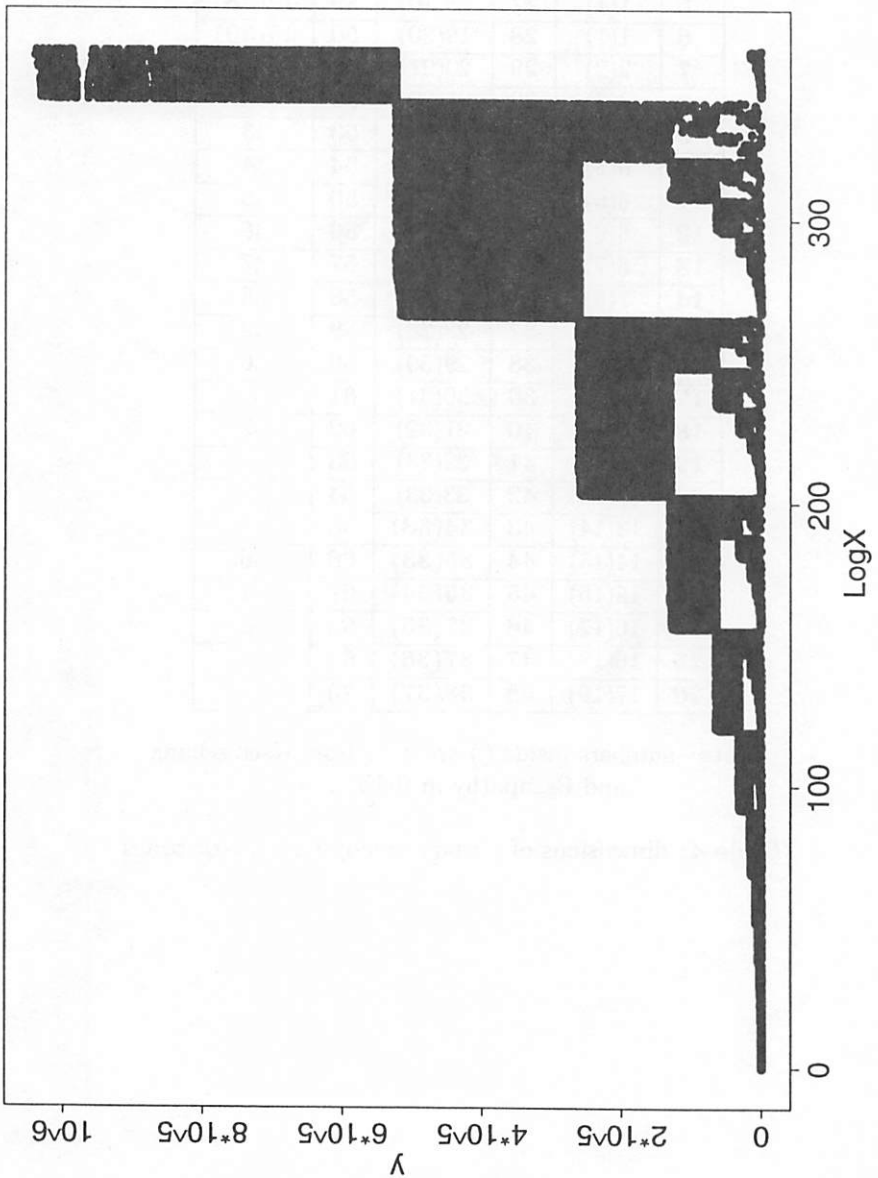

Figure 1(a): syndrome function for binary 2-error

