

On High Rate Maximum Runlength Limited (0, k) Blocked Line Codes

P.G.W. van Rooyen and H.C. Ferreira

ABSTRACT

The essential parameters of runlength constrained codes are defined, together with a brief discussion of the properties of these codes. An overview of previous work done on (0, k) codes is also presented. A new algorithm is introduced for the enumeration of (0, k) block codes. This algorithm is further compared in terms of complexity and speed to previously published algorithms. A new high rate code is presented and compared to previously published codes.

KEYWORDS

Information theory, codes and coding, binary line codes, (0, k) runlength constraints.

1 INTRODUCTION

Pioneered by investigators like Freiman and Wyner [1], Kautz [2], Gabor [3], Tang and Bahl [4] and notably Fransaszek [5], runlength limited (RLL) binary codes form a sub class of the classes of codes referred to as constrained codes, modulation codes, line codes or dk codes. Of special interest here, is the class of binary dk codes. A dk -limited binary sequence, in short, a (d, k) sequence, satisfies simultaneously the following two conditions:

- 1) d constraint: two logical "ones" are separated by a run of consecutive "zeros" of length at least d .
- 2) k constraint: any run of consecutive "zeros" is of length at most k .

In general, a (d, k) -sequence is implemented using a simple precoding step. A (d, k) -sequence is converted to a runlength limited channel sequence in the following way. Let the channel signals be represented by a sequence $\{y_i\}$, $y_i \in \{-1, 1\}$. The logical "ones" in the (d, k) -sequence indicate the positions of a transition $1 \rightarrow -1$ or $-1 \rightarrow 1$ of the corresponding RLL sequence. The mapping of the waveform by the precoding step is known as *non-return-to-zero-inversion* (NRZI), or *change of state* encoding.

Consider the following binary dk sequence at the output of the encoder:

0 1 0 0 0 1 0 0 1 0 0 0 1 0 1 0 1 ...

The RLL channel sequence thus obtained after the precoding step is:

1 -1 -1 -1 -1 1 1 1 -1 -1 -1 -1 1 1 -1 -1 1 ...

In essence the RLL codes are thus characterized by two parameters, $(d+1)$ and $(k+1)$ which stipulate, respectively, the minimum and maximum runlength of the same channel symbol that may occur in the sequence. The parameter d controls the highest transition frequency, and thus has a bearing on the intersymbol interference when the sequence is transmitted over a bandwidth-limited channel. In the transmission of binary data it is generally desirable that the received signal is self-synchronizing or self-clocking. The parameter k ensures adequate frequency of transitions for clock extraction and bit synchronization of the received data.

The ratio m/n of input word length m to code word length n is called the *code rate*, designated by R . Note that for binary constrained codes $m/n < 1$. There is also a maximum achievable rate, called the *Shannon capacity*, designated by C [6]. Note that for most implementations of constrained codes, $m/n < C$.

Another important parameter is the efficiency of a code, designated by $\eta = R/C$. Since $R = m/n$, with m and n positive integers, the efficiency is an indication of the codes ability to approach the theoretical channel capacity, which is in general a rational number.

This paper will consider a special class of constrained codes namely codes with $d = 0$. An attractive property of this class of (d, k) codes is that the channel capacities (C) of the codes asymptotically approach 1 bit of information per binary channel symbol, for $k \rightarrow \infty$. Further, they can be constructed with *practical* coding rates of $R = (n-1)/n$; for a relatively large n , clock extraction can thus be gained with a marginal loss in bandwidth. The bandwidth of constrained codes is inversely proportional to the code rate R ; the bandwidth is a measure of how rapidly the information-bearing portions of a signal can change. For constrained codes the required channel bandwidth usually extends to the first spectral null, which is inversely proportional to the code rate R . For instance, a rate $R = 1/2$ binary constrained code will need a normalized bandwidth of $1/R = 2$. Constrained codes with a high code rate is thus preferred in communication systems which are bandwidth limited.

In general $(0, k)$ codes are used in a variety of applications, i.e. magnetic storage [7], optical channels [10] and mobile communication channels [11] to facilitate clock extraction for bit synchronization of the received data.

2 PREVIOUS WORK ON $(0, k)$ CODES

At the moment two $(0, k)$ codes are widely used, the Group-Coded Recording (GCR) code, with rate $R = 4/5$, $(d, k) = (0, 2)$ [7] and the rate $R = 8/9$, $(d, k) = (0, 3)$ code [8]. Both these codes are used in a large variety of magnetic tape products. In the GCR code, 4 user bits are uniquely represented by 5 channel bits, while 8 user bits are mapped on 9 channel bits for the $(0, 3)$ code. The theoretical channel capacity [7] of a sequence with no runs of more than two "zeros" is $C(0, 2) \approx 0.879$ bits of information per binary channel symbol, while the theoretical channel capacity of a sequence with no more than three "zeros" is $C(0, 3) \approx 0.946$ bits/symbol. Hence, the efficiency of the aforementioned two codes are respectively $\eta = (4/5)/0.879 = 91\%$ and $\eta = (8/9)/0.946 = 94\%$.

