Evaluation of Gallager Codes for Short Block Length and High Rate Applications

DAVID J.C. MACKAY   MATTHEW C. DAVEY

To appear in proceedings of IMA workshop on Codes, Systems and Graphical Models 1999

Abstract

Gallager codes with large block length and low rate (*e.g.*, $N \simeq 10,000$–$40,000$, $R \simeq 0.25$–0.5) have been shown to have record-breaking performance for low signal-to-noise applications. In this paper we study Gallager codes at the other end of the spectrum. We first explore the theoretical properties of binary Gallager codes with very high rates and observe that Gallager codes of any rate offer runlength-limiting properties at no additional cost.

We then report the empirical performance of high rate binary and non-binary Gallager codes on three channels: the binary input Gaussian channel, the binary symmetric channel, and the 16-ary symmetric channel.

We find that Gallager codes with rate $R = 8/9$ and block length $N = 1998$ bits outperform comparable BCH and Reed-Solomon codes (decoded by a hard input decoder) by more than a decibel on the Gaussian channel.

Keywords: Error-correcting codes, Sum-product algorithm, Magnetic recording.

1 Introduction

1.1 Definition of Gallager codes

A regular Gallager code (Gallager, 1962) has a parity check matrix with uniform column weight j and uniform row weight k, both of which are very small compared to the blocklength. If the code has transmitted blocklength N and rate R then the parity check matrix $\mathbf{H}$ has N columns and M rows, where $M \geq N(1-R)$. [Normally parity check matrices have $M = N(1-R)$, but the matrices we construct may have a few redundant rows so that their rate could be a little higher than $1 - M/N$.]

In this paper we explore whether Gallager codes are useful for high rates ($R > 2/3$) and small block lengths ($N < 5000$).

1.2 High-rate codes

Reed-Solomon codes are the industry standard error-correcting codes for high rate, low block length applications such as magnetic disc drives and compact discs. They have good distance properties and they have an efficient bounded-distance decoder.

When we proposed evaluating Gallager codes with high rate and small block length for disc drive applications, a common response was 'why bother? You'll never beat Reed-Solomon codes.' But there are several reasons for checking the performance of Gallager codes.

1. Gallager codes with large block length N have good distance properties, with high probability (Gallager, 1963; MacKay, 1999c). Given an optimal decoder, Gallager codes can get arbitrarily close to the Shannon limit of a wide variety of channels (MacKay, 1999c).

2. There is a practical sum-product decoder for Gallager codes which works well for codes with block lengths of order $N = 10,000$ and rates of order $R = 1/4$–$2/3$ (MacKay and Neal, 1996).

   At rates of $R = 1/2$ and $R = 1/4$, regular binary Gallager codes decoded using this algorithm have near-Shannon limit performance. Irregular binary and non-binary Gallager codes with these rates perform better on the binary Gaussian channel than all known practical codes, including turbo codes (Davey and MacKay, 1998a; Urbanke *et al.*, 1999).

   This decoder has three important features:

   (a) It is better than a bounded-distance decoder — it works well at noise levels significantly larger than the Gilbert noise level (that is, the noise level at which typical error events have weight greater than half the minimum distance of a code at the Gilbert bound).

   (b) It is a soft-input decoder, able to make use of likelihood information from the channel output. Such decoders can have considerable advantages over decoders that take hard inputs (Gallager, 1963).

   (c) The decoder can be generalized to infer bursts if the channel is believed to be a bursty channel (Worthen and Stark, 1998).

3. According to Berlekamp (1968), one reason that high rate Reed-Solomon codes are used is that lower rate Reed-Solomon codes are more costly to encode and decode — the complexity increases with increasing redundancy.

   In contrast, the encoding and decoding complexity for Gallager codes hardly depend on rate. Furthermore, Gallager codes of any desired rate and block length can easily be constructed.

If Reed-Solomon codes can be surpassed, the disc drive industry could benefit in various ways. A higher rate code with the same probability of error would allow a small increase in the storage capacity of a drive. Alternatively, a code that can cope with larger raw error rates would make the system more tolerant to tracking errors, and the disc could be spun faster, offering a higher data rate.

1.3 Outline of paper

In section 2, we explore four theoretical issues. First, we ask how good is the ensemble of random, high-rate, regular Gallager codes, if we do not constrain the overlap between the columns? We find that the expected distance properties of these codes are not good. Second, we ask what are the highest possible rates that regular Gallager codes could have if we do constrain the overlap between columns — these codes correspond to 'Steiner systems' — and could these codes have good distance properties? We prove that such codes, with $j = 3$, have bad distance, and we give a conjecture that, for larger j, the codes might be good. Third, we prove that a particular construction of Gallager codes in terms of permutation matrices leads to bad codes. Fourth, we show that Gallager codes can be constructed to have fortuitous runlength-limiting properties.

In section 3, we describe the empirical performance of high-rate binary regular Gallager codes with $j = 4$ and of a high-rate non-binary regular Gallager code with $j = 3$. We compare these Gallager codes with codes similar to those used in discdrives and show that their performance is good.

In section 4, we discuss difference-set cyclic codes, which are codes similar to Gallager codes, but having the special property that they satisfy many more than M low-weight parity constraints. They outperform equivalent Gallager codes by a significant margin (Tanner, 1981; Lucas *et al.*, 1999). If we could find more codes like these, they could be very useful.

2 Theory of high rate Gallager codes

2.1 Distance properties of random Gallager codes

The expectation of the weight enumerator function of a random Gallager code with $M \times N$ parity check matrix can be computed for two ensembles.