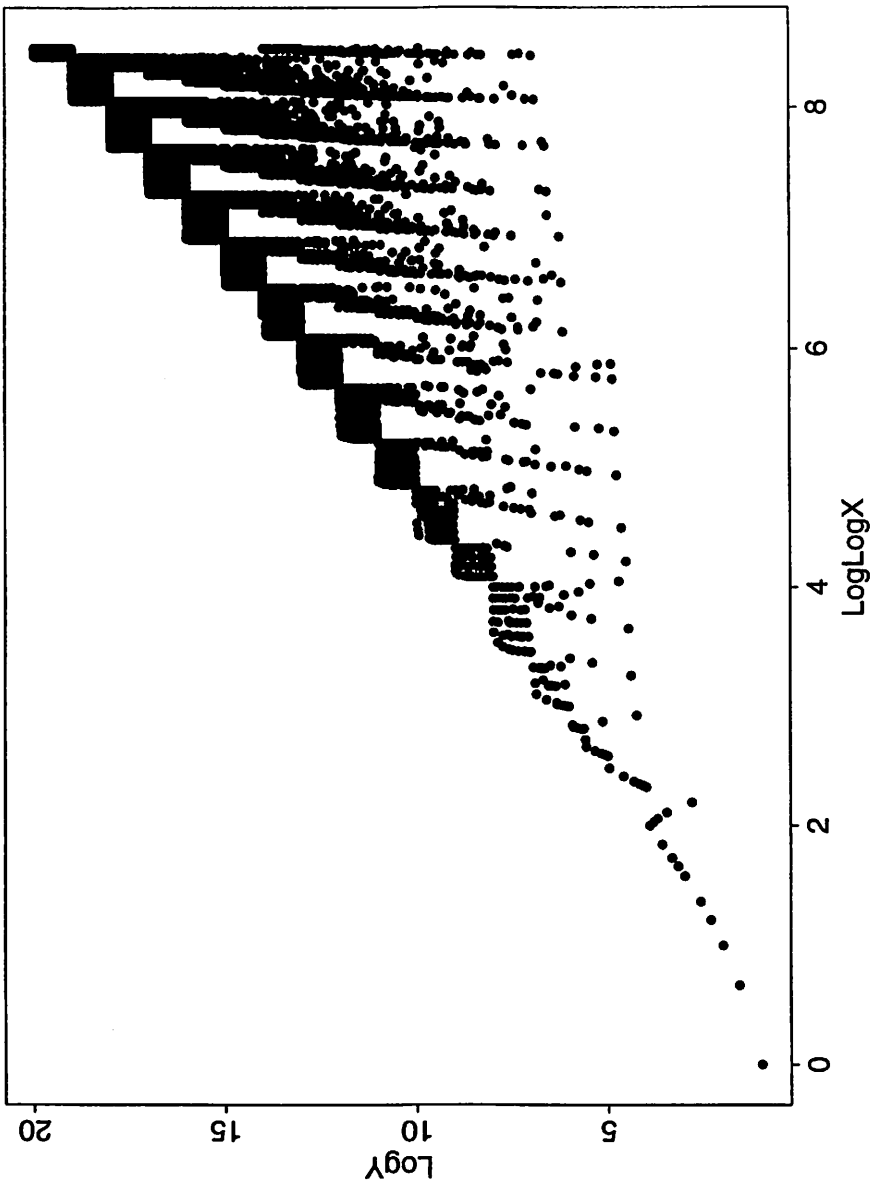

Figure 1(b): syndrome function for binary 2-error

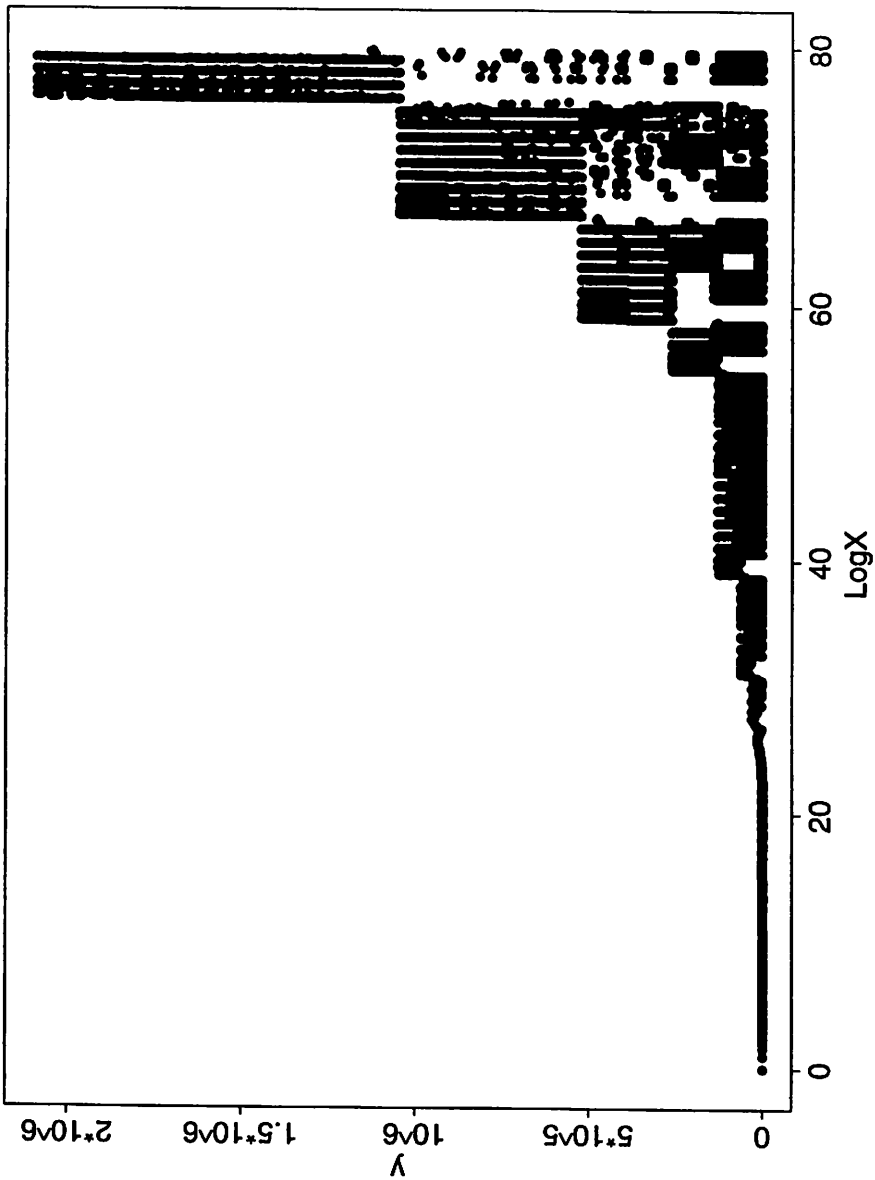