As pointed out in [7], the dk constraints define a number of channel states; these states can be represented by a Markov model for a given dk constraint. Using previously published code synthesis procedures [5,7] the crucial problem for the creation of fixed-length or blocked (d, k) codes of minimum length, is to find a subset of states, referred to as *principal states*, from which originate a sufficient number of sequences of length n terminating at other principal states. The existence of a set of principal states can also be used to verify the existence of a code with a specified rate and codeword length. Franaszek developed a recursive search technique for determining the existence of a set of principal states through operations on the connection matrix of the Markov model [7].

In this paper we present an alternative algorithm, the TS algorithm, to search for valid candidate code words. The advantage of the TS algorithm over the method developed by Franaszek, is that a code book can be obtained while searching for the code's existence, with a negligible loss in speed. A timesaving with the TS algorithm for large n is also anticipated, since the Franaszek algorithm involves plenty of multiplications (raising matrices to higher powers).

A description of the TS algorithm is presented in the next section.

3 ENUMERATION OF $(0,k)$ BLOCK CODES

Consider the following representation of a candidate code word, consisting of n bits; starting from the least significant bit (LSB), number 0, and ending with the most significant bit (MSB), number $n-1$:

	$n-1$	$n-2$...	$k-1$...	0
Potential code word	0	1	...	x	...	x

The choice of the bits 0 and 1, respective, in positions $n-1$ and $n-2$ will become apparent shortly. With this representation in mind, the TS algorithm can be described step by step in the following way:

- a) When searching for valid code words, the first 2^{n-2} n -bit binary words (in normal counting or lexicographical order) are excluded, since we shall not include code words ending with more than one zero. In other words, if bit $n-1$ is zero (the MSB), bit $n-2$ is not allowed to be zero. This was done since experimentation showed that the maximum number of concatenable valid code words as reported in [7] can be obtained in the least time with this restriction;
- b) Next, bits 0 to $k-1$ of the remaining candidate code words are searched for the occurrence of k consecutive zeros which will violate the k constraint when bit $n-1$ of the previous word is zero. Simultaneously we search for zeros that violate the k constraint from bit $n-2$ to bit k ;
- c) Finally, the number of valid code words obtained in the previous search must be at least 2^{n-1} for the $(0, k)$ code to be of interest to us, since we shall confine our search to codes with a code rate of $R = (n-1)/n$.

We confirmed the validity of the TS algorithm by duplicating the known results for the GCR code and rate $R = 8/9$, $(d, k) = (0, 3)$ code. For the GCR code; from the 32 possible unconstrained combinations of 5 bits, 15 were eliminated, leaving 17 candidate $(0, 2)$ constrained code words. From the 512 possible unconstrained combinations of 9 bits, 219 can be eliminated leaving 293 candidate code words in the $(0, 3)$ code. These numbers, which can also be obtained from Immink [7], serves as confirmation of the accuracy and optimality of the TS algorithm, since exactly the same number of valid code words were obtained using the TS algorithm.

We firstly applied the TS algorithm to a constraint of $k = 4$. Figure 1 compares the time in seconds versus codeword length, n , for the TS algorithm and an exhaustive search for valid code words (i.e. exhaustively checking binary words 0 to 2^n-1 for a violation of the k constraint), with $k = 4$ and $14 \leq n \leq 20$. The graph shows a constant improvement in speed for a fixed value of k . The TS algorithm achieves more than twice the speed of the exhaustive search. The gain of the TS algorithm is directly proportional to n ; the larger n , the better the absolute value of the improvement on an exhaustive search.

From the 2^n possible unconstrained combinations of n bits, the valid code words in the lexicographically first 2^{n-1} unconstrained words, designated N_0 , were recorded, together with the valid code words in the lexicographically second 2^{n-1} unconstrained words, designated N_1 . Figure 2 shows these valid words (and the sum

$N_0 + N_1$) for $k = 4$ and different values of n . The reason why both N_0 and N_1 were recorded, is to confirm the anticipation of more valid code words in the second 2^{n-1} unconstrained words, N_1 . Figure 3 shows a comprehensive 3D graph with $N_0 + N_1$, $1 \leq k \leq 20$ and $2 \leq n \leq 20$.

Since the TS algorithm is so powerful and easy to implement on a digital computer, it was decided to do a complete search for the class of $(0, k)$ codes in the range $1 \leq k \leq 20$ and $2 \leq n \leq 20$ on an IBM AT compatible with a 16 MHz clock. Table 1 to 20 show the efficiencies and number of valid code words for $1 \leq k \leq 20$ and $2 \leq n \leq 20$. Note that encoders and decoders for codes with $n = 20$ can be implemented with currently available 1 Mbit ROM's. Following [4], the capacity of the codes is indicated at the bottom of each table in the form of

$$C(d, k) = \log_2 \lambda_{\max} \quad (1)$$

with λ_{\max} the largest, positive eigenvalue of

$$\sum_{i=d}^k z^{(n-i-1)} = 0 \quad (2)$$

which is the characteristic equation for RLL sequences [7].

For some values of n , Tables 2 to 20 are truncated. There may be two possible reasons for this; firstly, the search may have yielded fewer than 2^{n-1} valid code words for large n when $(n-1)/n > C$, and secondly, since these are block codes, the k constraint may not have been accomplished with short block lengths. This happens when $k > n$. Only Table 1 is complete, all entries are indicated, including the entries representing fewer than 2^n codewords. Tables 2 to 20 are shortened, showing only entries for $R = (n-1)/n$ codes, to save space.