Ensemble G: In Gallager's (1963) ensemble, a row weight k is selected, and a blocklength N. We find the weight enumerator function $A(w; 1)$ of the following submatrix with column weight 1 (illustrated for the case $k = 4$):

$$\mathbf{H}^{(1)} = \begin{bmatrix} 1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\ldots \end{bmatrix}. \tag{1}$$

As shown by Gallager (1963), $A(w;1)$ is given by convolving ($\star$) together N/k copies of the function

$$a(w) = \begin{cases} \binom{k}{w} & w \text{ even} \\ 0 & w \text{ odd} \end{cases} : \tag{2}$$

$$A(w;1) = a(w) \star a(w) \star \ldots \star a(w). \tag{3}$$

We define an ensemble of random Gallager codes with column weight j by stacking j copies of $\mathbf{H}^{(1)}$ vertically above each other, each individual copy having its columns randomly permuted. We can then find the expected weight enumerator function $A(w;j)$ of the resulting (N, M, j, k) code with $M = \frac{j}{k}N$ using:

$$\langle A(w;j) \rangle = A_{\mathrm{G}}(w;j) \equiv A(w;1) \left[\frac{A(w;1)}{\binom{N}{w}} \right]^{j-1}. \tag{4}$$

Ensemble M: An alternative ensemble of matrices that have column weight at most j, and arbitrary M and N, but do not have fixed row weight k, was used by MacKay (1999c). Each column of the matrix $\mathbf{H}$ is created by flipping j not-necessarily-distinct entries. With high probability, any particular column has weight j, but it may, with smaller probability, have weight $j-2$, etc. The expected weight enumerator function is

$$\langle A(w;j) \rangle = A_{\mathrm{M}}(w;j) \equiv \binom{N}{w} p_{00}^{(wj)} \tag{5}$$

where

$$p_{00}^{(r)} = 2^{-M} \sum_{i=0}^{M} \binom{M}{i} \left(1 - \frac{2i}{M}\right)^{r}. \tag{6}$$

These ensembles are not the best ensembles for making good Gallager codes, but they are convenient for estimating weight enumerator functions and getting a feel for the dependence on block length and rate. Figure 1 shows the expected weight enumerator functions for a sequence of codes with block length 540 bits and rate increasing from 1/3 to 8/9.

It seems that for small block lengths and large rates such as $R = 8/9$, codes constructed by this random construction will almost certainly be bad codes, in that their distance will be nowhere near the Gilbert distance.

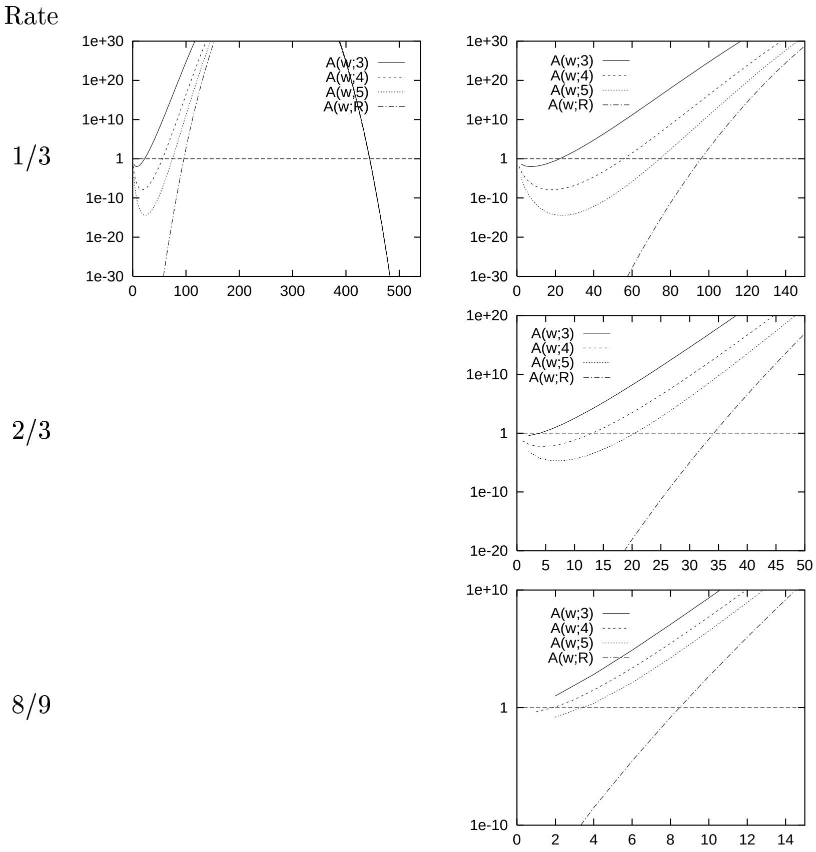


Figure 1: Expected weight enumerator functions computed using ensemble M. [The results for ensemble G are similar.] Block length is 540 bits in all cases. Top figure shows the expected weight enumerator function for codes with rate 1/3 having $j = 3$, 4 and 5. The lowest line shows the expected weight enumerator function of a random linear code with the same N and M. This line crosses the horizontal line $A(w) = 1$ at the Gilbert distance. The neighbouring figure shows detail from the first figure. Subsequent figures show the corresponding graphs for rates 2/3 and 8/9. It is evident that the typical distance of a Gallager code is becoming an increasingly small fraction of the Gilbert distance as the rate increases.

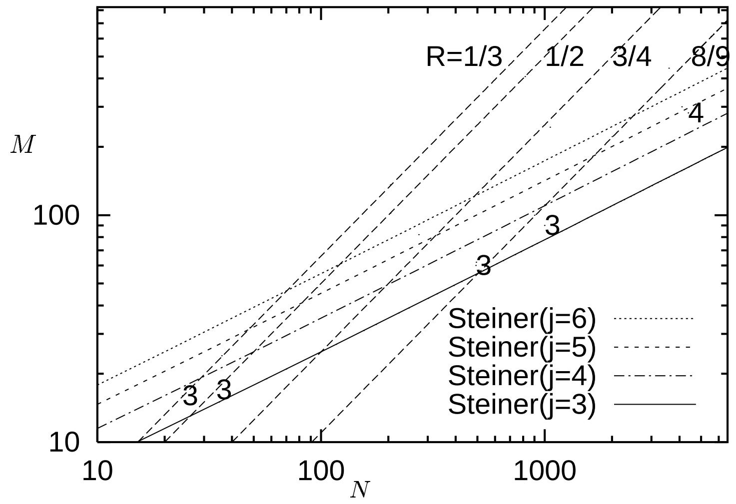


Figure 2: Parameters (N, M, j) that can be built without violating the constraint on column overlaps. Horizontal axis: N; vertical axis: M; the labels '3' and '4' show examples of parameters that have been built by random construction methods, including codes presented in this paper. The 45-degree lines are lines of constant rate $R = (N - M)/N$. The near-horizontal lines show the curves (N, M) defined by Steiner systems for various j (equation (8)). Points (N, M, j) below the Steiner curve $S(j)$ are not buildable. For low rates such as $1/2$ or $1/3$, very small blocklengths are buildable but as the rate is increased, the smallest possible blocklength becomes quite large.