Figure 2(a): syndrome function for binary 3-error

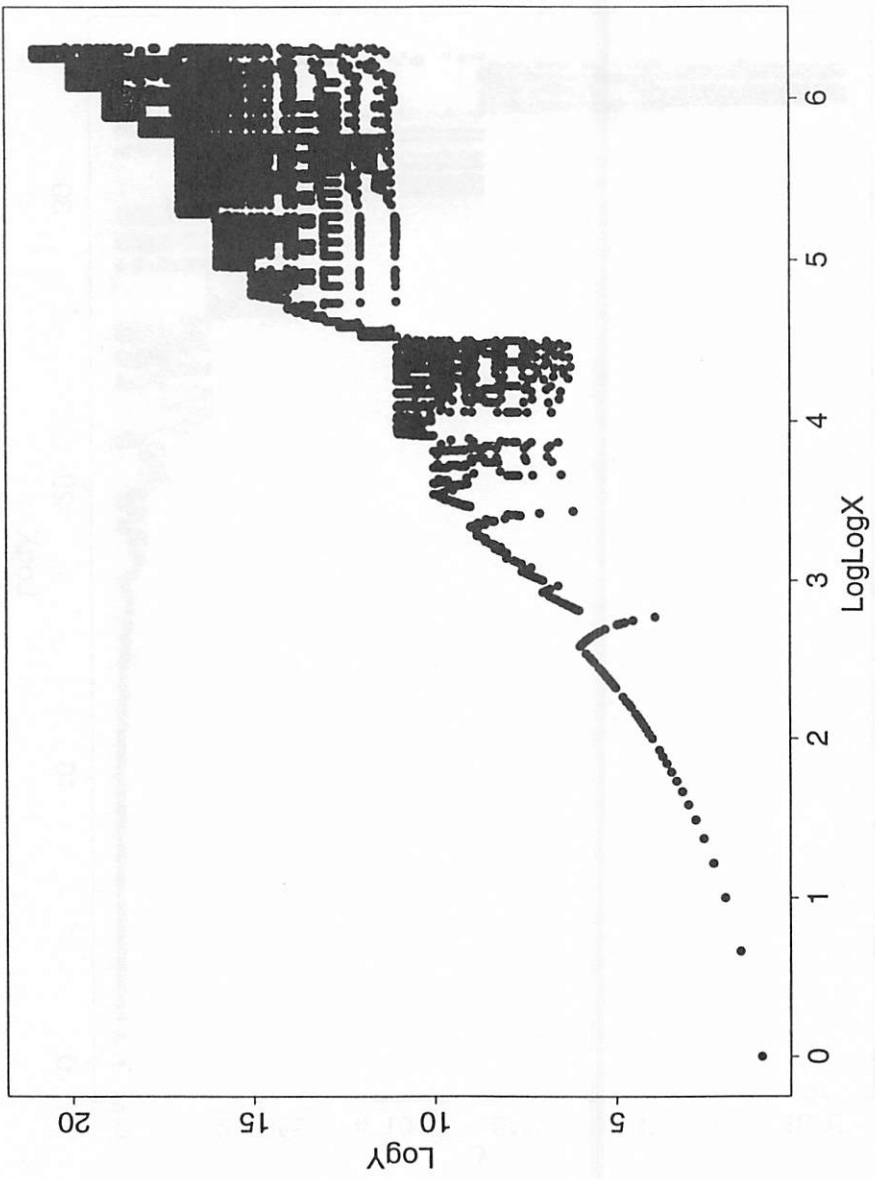