4 A NEW HIGH RATE CODE

As mentioned earlier a rate $R = 8/9$, $(d, k) = (0, 3)$ code was developed by Patel [8]. Consulting Table 3, it is evident that this code is not of optimal efficiency. It was thus decided to improve on this code by developing a maximal rate $R = 11/12$, $(d, k) = (0, 3)$ code. The code is said to be maximal since $n-1 = 11$ and $n = 12$ are the largest possible integers to choose such that $(n-1)/n \leq C(0, 3)$ (Table 3).

The code book for the encoder and decoder are too large to include in this paper, since the encoder table consist of $2^{11} = 2048$ words and the decoder table consist of $2^{12} = 4096$ words. However, reference can be made to [9] for the encoder and decoder tables.

Figure 4 depicts the normalized spectra of the well known rate $R = 1/2$, $(d, k) = (1, 3)$ Miller code, the rate $R = 8/9$, $(d, k) = (0, 3)$ Patel code, the newly developed rate $R = 11/12$, $(d, k) = (0, 3)$

code and a PN-sequence of length $2^9 - 1 = 511$ bits. All these spectra were measured at a data rate of 1200 bps.

This figure confirms the previous statement that RLL codes' bandwidth are inversely proportional to the code rate R ; the first spectral null is encountered at $1/R$. The horizontal axis depicts the frequency in normalized form. Normalization is relative to the PN-sequence bandwidth, i.e. the normalized bandwidth of a PN-sequence at a specific baud rate will be 1, where the normalized bandwidth of a RLL sequence with rate $R = 1/2$ will thus be 2. The PN-sequence thus has a normalized bandwidth of one, which is proportional to the baud rate, while the rate $R = 1/2$ Miller code has a normalized bandwidth of two, the rate $R = 8/9$ Patel code has a normalized bandwidth of $9/8$ and the newly developed rate $R = 11/12$ code has a normalized bandwidth of $12/11$. It is thus evident that the higher the code rate the more efficient the code becomes in terms of bandwidth efficiency; a property of utmost importance in some digital communication systems.

The newly developed rate $R = 11/12$ code's encoder can easily be implemented by a $2^{11} \times 12$ bit = 2 kbit \times 12 bit ROM look up table, while the decoder can be realized with a $2^{12} \times 12$ bit = 4 kbit \times 12 bit ROM look up table.

5 CONCLUSIONS

Maximum runlength constrained codes facilitate clock extraction and bit synchronization on many digital communication channels. Codes with a constraint of $(0, k)$ and a rate of $R = (n-1)/n$ approaches the channel capacity C for appropriate choice of n . These codes have very high efficiencies and can further be used on channels with a severely limited bandwidth, such as mobile communication channels. At present there are not many known codes that have these parameters. Since $(0, k)$ codes can be realized with a high code rate of $R = (n-1)/n$ and find application on diverse channels such as magnetic, optical and mobile communication channels, it is important to have an easy implementable algorithm to search for the existence of these codes. Against this backdrop the new rate $R = 11/12$, $(d, k) = (0, 3)$ code is of practical interest, with the TS algorithm and Tables 1 to 20 making it easier in future to develop codes with the desired properties.

6 REFERENCES

- [1] C.V. Freiman and A.D. Wyner, "Optimal block codes for noiseless input restricted channels", Information and Control, vol. 7, 1964, pp 398-415.
- [2] W.H. Kautz, "Fibonacci codes for synchronization control", IEEE Transactions on Information Theory, vol. IT-11, 1965, pp 284-292.
- [3] A. Gabor, "Adaptive coding for self-clocking recording", IEEE Transactions on Electronic Computers, vol. EC-16, 1967, pp 866-868.

- [4] D.T. Tang and L.R. Bahl, "Block codes for a class of constrained noiseless channels", Information and Control, vol. 17, 1970, pp 436-461.
- [5] P.A. Franaszek, "Sequence-state encoding for digital transmission", Bell Systems Technical Journal, vol 47, January 1968, pp 143-157.
- [6] C.E. Shannon, "A mathematical theory of communication", Bell Systems Technical Journal, vol 27, July 1948, pp 379-423.
- [7] K.A. Schouhammer Immink, "Coding techniques for optical and magnetic recording channels", Prentice Hall Inc, New York, 1990.
- [8] A.M. Patel, "Improved encoder and decoder for a byte-oriented rate 8/9 (0, 3) code", IBM Technical Disclosure Bulletin, vol. 28, 1985, pp 1938.
- [9] P.G.W. van Rooyen, "Modulation codes for mobile communications", M.Eng thesis, Rand Afrikaans University, June 1991.
- [10] N. Yoshikai, K. Katagiri and T Ito, "mB1C code and its performance in an optical communication system", IEEE Transactions on Communications, vol. COM-32, no. 2, February 1984, pp 163-168.
- [11] CCIR: IRCC. "Recommendations and reports of the CCIR, 1981", report 780-1.