We therefore use constrained random constructions. The constraint used by Gallager (1963) and MacKay and Neal (1996), which we also use here, constrains the maximum overlap between any two columns in the matrix to be one. We will call this the overlap constraint.

2.2 Steiner systems and Gallager codes

2.2.1 Existence of high rate codes.

If we insist on the constraint that the overlap between any two columns in the parity check matrix should be at most one, then it is not possible to build Gallager codes with arbitrary values of (N, M, j); in particular, we cannot make the blocklength N arbitrarily large for fixed number of rows M. The blocklength N of such a code with column weight j and M rows is bounded above by the size of a Steiner system $S(M, j, 2)$.

A Steiner system $S(M, j, t)$ is a set $\mathcal{M}$ of M points, and a collection $\mathcal{N}$

of subsets of $\mathcal{M}$ of size j, called blocks, such that any subset of t points of $\mathcal{M}$ are in exactly one of the blocks. The size of the Steiner system, N, is defined to be the number of blocks $N = |\mathcal{N}|$. The special case $j = 3$, $t = 2$ is called a Steiner triple system.

The number of subsets of size t in $\mathcal{M}$ is $\binom{M}{t}$ and the number of subsets of size t in a block is $\binom{j}{t}$, so the size of an (M, j, t) Steiner system is

$$N_S(M, j, t) = \binom{M}{t} \Big/ \binom{j}{t} \tag{7}$$

In the case of interest, $t = 2$, we obtain:

$$N_S(M, j) = \frac{M(M-1)}{j(j-1)}. \tag{8}$$

The row weight of $\mathbf{H}$, k, is

$$k = j\frac{N_S(M, j)}{M} = \frac{(M-1)}{(j-1)} \tag{9}$$

Any $(M, j, 2)$ Steiner system defines an (N, M, j) Gallager code with $N = N_S(M, j)$. If N exceeds $N_S(M, j)$, it is impossible to make a regular Gallager code with parameters N, M, j that satisfies the overlap constraint. So for any chosen M and j the overlap constraint implies a maximum possible rate, and for any rate R and column weight j, it implies a minimum possible blocklength.

These constraints are illustrated in figure 2, which shows $N_S(M, j)$ as a function of M for various values of j. This figure also shows some actual values of (N, M, j) that have been constructed by the random constructions mentioned above. Fortunately, codes with column weight $t = 4$, blocklength $N \simeq 2000$ and rate $R \simeq 0.9$ are just buildable. Considerable computer time was spent searching for the highest rate codes that appear in later sections.

2.2.2 Weakness of Steiner system codes with $j = 3$.

Having established that certain high rate, small blocklength codes can be constructed, we now ask whether we expect these codes to be good codes. Randomly chosen Gallager codes with large enough blocklength N have good distance properties (Gallager, 1963; MacKay, 1999c), but for small N, some of these properties deteriorate. In the case $j = 3$, there is bad news.

Theorem 1 *Any Gallager code defined by a Steiner system with $j = 3$ has minimum distance less than or equal to 10.*

	n_1	n_2	n_3	n_4	n_5
a	1	1			
b_1	1		1		
b_2	1			1	
b_3		1	1		
b_4		1		1	
c_1			1		1
c_2				1	1
d					1

Figure 3: Construction of a $(5, 1)$ near-codeword (or a weight 4 codeword) in a Steiner system code.

Proof: We define a (w, v) *near-codeword* of a code with parity check matrix $\mathbf{H}$ to be a vector $\mathbf{x}$ with weight w whose syndrome $\mathbf{z}(\mathbf{x}) \equiv \mathbf{H}\mathbf{x}$ has weight v.

We prove that the Gallager code derived from a Steiner triple system has words of weight 10 by counting how many $(5, 1)$ near-codewords it has. If a code with $M \times N$ parity check matrix $\mathbf{H}$ has more than M distinct $(w, 1)$ near-codewords, then its minimum distance is at most $2w$, because, by the pigeonhole principle, there must be at least two of them whose syndromes are identical; the sum of these two near-codewords must be a codeword of weight at most $2w$.

We can generate a $(5, 1)$ near-codeword using the Steiner system property as follows. First, pick a row of $\mathbf{H}$; we call this row a. (M choices.) Second, pick two columns n_1, n_2 satisfying $H_{an} = 1$. ($\binom{k}{2}$ choices.) These two columns define a $(2, 4)$ near-codeword. Call the rows in which the syndrome of this word is non-zero rows b_1, b_2, b_3 and b_4. Third, add two more columns n_3, n_4 to make either a $(4, 2)$ near-codeword or a weight 4 codeword. There are two choices for n_3, n_4, one of which is illustrated diagrammatically in figure 3. Either, as shown in the figure, n_3 is the column in which points b_1 and b_3 appear and column n_4 contains b_2 and b_4; or n_3 contains b_1 and b_4 and n_4 contains b_2 and b_3. Call the new rows introduced by columns n_3 and n_4 rows c_1 and c_2. These rows might be the same as each other, in which case we have found a weight 4 codeword (n_1, n_2, n_3, n_4). Otherwise, rows c_1 and c_2 take us to a unique fifth column n_5 which contains points c_1 and c_2. Adding this column, we have a $(5, 1)$ near-codeword. The final row d is distinct from rows b_*–c_* but might be identical to row a.