Figure 2(b): syndrome function for binary 3-error

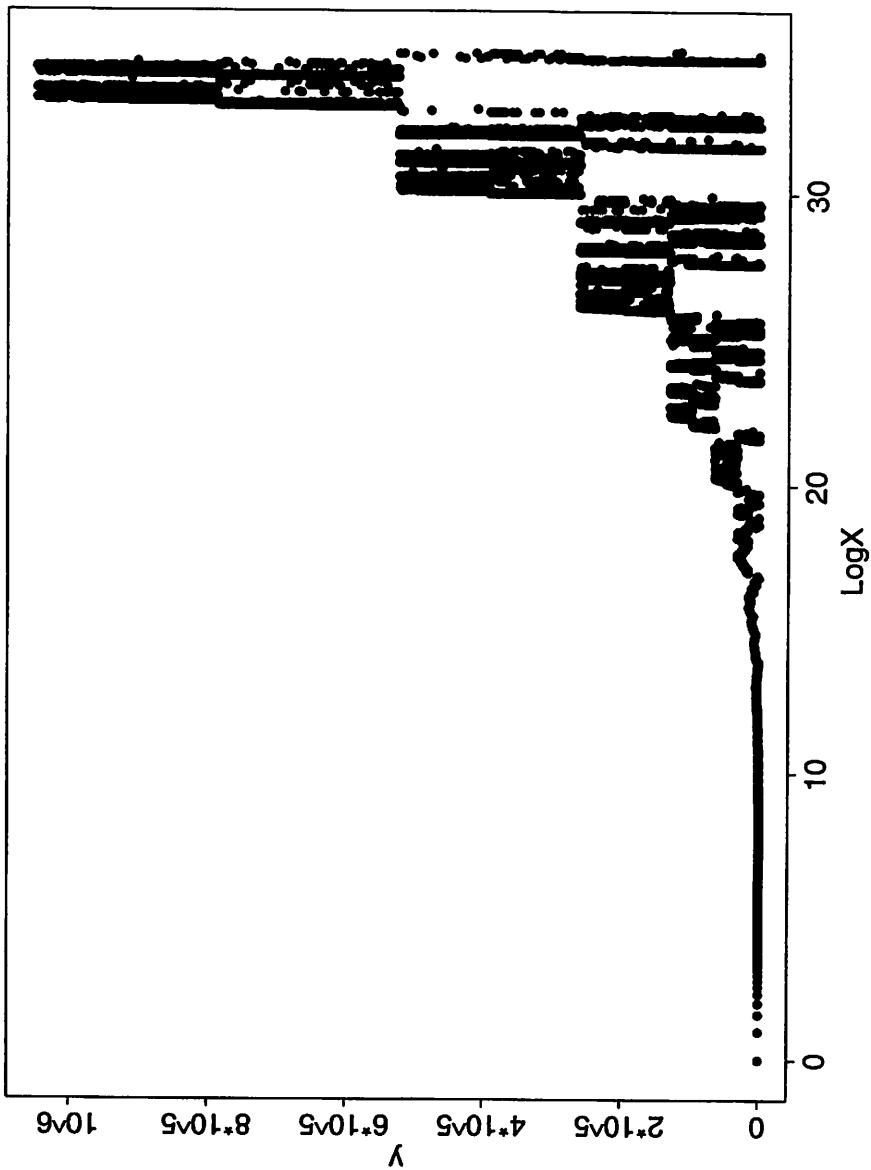

Figure 3(a): syndrome function