AUTHORS BIOGRAPHY

P. van Rooyen received his B.Eng (Elec) and M.Eng (Elec) degrees from the Rand Afrikaans University, in 1989 and 1991 respectively. He is currently working towards his D.Eng degree at the same university.

H.C. Ferreira received the B.Sc. degree in electrical engineering, the M.Sc. degree in electronic engineering and the D.Sc. (Eng) degrees from the University of Pretoria, in 1976, 1978 and 1980 respectively. He is currently Professor and chairman of the Laboratory for Cybernetics, Faculty of Engineering, Rand Afrikaans University.

AUTHORS AFFILIATION

P. van Rooyen, B.Eng (Elec), M.Eng (Elec) and
 Prof. H.C. Ferreira, Pr. Eng., D.Sc. (Eng), MSAIEE, MIEEE
 Cybernetics Laboratory
 Rand Afrikaans University
 P.O. Box 524
 Johannesburg
 2000 South Africa

n	η	N_0	N_1	N_0+N_1
2	0.720	1	1	2
3	0.961	1	2	3
4	-	2	3	5
5	-	3	5	8
6	-	5	8	13
7	-	8	13	21
8	-	13	21	34
9	-	21	34	55
10	-	34	55	89
11	-	55	89	144
12	-	89	144	233
13	-	144	233	377
14	-	233	377	610
15	-	377	610	987
16	-	610	987	1597
17	-	987	1597	2584
18	-	1597	2584	4181
19	-	2584	4181	6765
20	-	4181	6765	10946

TABLE 1 - $k = 1, C(0, 1) = 0.69424$

n	η	N_0	N_1	N_0+N_1
3	0.704	2	4	6
4	0.792	4	7	11
5	0.844	7	14	21
6	0.880	14	27	41
7	0.905	27	52	79
8	0.924	52	100	152
9	0.938	100	193	293
10	0.950	193	372	565
11	0.960	372	717	1089
12	0.968	717	1382	2099
13	-	1382	2664	4046

TABLE 3 - $k = 3, C(0, 3) = 0.946777$

n	η	N_0	N_1	N_0+N_1
2	0.569	1	2	3
3	0.758	2	3	5
4	0.853	3	6	9
5	0.910	6	11	17

TABLE 2 - $k = 2, C(0, 2) = 0.879146$

n	η	N_0	N_1	N_0+N_1
4	0.769	4	8	12
5	0.820	8	15	23
6	0.854	15	30	45
7	0.878	30	59	89
8	0.897	59	116	175
9	0.911	116	228	344
10	0.922	228	448	676
11	0.932	448	881	1329
12	0.939	881	1732	2613
13	0.946	1732	3405	5137
14	0.952	3405	6694	10099
15	0.957	6694	13160	19854
16	0.961	13160	25872	39032
17	0.965	25872	50863	76735
18	0.968	50863	99994	150857
19	0.971	99994	196583	296577
20	0.974	196583	386472	583055

TABLE 4 - $k = 4, C(0, 4) = 0.975225$

n	η	N_0	N_1	N_0+N_1
5	0.809	8	16	24
6	0.843	16	31	47
7	0.867	31	62	93
8	0.885	62	123	185
9	0.899	123	244	367
10	0.910	244	484	728
11	0.920	484	960	1444
12	0.927	960	1904	2864
13	0.934	1904	3777	5681
14	0.939	3777	7492	11269
15	0.944	7492	14861	22353
16	0.948	14861	29478	44339
17	0.952	29478	58472	87950
18	0.955	58472	115984	174456
19	0.958	115984	230064	346048
20	0.961	230064	456351	686415

TABLE 5 - $k = 5, C(0, 5) = 0.988109$

n	η	N_0	N_1	N_0+N_1
6	0.838	16	32	48
7	0.862	32	63	95
8	0.880	63	126	189
9	0.894	126	251	377
10	0.905	251	500	751
11	0.914	500	996	1496
12	0.922	996	1984	2980
13	0.928	1984	3952	5936
14	0.933	3952	7872	11824
15	0.938	7872	15681	23553
16	0.942	15681	31236	46917
17	0.946	31236	62221	93457
18	0.949	62221	123942	186163
19	0.952	123942	246888	370830
20	0.955	246888	491792	738680

TABLE 6 - $k = 6, C(0, 6) = 0.994192$

n	η	N_0	N_1	N_0+N_1
7	0.859	32	64	96
8	0.877	64	127	191
9	0.891	127	254	381
10	0.902	254	507	761
11	0.911	507	1012	1519
12	0.919	1012	2020	3032
13	0.925	2020	4032	6052
14	0.931	4032	8048	12080
15	0.936	8048	16064	24112
16	0.940	16064	32064	48128
17	0.943	32064	64001	96065
18	0.947	64001	127748	191749
19	0.950	127748	254989	382737
20	0.952	254989	508966	763955

TABLE 7 - $k = 7, C(0, 7) = 0.997134$

n	η	N_0	N_1	N_0+N_1
8	0.876	64	128	192
9	0.890	128	255	383
10	0.901	255	510	765
11	0.910	510	1019	1529
12	0.917	1019	2036	3055
13	0.924	2036	4068	6104
14	0.929	4068	8128	12196
15	0.934	8128	16240	24368
16	0.938	16240	32448	48688
17	0.942	32448	64832	97280
18	0.945	64832	129536	194368
19	0.948	129536	258817	388353
20	0.951	258817	517124	775941