We can create such $(5, 1)$ near-codewords in $M \times \binom{k}{2} \times 2$ ways. We will assume that none of these constructions generated a weight 4 codeword — if one did, then we already have the desired result that the minimum distance $d \leq 10$. Now, are all these

$$2M\binom{k}{2} = Mk(k-1) = M(M-1)(M-3)/4 \tag{10}$$

$(5,1)$ near-codewords distinct, or have we created duplicates? If rows a and d are different, then they are all distinct, because we can hang each subgraph defined by a $(5,1)$ near-codeword from row d; the nearest neighbours of row d are rows c_1 and c_2; the next nearest neighbours are the rows b_*; and the furthest row in the subgraph from d is row a. Thus we can recover the $2M\binom{k}{2}$ choices that produced the word. If a and d are equal in one $(5,1)$ near-codeword, however, then the above procedure will generate the same near-codeword in three ways (starting from (n_1, n_2), (n_1, n_5), and (n_2, n_5)). So the number of $(5,1)$ near-codewords is at least

$$C = M(M-1)(M-3)/(4 \times 3). \tag{11}$$

If M exceeds 7, then C exceeds M, so, by the pigeonhole principle, the code has minimum distance at most 10. □

This negative result for binary Gallager codes with $j = 3$ gives a reason for concentrating on larger values of j when dealing with high rate binary codes.

2.2.3 Properties of high rate codes with small blocklength and $j \geq 4$.

Do the codes derived from Steiner systems with $j \geq 4$ have better distance properties? We do not have a theorem, but using similar pigeonhole arguments to those used above, we conjecture that the best codes corresponding to Steiner systems have minimum distance satisfying the following scaling laws:

$$j = 4: \qquad d \gtrsim \log M \tag{12}$$

$$j \geq 5: \qquad d \gtrsim M^{\frac{j-4}{j-2}} \quad e.g., \left\{ \begin{array}{ll} j = 5: & d \gtrsim M^{1/3} \\ j = 6: & d \gtrsim M^{1/2} \\ j = 8: & d \gtrsim M^{2/3} \end{array} \right. \tag{13}$$

2.3 Weakness of any Gallager codes built from commuting permutations.

Some Steiner systems and other constructions of Gallager codes have the property that the parity check matrix contains a grid of non-overlapping permutation matrices. For example, the matrix

$$\mathbf{H} = \begin{bmatrix} \mathbf{R}_{11} & \mathbf{R}_{12} & \mathbf{R}_{13} & \mathbf{R}_{14} \\ \mathbf{R}_{21} & \mathbf{R}_{22} & \mathbf{R}_{23} & \mathbf{R}_{24} \\ \mathbf{R}_{31} & \mathbf{R}_{32} & \mathbf{R}_{33} & \mathbf{R}_{34} \end{bmatrix}, \tag{14}$$

where $\{\mathbf{R}_i\}$ are permutation matrices, defines a rate 1/4 Gallager code with $j = 3$ and $k = 4$. If the permutations commute, that is, $\mathbf{R}_{ij}\mathbf{R}_{kl} = \mathbf{R}_{kl}\mathbf{R}_{ij}$ — which need not be the case for random constructions, but is the case for some algebraic constructions (John Fan, personal communication) — then the distance properties of the code are limited by the following theorem.

Theorem 2 *If a parity check matrix of height M contains a submatrix of height M and width $(j+1)M/j$ containing $j(j+1)$ non-overlapping permutation matrices that all commute with each other, then the corresponding code has minimum distance less than or equal to $(j+1)!$.*

This result applies to Gallager codes of any rate, not just high rate codes. For example, a code with $j = 3$ built from commuting permutations has distance at most 24, and a similar code with $j = 4$ has distance at most 120.

Proof: We call the vertical and horizontal divisions of the matrix of size M/j 'blocks'. We will construct a codeword of a matrix $\mathbf{H}$ whose size is j row-blocks $\times$ $(j+1)$ column-blocks, for example, if $j = 3$, the matrix in equation (14). This matrix is in general a sub-matrix of the parity check matrix from which we started. In each of the column-blocks $1, 2, \ldots, (j+1)$ we will set $j!$ bits to 1 as follows. Define the operator d_h associated with column-block h ($h = 1 \ldots (j+1)$) to be the 'determinant' (modulo 2) obtained from the $j \times j$ matrix given by deleting column-block h from the matrix $\mathbf{H}$. For example, for the case $j = 3$,

$$\begin{aligned} d_2 \quad = \quad & \mathbf{R}_{11}\mathbf{R}_{23}\mathbf{R}_{34} + \mathbf{R}_{11}\mathbf{R}_{24}\mathbf{R}_{33} + \mathbf{R}_{13}\mathbf{R}_{21}\mathbf{R}_{34} + \\ & \mathbf{R}_{13}\mathbf{R}_{24}\mathbf{R}_{31} + \mathbf{R}_{14}\mathbf{R}_{21}\mathbf{R}_{33} + \mathbf{R}_{14}\mathbf{R}_{23}\mathbf{R}_{31} \end{aligned} \tag{15}$$

Each of these operators has weight $j!$, that is, if we hit a weight-one vector of length M/j with d_h, we get a vector of weight at most $j!$. We can now make a codeword $\mathbf{w}$ starting from any weight-one vector of length M/j, $\mathbf{x}$, thus:

$$\mathbf{w} = (d_1\mathbf{x}, d_2\mathbf{x}, d_3\mathbf{x}, \ldots d_{j+1}\mathbf{x}). \tag{16}$$