for binary 4-error

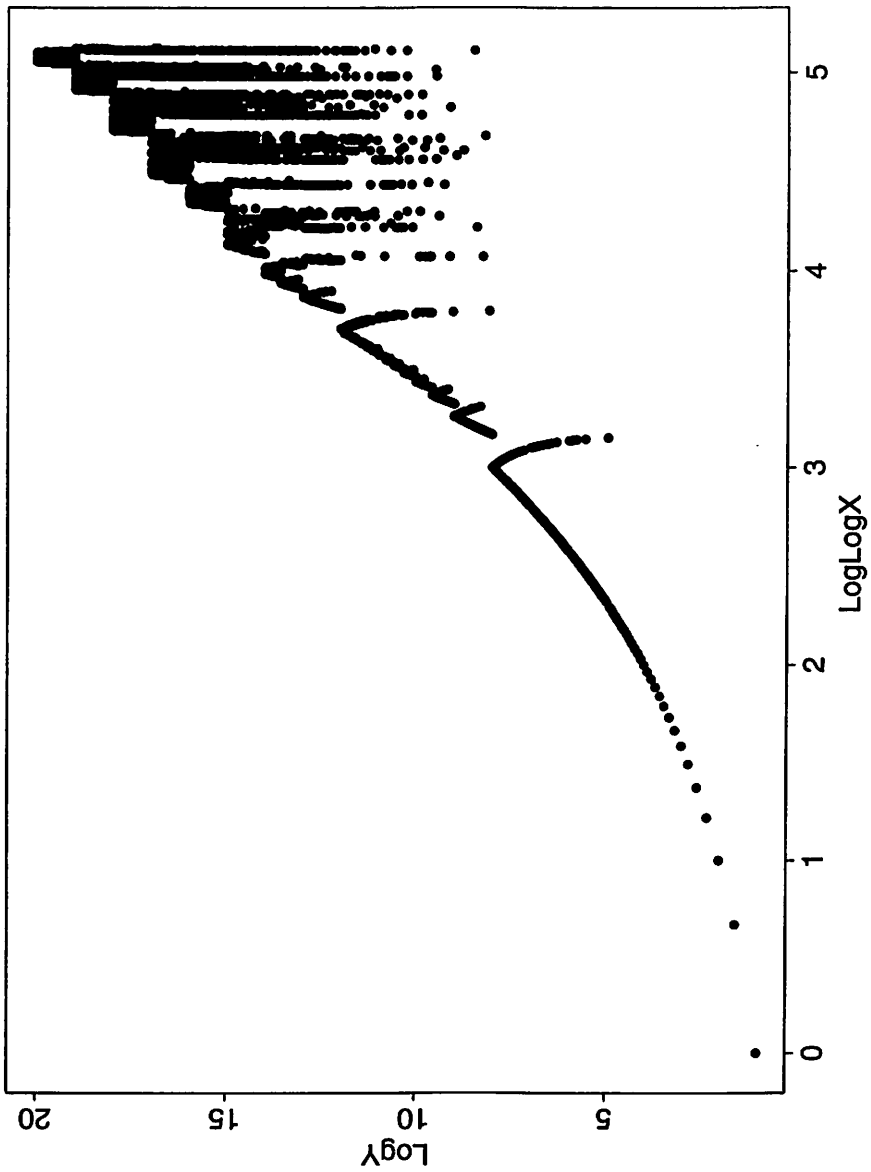

Figure 3(b): syndrome function for binary 4-error