TABLE 8 - $k = 8, C(0, 8) = 0.998578$

n	η	N_0	N_1	N_0+N_1
9	0.889	128	256	384
10	0.900	256	511	767
11	0.909	511	1022	1533
12	0.917	1022	2043	3065
13	0.923	2043	4084	6127
14	0.929	4084	8164	12248
15	0.933	8164	16320	24484
16	0.938	16320	32624	48944
17	0.941	32624	65216	97840
18	0.945	65216	130368	195584
19	0.948	130368	260608	390976
20	0.950	260608	520960	781568

TABLE 9 - $k = 9, C(0, 9) = 0.999292$

n	η	N_0	N_1	N_0+N_1
10	0.900	256	512	768
11	0.909	512	1023	1535
12	0.916	1023	2046	3069
13	0.923	2046	4091	6137
14	0.928	4091	8180	12271
15	0.933	8180	16356	24536
16	0.937	16356	32704	49060
17	0.941	32704	65392	98096
18	0.944	65392	130752	196144
19	0.947	130752	261440	392192
20	0.950	261440	522752	784192

TABLE 10 - $k = 10, C(0, 10) = 0.999647$

n	η	N_0	N_1	N_0+N_1
11	0.909	512	1024	1536
12	0.916	1024	2047	3071
13	0.923	2047	4094	6141
14	0.928	4094	8187	12281
15	0.933	8187	16372	24559
16	0.937	16372	32740	49112
17	0.941	32740	65472	98212
18	0.944	65472	130928	196400
19	0.947	130928	261824	392752
20	0.950	261824	523584	785408

TABLE 11 - $k = 11, C(0, 11) = 0.999824$

n	η	N_0	N_1	N_0+N_1
12	0.916	1024	2048	3072
13	0.923	2048	4095	6143
14	0.928	4095	8190	12285
15	0.933	8190	16379	24569
16	0.937	16379	32756	49135
17	0.941	32756	65508	98264
18	0.944	65508	131008	196516
19	0.947	131008	262000	393008
20	0.950	262000	523968	785968

TABLE 12 - $k = 12, C(0, 12) = 0.999912$

n	η	N_0	N_1	N_0+N_1
13	0.923	2048	4096	6144
14	0.928	4096	8191	12287
15	0.933	8191	16382	24573
16	0.937	16382	32763	49145
17	0.941	32763	65524	98287
18	0.944	65524	131044	196568
19	0.947	131044	262080	393124
20	0.950	262080	524144	786224

TABLE 13 - $k = 13$, $C(0, 13) = 0.999956$

n	η	N_0	N_1	N_0+N_1
15	0.933	8192	16384	24576
16	0.937	16384	32767	49151
17	0.941	32767	65534	98301
18	0.944	65534	131067	196601
19	0.947	131067	262132	393199
20	0.950	262132	524260	786392

TABLE 15 - $k = 15$, $C(0, 15) = 0.999989$

n	η	N_0	N_1	N_0+N_1
17	0.941	32768	65536	98304
18	0.944	65536	131071	196607
19	0.947	131071	262142	393213
20	0.950	262142	524283	786425