Here, the commas correspond to the block boundaries. That this is a codeword can be seen by computing the syndrome in each row. In the top row-block, for example, the syndrome is:

$$\mathbf{R}_{11}d_1\mathbf{x} + \mathbf{R}_{12}d_2\mathbf{x} + \mathbf{R}_{13}d_3\mathbf{x} + \ldots + \mathbf{R}_{1(j+1)}d_{j+1}\mathbf{x} \tag{17}$$

which is equal to the product of $\mathbf{x}$ and the determinant of the square matrix:

$$\begin{vmatrix} \mathbf{R}_{11} & \mathbf{R}_{12} & \cdots & \mathbf{R}_{1(j+1)} \\ \mathbf{R}_{11} & \mathbf{R}_{12} & \cdots & \mathbf{R}_{1(j+1)} \\ \mathbf{R}_{21} & \mathbf{R}_{22} & \cdots & \mathbf{R}_{2(j+1)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{R}_{j1} & \mathbf{R}_{j2} & \cdots & \mathbf{R}_{j(j+1)} \end{vmatrix}, \tag{18}$$

which is zero, since the top two rows are equal. Similarly, the syndrome in any row-block h is the product of a determinant that is equal to zero with $\mathbf{x}$. Thus $\mathbf{w}$ is a codeword, and the distance is at most the weight of $\mathbf{w}$, which is at most $(j+1)!$. □

2.4 Gallager codes are fortuitous runlength-limiting codes

A potential benefit of Gallager codes is that they can be constructed to have a runlength-limiting property.

Optimal runlength-limiting codes for noiseless channels are nonlinear. But if we are using an error correcting code for a noisy channel, it would be nice if we could get the runlength-limiting property for free, as part of the error-correcting code. The standard procedure in discdrives is to use a small inner runlength-limiting code, for example, a nonlinear $(N,K) = (16,15)$ code, and an outer code such as a Reed-Solomon code. This method has the disadvantage that the outer code cannot be given detailed likelihood information from the noisy channel; the errors introduced by the decoder of the inner code are complex.

2.4.1 Getting runlength constraints for free.

If a Gallager code has row weight k and there are N/k rows in the parity check matrix like this (if $k = 5$):

$$\left[\begin{array}{l}
\mathtt{1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0} \\
\mathtt{0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0} \\
\mathtt{0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0} \\
\mathtt{0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1}\ldots
\end{array}\right. \tag{19}$$

or this (if $k = 4$):

$$\left[\begin{array}{l}
\mathtt{1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0} \\
\mathtt{0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0} \\
\mathtt{0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0} \\
\mathtt{0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1}\ldots
\end{array}\right. \tag{20}$$

then these constraints enforce local properties that we can use.

- If k is odd (as in (19)), then these constraints force each block of k successive transmitted bits to have even parity. Since k is odd, this means that there must be at least one `0` in every block of k bits. Thus a Gallager code with odd k is automatically a runlength-limiting code with maximum runlength of `1`s equal to $2(k-1)$. There is no constraint on the maximum runlength of `0`s.

- If k is even, then the original Gallager code is not necessarily a runlength-limiting code, but we can modify the code by adding a constant vector to all codewords in the code. For example, if $k = 4$, we could add the vector

$$\left[\; \mathtt{1\,0\,0\,0\,1\,0\,0\,0\,1\,0\,0\,0 \cdots 1\,0\,0\,0} \right] \tag{21}$$

to all codewords, modifying the decoder appropriately. Now, the number of `1`s in any block of k bits is odd, and so is the number of `0`s. So there must be at least one `1` and one `0` in each of these blocks. So the maximum runlength for `1`s is $2(k-1)$, and the maximum runlength for `0`s is $2(k-1)$.

One could also construct a Gallager code, without impairing its error-correcting capabilities (at least if the channel is a memoryless channel), so that its top rows look like this:

$$\left[\begin{array}{l} \mathtt{1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0} \\ \mathtt{0\,0\,0\,0\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0} \\ \mathtt{0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1\,0\,0\,0\,0} \\ \mathtt{0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,1\,1} \ldots \end{array} \right. , \tag{22}$$

in which case the maximum runlength would be $2k-3$.

For practical purposes, Gallager codes have to have a column weight roughly equal to $j = 3$ or 4. A code with rate $R = (N-M)/N$ and column weight j has a row weight $k \simeq jN/M = j/(1-R)$. So examples of the fortuitous runlength limits that can be obtained with Gallager codes are as follows:

R	$j=3$		$j=4$	
	$2(k-1)$	$2k-3$	$2(k-1)$	$2k-3$
0.25	6	5	9	8
0.50	10	9	14	13
0.75	22	21	30	29
0.90	58	57	78	77

If the rate is about 0.9 then these runlengths are not of much use — the maximum runlength used in current discdrives is about 15 — but perhaps as technology advances, this idea will become useful, especially if lower rate codes are used. The benefits could be substantial: not only would there be an increase of about 6% in the storage capacity if the inner code were removed, but the outer code could be provided with better likelihood information, which, as we will see below, can give a great improvement in performance for codes with appropriate decoders.

Almost-certainly runlength-limiting codes

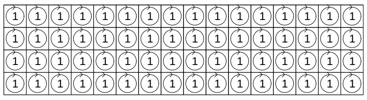
Further methods for making high-rate runlength-limited transmissions without using a nonlinear inner code are described in (MacKay, 1999a). The three key ideas are (a) make the outer code a coset code with the offset varying pseudorandomly from block to block; (b) space the parity bits of the outer code uniformly through the block and put aside a small number of source bits that can be set arbitrarily so that the parity bits take on the values required to satisfy the runlength constraints; and (c) flip any remaining bits that need to be changed, and rely on the outer code to correct them. By combining these methods, we can remove the need for any complicated inner code.

3 Empirical results

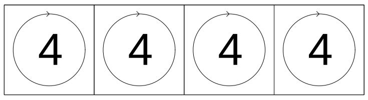
3.1 Construction of Gallager codes

There are various methods for randomly constructing a parity check matrix with given j and k. When we make codes with large blocklengths these alternative methods generally give codes with equivalent performance. When the blocklengths are small, however, good codes become more difficult to find. We have implemented two construction methods. Both these methods attempt to constrain the maximum overlap between two columns in the matrix to be one. We find this constraint to be more important in codes with small blocklengths than it was with large blocklengths.

Permutation matrix method. This method is similar to Gallager's (see the appendix of his book), except that the largest possible sizes of random permutation matrix are used. This distinction is shown pictorially for rate 3/4 codes by the following figures, in which integers show the number of superposed permutation matrices in each square.



Small permutation matrices



Large permutation matrices

Left to right method. This is construction 1A from MacKay and Neal.

We have concentrated on regular constructions with columns weights $j = 3$ and $j = 4$. For high rate codes with small blocklengths, a column weight $j = 3$ gives weak codes, having small numbers of low-weight codewords. We therefore report results for column weight $j = 4$ only.

3.2 Binary Gallager codes, Gaussian channel

3.2.1 Method.

A sequence of noise levels was selected. At each noise level, a large number of block decodings was simulated. The decoding algorithm was run for up to 1000 iterations, halting earlier if the best guess of the decoder corresponded to a valid codeword. The outcome of each decoding was either a success (*i.e.*, the algorithm returns the transmitted codeword without any errors) or a failure. There are two possible types of failures.

Detected errors. The decoding algorithm failed to find a valid codeword. We could call these failures block erasures.

Undetected errors. The decoding algorithm halts in a valid codeword that differs from the transmitted codeword. This failure mode is expected to be very rare in codes that have good distance properties.

3.2.2 Errors observed.

The codes with column weight $j = 4$ have never made undetected errors in these experiments. In the graphs, the 'undetected' error bars show empirical *upper bounds* on the probability of undetected error. We might conjecture that these codes have minimum distance similar to the Gilbert-Varshamov distance, and that undetected errors only occur when the maximum likelihood decoder would also make undetected errors. Using this conjecture, the probability of undetected error can be bounded above by the probability of error of the maximum likelihood decoder, which could probably be computed.

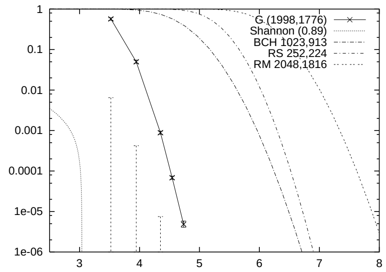
3.2.3 Decoding times.

Figure 4 shows the cumulative distribution of decoding times for the code `s2.94.594` at three noise levels. The decoding usually halts in fewer than ten iterations. Under good conditions three iterations usually suffice.

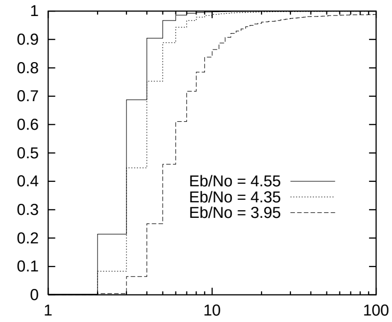
The number of arithmetical operations per iteration is about four times the number of 1s in the parity check matrix. That makes 16 operations per iteration per transmitted bit, or 32000 operations per iteration if $N = 2000$.

3.3 Non-binary Gallager codes, q-ary symmetric channel

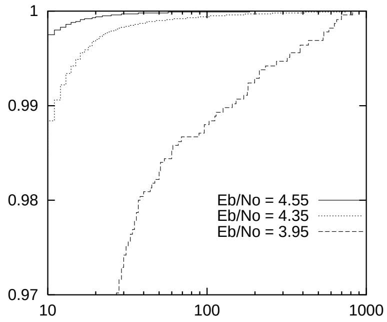
Gallager codes over $GF(q)$ were first reported in (Davey and MacKay, 1998b); improved performance was gained at the expense of a decoding complexity that scaled as q^2. This complexity can be reduced using a Fourier transform of the probabilities (Richardson and Urbanke, 1998).



(a)



(b)



(c)

Figure 4: Regular Gallager code with rate $R = 8/9$ and $N = 1998$. (a) Dependence of block error rate on signal to noise ratio. Weight per column $t = 4$ and transmitted blocklength $N = 1998$. Vertical axis: block error rate. Horizontal axis: E_b/N_0 (decibels). Also shown are performance curves for Reed-Solomon, Reed-Muller and BCH codes with similar rate. These curves assume that the channel outputs are thresholded to give binary signals to the decoder. (That is, no soft decoders for the algebraic codes.) (b) Decoding times, cumulative distribution. (c) Detail from (b). The parity check matrix of this code, `s2.94.594`, can be found in the online archive (MacKay, 1999b).

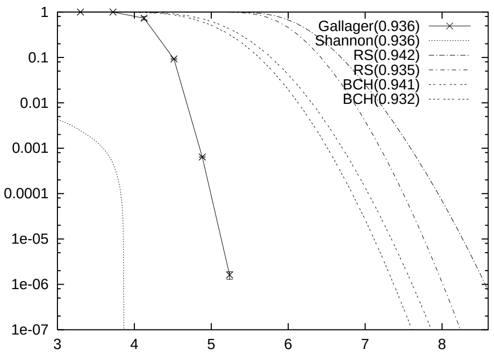


Figure 5: Regular Gallager code with blocklength 4376 and rate 0.936. Dependence of block error rate on signal to noise ratio. Weight per column $t = 4$. Also shown are performance curves for two nearby RS codes and two BCH codes with similar rate and blocklength 1023. These curves assume that the RS symbols are transmitted over the binary Gaussian channel and that the outputs are thresholded to give binary signals to the decoder. (That is, no soft decoders for the algebraic codes.) The parity check matrix of this code, `4376.282.4.9598`, can be found in the online archive (MacKay, 1999b).

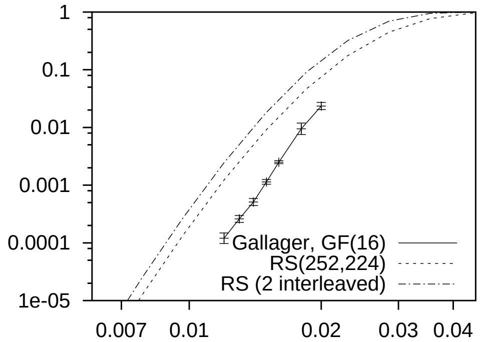


Figure 6: Gallager code over GF(16) applied to 16-ary symmetric channel. Weight per column $t = 3$. Vertical axis: block error rate. Horizontal axis: symbol error probability. The code is compared with two Reed-Solomon codes similar to those used in discdrives.