TABLE 17 - $k = 17$, $C(0, 17) = 0.999997$

n	η	N_0	N_1	N_0+N_1
19	0.947	131072	262144	393216
20	0.950	262144	524287	786431

TABLE 19 - $k = 19$, $C(0, 19) = 0.999999$

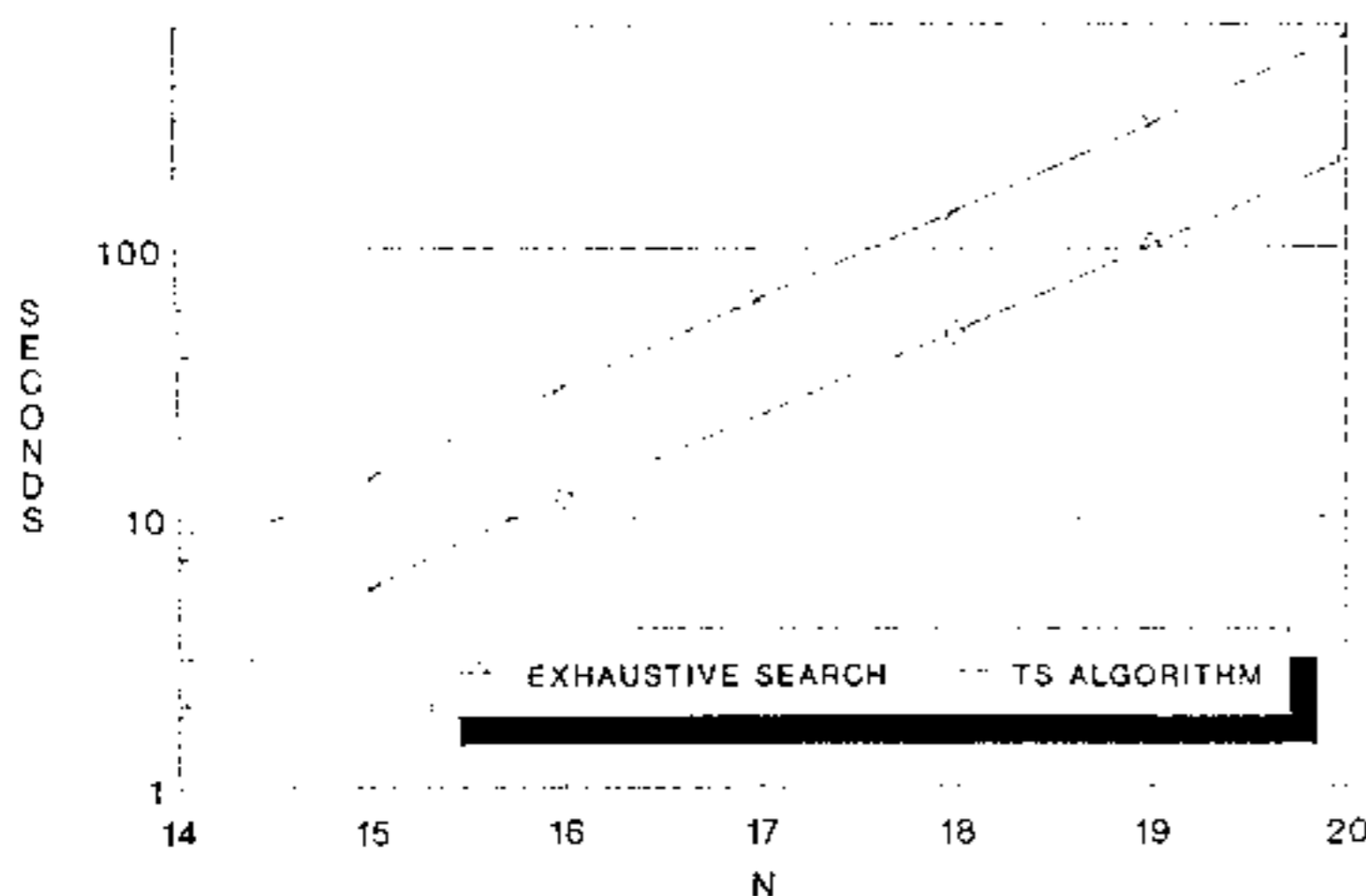


Figure 1

Time comparison of exhaustive search and TS algorithm for $k = 4$

n	η	N_0	N_1	N_0+N_1
14	0.928	4096	8192	12288
15	0.933	8192	16383	24575
16	0.937	16383	32766	49149
17	0.941	32766	65531	98297
18	0.944	65531	131060	196591
19	0.947	131060	262116	393176
20	0.950	262116	524224	786340

TABLE 14 - $k = 14$, $C(0, 14) = 0.999978$

n	η	N_0	N_1	N_0+N_1
16	0.937	16384	32768	49152
17	0.941	32768	65535	98303
18	0.944	65535	131070	196605
19	0.947	131070	262139	393209
20	0.950	262139	524276	786415

TABLE 16 - $k = 16$, $C(0, 16) = 0.999994$

n	η	N_0	N_1	N_0+N_1
18	0.944	65536	131072	196608
19	0.947	131072	262143	393215
20	0.950	262143	524286	786429