In the following, we use the notation of (Davey and MacKay, 1998b). Let $\mathcal{N}(m) := \{n : H_{mn} \neq 0\}$ be the set of noise symbols that participate in check m. The decoder needs to update quantities r^a_{mn}, the probability of check m being satisfied if symbol n of the message $\mathbf{x}$ is considered fixed at $a \in GF(q)$ and the other noise symbols have a separable distribution given by the probabilities $\{s^b_{mn'} : n' \in \mathcal{N}(m) \backslash n\}$. The new value of r^a_{mn} is:

$$r^a_{mn} = \sum_{\mathbf{x}:x_n=a} \delta \left(\sum_{n' \in \mathcal{N}(m)} H_{mn'} x_{n'} = z_m \right) \prod_{j \in \mathcal{N}(m) \backslash n} s^{x_j}_{mj} \tag{23}$$

This function of a is a convolution of the quantities s^a_{mj}, and so the summation can be replaced by a product of the Fourier transforms (taken over the additive group of $GF(q)$) of s^a_{mj} for $j \in \mathcal{N}(m) \backslash n$, followed by an inverse Fourier transform. The Fourier transform F of a function f over $GF(2)$ is given by $F^0 = f^0 + f^1, F^1 = f^0 - f^1$. Transforms over $GF(2^k)$ can be viewed as a sequence of binary transforms in each of k dimensions. Hence for $GF(4)$ we have

$$\begin{aligned}
F^0 &= [f^0 + f^1] + [f^2 + f^3] && (24)\\
F^1 &= [f^0 - f^1] + [f^2 - f^3] && (25)\\
F^2 &= [f^0 + f^1] - [f^2 + f^3] && (26)\\
F^3 &= [f^0 - f^1] - [f^2 - f^3] && (27)
\end{aligned}$$

The inverse transform is the same, except that we also divide by 2^k.

With a slight abuse of notation, let $(S^0_{mj}, \ldots, S^{q-1}_{mj})$ represent the Fourier transform of the vector $(s^0_{mj}, \ldots, s^{q-1}_{mj})$, after permuting the components to take account of the matrix entry H_{mj}. Now r^a_{mn} is the ath coordinate of the inverse transform of

$$\left(\left(\prod_{j \in \mathcal{N}(m) \backslash n} S^0_{mj} \right), \ldots, \left(\prod_{j \in \mathcal{N}(m) \backslash n} S^{q-1}_{mj} \right) \right) \tag{28}$$

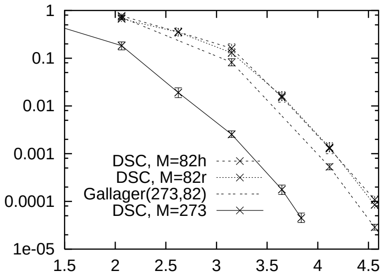
The update of the quantities s is unchanged.

Each fast Fourier transform takes $q \log_2 q$ additions and q multiplications. Assuming a column weight $j = 3$ and taking $q = 16$, the total cost per iteration is 96 additions and 72 multiplications per bit. All these operations can be implemented in low precision arithmetic with a small loss in performance (Richardson and Urbanke, 1998).

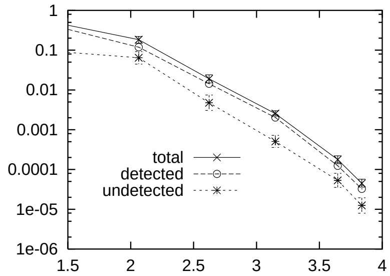
Figure 6 shows the performance of a column weight 3 rate 8/9 Gallager code over $GF(16)$ applied to 16-ary symmetric channel. The code is compared with two Reed-Solomon codes.

N	7	21	73	273	1057	4161
M_{True}	4	**10**	**28**	**82**	**244**	**730**
K	3	11	45	191	813	3431
d	4	**6**	**10**	**18**	34	66
k	3	**5**	**9**	17	33	65

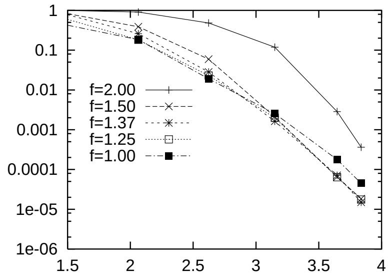
(a)



(b)



(c)



(d)

Figure 7: Difference-set cyclic codes — low–density parity–check codes satisfying many redundant constraints — outperform equivalent Gallager codes. (a) The table shows the N, M_{True} (the number of independent rows in the parity check matrix, as opposed to the total number of rows M), K, distance d, and row weight k of some difference-set cyclic codes, highlighting the codes that have large d/N, small k, and large N/M. All DSC codes satisfy N constraints of weight k. (b) In the comparison the Gallager code had $(j, k) = (4, 13)$, and rate identical to the DSC codes. Vertical axis: block error probability; horizontal axis: E_b/N_0/dB.
(c) Decomposition of the DSC code's errors into detected and undetected errors.
(d) The error rate of the DSC code can be slightly reduced by using a 'fudge factor' of 1.25 or 1.37 during the sum-product decoding.

4 Difference-set cyclic codes

The performance of Gallager codes can be enhanced by making a non-random code with *redundant sparse constraints* (Tanner, 1981; Lucas *et al.*, 1999). There is a difference-set cyclic code, for example, that has $N = 273$, and $K = 191$, but the code satisfies not $M = 82$ but N, *i.e.*, *273*, low-weight constraints (figure 7). It is impossible to make random Gallager codes that have anywhere near this much redundancy among their checks. The redundant checks allow the sum-product algorithm to work substantially better, as shown in figure 7, in which a DSC code outperforms a comparable regular binary Gallager code by about 0.7 dB. The (73,45) DSC code has been implemented on a chip by Karplus and Krit (1991) following a design of Tanner (1981). Product codes are another family of codes with redundant constraints. For example, the product with itself of a $(n, k) = (64, 57)$ Hamming code satisfying $m = 7$ constraints is a $(N, K) = (n^2, k^2) = (4096, 3249)$ code. The number of independent constraints is $M = 847$, but the sum-product decoder can make use of $2nm = 896$ equivalent constraints. Such codes have recently been named 'turbo product codes', but we think they should be called 'Tanner product codes', since they were first investigated by Tanner (1981). Product codes have the disadvantage, however, that their distance does not scale well with blocklength; the distance of a product code with blocklength n^2, built from two codes with distance d, is only d^2, so the ratio of distance to blocklength falls.

An open problem is to discover codes sharing the remarkable properties of the difference-set cyclic codes but with larger blocklengths and arbitrary rates. I call this task the Tanner challenge, in honour of Michael Tanner, who recognised the importance of such codes twenty years ago.

4.1 Notes on DSC codes

Do the extra checks help?

To confirm that the extra 191 redundant parity checks are responsible for the good performance of the DSC code, we tried decoding the code using only 82 of the parity checks. In case `82h`, we took the first 82 rows of the cyclic parity check matrix; in case `82r` we picked 82 random non-redundant rows from the matrix. This choice appears to make little difference. Either way, the performance is much worse than that of the code using the full $M = 273$ checks (figure 7(b)). The random Gallager code with $j = 4$ performs slightly better than either of the crippled DSC codes.

Undetected errors.

The (273,191) DSC code makes undetected errors. The frequency of these errors is shown in figure 7(c).

Rescaling the log-probability messages.

An *ad hoc* procedure found helpful by Tanner (personal communication) involves scaling down the log-probabilities by a 'fudge factor' f at the end of each iteration. We seize the message $a_{mn} \equiv \log \frac{q^{(1)}_{mn}}{q^{(0)}_{mn}}$ on its way from bit n to check m, and replace it by a_{mn}/f. Experimenting with a range of values of f, we find that values slightly greater than 1 reduce the error probability a little at large E_b/N_0. Figure 7(d) shows graphs for $f = 1$, $f = 1.25$, $f = 1.37$, $f = 1.50$, and $f = 2$. This fudge appears to reduce the frequency of detected errors and has little effect on the frequency of undetected errors, so that the error probability is dominated by undetected errors. We speculate that the fudged algorithm is indistinguishable from the optimal decoder for this DSC code, and the performance is only limited by the code's distance properties.

5 Discussion

Comparisons of Gallager codes with Reed-Solomon codes have been made before by Worthen and Stark (1998). They made a belief propagation decoder appropriate for bursty channels and achieved 3 dB performance gain over the Reed-Solomon code for rate 1/2 and block size of about 10000. Worthen and Stark attributed 2 dB of the gain to the use of soft decisions rather than hard decisions and 1 dB to code improvement. Using that reasoning, the gain of the shorter-blocklength Gallager codes over Reed-Solomon codes in our paper can be attributed entirely to using soft decisions. Our work makes a case for finding good short length codes that use soft decisions. Gallager codes over $GF(16)$ appear to be good candidates for this role.

The task of constructing Gallager codes with very short block lengths remains an interesting area for further research.

Acknowledgements

This work was supported by the Gatsby Foundation. DJCM thanks the researchers at the Sloane Center, Department of Physiology, University of California at San Francisco, for their generous hospitality, and Elywn Berlekamp, Michael Luby, Virginia de Sa, Bob McEliece, Emina Soljanin, Clifton Williamson, John Morris and Bernardo Rub for helpful discussions.

References

Berlekamp, E. R. (1968) *Algebraic Coding Theory*. New York: McGraw-Hill.

Davey, M. C., and MacKay, D. J. C. (1998a) Low density parity check codes over GF(q). In *Proceedings of the 1998 IEEE Information Theory Workshop*, pp. 70–71. IEEE.

Davey, M. C., and MacKay, D. J. C. (1998b) Low density parity check codes over GF(q). *IEEE Communications Letters* **2** (6): 165–167.

Gallager, R. G. (1962) Low density parity check codes. *IRE Trans. Info. Theory* **IT-8**: 21–28.

Gallager, R. G. (1963) *Low Density Parity Check Codes*. Number 21 in Research monograph series. Cambridge, Mass.: MIT Press.

Karplus, K., and Krit, H. (1991) A semi–systolic decoder for the PDSC–73 error–correcting code. *Discrete Applied Mathematics* **33**: 109–128.

Lucas, R., Fossorier, M., Kou, Y., and Lin, S., (1999) Iterative decoding of one-step majority logic decodable codes based on belief propagation. Submitted.

MacKay, D. J. C., (1999a) Almost–certainly runlength–limiting codes. `wol.ra.phy.cam.ac.uk/mackay`.

MacKay, D. J. C., (1999b) Encyclopedia of sparse graph codes (hypertext archive). `http://wol.ra.phy.cam.ac.uk/mackay/codes/data.html`.

MacKay, D. J. C. (1999c) Good error correcting codes based on very sparse matrices. *IEEE Transactions on Information Theory* **45** (2): 399–431.

MacKay, D. J. C., and Neal, R. M. (1996) Near Shannon limit performance of low density parity check codes. *Electronics Letters* **32** (18): 1645–1646. Reprinted *Electronics Letters*, **33**(6):457–458, March 1997.

Richardson, T., and Urbanke, R., (1998) The capacity of low-density parity check codes under message-passing decoding. Submitted to IEEE Trans. on Information Theory.

Tanner, R. M. (1981) A recursive approach to low complexity codes. *IEEE Transactions on Information Theory* **27** (5): 533–547.

Urbanke, R., Richardson, T., and Shokrollahi, A., (1999) Design of provably good low density parity check codes. Submitted.

Worthen, A., and Stark, W. (1998) Low–density parity check codes for fading channels with memory. In *Proceedings of the 36th Allerton Conference on Communication, Control, and Computing, Sept. 1998*, pp. 117–125.