TABLE 18 - $k = 18$, $C(0, 18) = 0.999999$

n	η	N_0	N_1	N_0+N_1
20	0.950	262144	524288	786432

TABLE 20 - $k = 20$, $C(0, 20) = 0.999999$

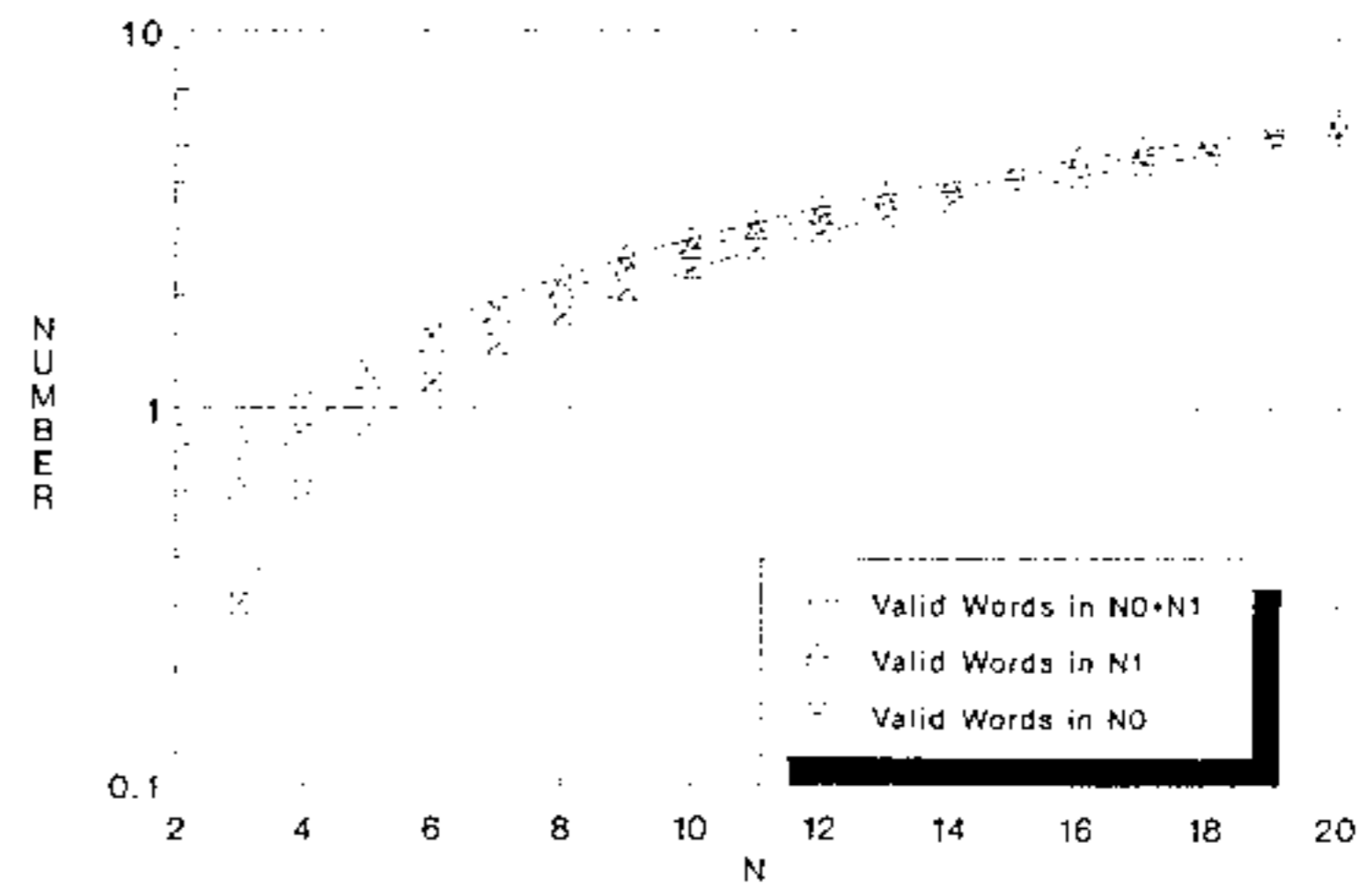


Figure 2

Valid codewords for $k = 4$, $2 \leq n \leq 20$

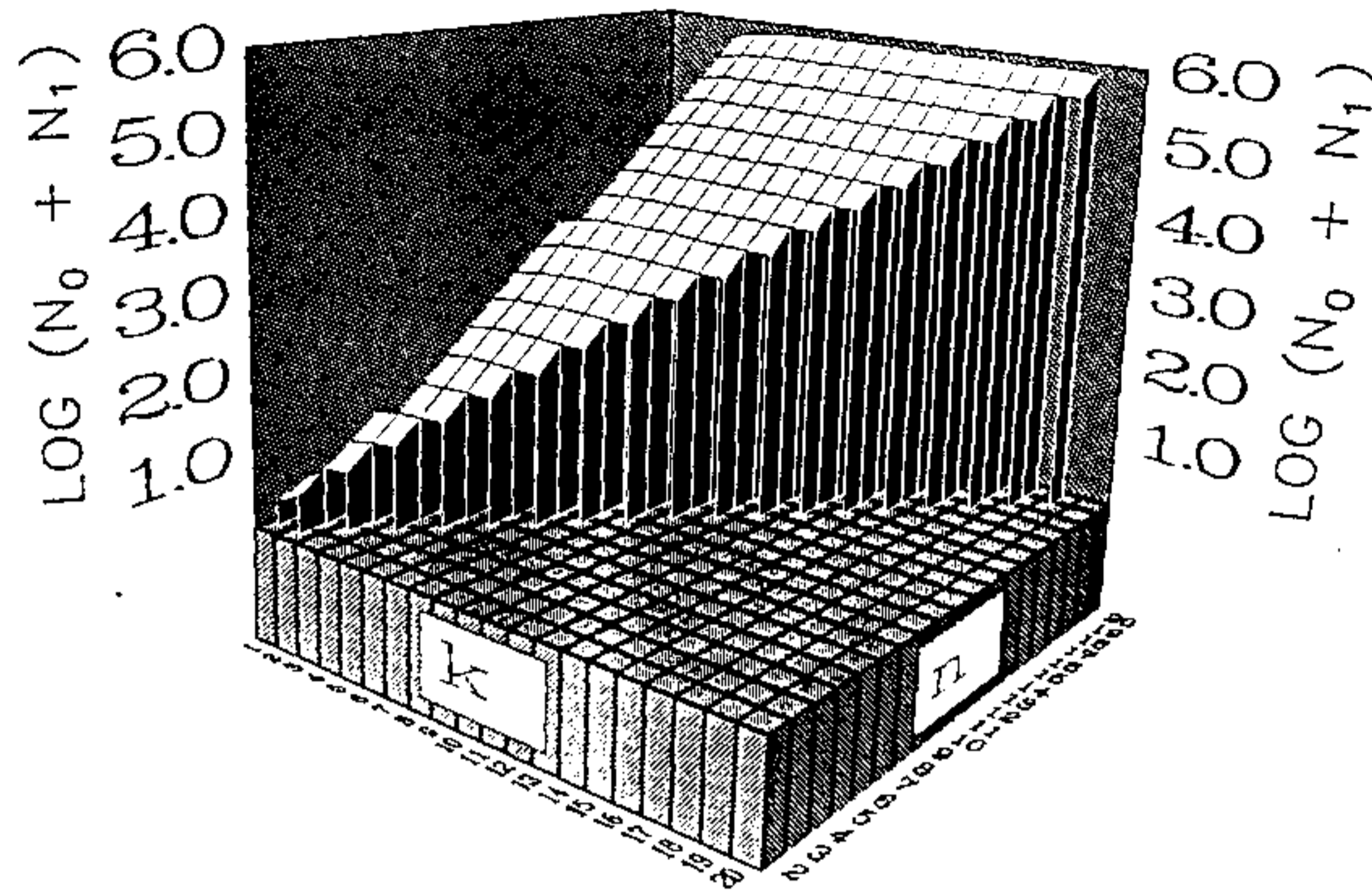


Figure 3

Valid codewords obtained with TS algorithm for $1 \leq k \leq 20, 2 \leq n \leq 20$

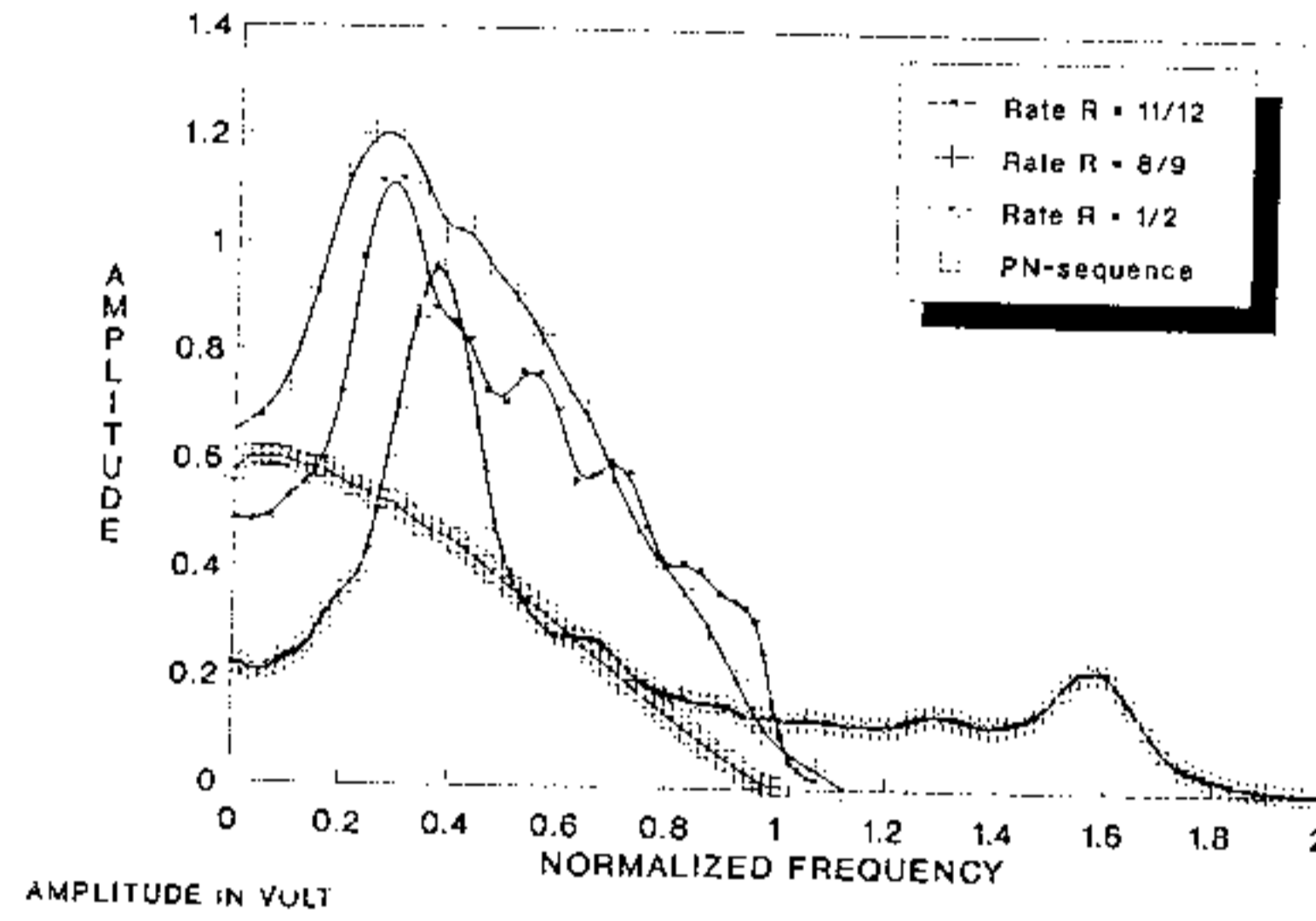


Figure 4

Spectra comparison of PN-sequence to RLL codes