Distance Bounds for Convolutional Codes and Some Optimal Codes

Heide Gluesing-Luerssen* and Wiland Schmale*

May 6, 2003

Abstract

After a discussion of the Griesmer and Heller bound for the distance of a convolutional code we present several codes with various parameters, over various fields, and meeting the given distance bounds. Moreover, the Griesmer bound is used for deriving a lower bound for the field size of an MDS convolutional code and examples are presented showing that, in most cases, the lower bound is tight. Most of the examples in this paper are cyclic convolutional codes in a generalized sense as it has been introduced in the seventies. A brief introduction to this promising type of cyclicity is given at the end of the paper in order to make the examples more transparent.

Keywords: Convolutional coding theory, distance bounds, cyclic convolutional codes.

MSC (2000): 94B10, 94B15, 16S36

1 Introduction

The fundamental task of coding theory is the construction of good codes, that is, codes having a large distance and a fast decoding algorithm. This task applies equally well to block codes and convolutional codes. Yet, the state of the art is totally different for these two classes of codes. The mathematical theory of block codes is highly developed and has produced many sophisticated classes of codes, some of which, like BCH-codes, also come with an efficient decoding algorithm. On the other hand, the mathematical theory of convolutional codes is still in the beginnings. Engineers make use of these codes since decades, but all convolutional codes used in practice have been found by systematic computer search and their distances have been found by computer as well, see for instance [12] and [9, Sec. 8] for codes having the largest distance among all codes with the same parameters. Moreover, in all practical situations decoding of convolutional codes is done by search algorithms, for instance the Viterbi algorithm or one of the sequential decoding algorithms, e. g. the stack algorithm. It depends on the algorithm how complex a code may be without exceeding the range of the decoding algorithms. However, the important fact about the theory of convolutional codes is that so far no specific codes are known that allow an *algebraic decoding* (in the present paper

*Department of Mathematics, University of Oldenburg, 26 111 Oldenburg, Germany, email: gluesing@mathematik.uni-oldenburg.de and wiland.schmale@uni-oldenburg.de

a decoding algorithm will be called algebraic if it is capable to exploit the specific structure of the given code in order to avoid a full search).

Since the seventies quite some effort has been made in order to find algebraic constructions of convolutional codes that guarantee a large (free) distance [10, 15, 11, 23, 4]. The drawbacks of all these constructions are that, firstly, the field size has to be adapted and in general becomes quite large and, secondly, so far no algebraic decoding for these codes is known. A main feature of most of these constructions is that they make use of cyclic block codes in order to derive the desired convolutional code.

Parallel to these considerations there was an independent investigation of convolutional codes that have a cyclic structure themselves, which also began in the seventies [18, 19, 6, 5]. It was the goal of these papers to see whether this additional structure has, just like for block codes, some benefit for the error-correcting capability of the code. The first and very important observation of the seventies was the fact that a convolutional code which is cyclic in the usual sense is a block code. This negative insight has led to a more complex notion of cyclicity for convolutional codes. The algebraic analysis of these codes has been completed only recently in [5] and yields a nice, yet nontrivial, generalization of the algebraic situation for cyclic block codes. Furthermore, by now plenty of optimal cyclic convolutional codes have been found in the sense that their (free) distance reaches the Griesmer bound. To the best of our knowledge it was, for most cases of the parameters, not known before whether such optimal codes existed. Many of these codes are over small fields (like the binary field) and are therefore well-suited for the existing decoding algorithms. Along with the algebraic theory of [5] all this indicates that this notion of cyclicity is not only the appropriate one for convolutional codes but also a very promising one. Yet, the theory of these codes is still in the beginnings. So far, no theoretical results concerning the distance of such a code or its decoding properties are known. But we are convinced that this class of codes deserves further investigation and that the theory developed so far will be a good basis for the next steps.

It is the aim of this paper to present many of these examples in order to introduce the class of cyclic convolutional codes to the convolutional coding community. The examples are presented via a generator matrix so that no knowledge about cyclicity for convolutional codes is required from the reader. The (free) distances of all these codes have been obtained by a computer program. A detailed discussion of various distance bounds for convolutional codes over arbitrary fields shows that all the given codes are optimal with respect to their distance. It is beyond the scope of this paper to acquaint the reader with the theory of cyclic convolutional codes. However, in Section 5 we will give a very brief introduction into this subject so that the reader may see how the examples have been constructed. The details of the theory can be found in [5].

The outline of the paper is as follows. After reviewing the main notions of convolutional coding theory in the next section we will discuss in Section 3 various bounds for the free distance of a convolutional code, the Griesmer bound, the Heller bound and the generalized Singleton bound. The first two bounds are well-known for binary convolutional codes and can straightforwardly be generalized to codes over arbitrary fields. It is also shown that for all sets of parameters the Griesmer bound is at least as good as the Heller bound. The generalized Singleton bound is an upper bound for the free distance of a code of given length, dimension, and complexity, but over an arbitrary field. Just like for block codes a code reaching this bound is called an MDS code [22]. The Griesmer bound is used for showing how large the field size has to be in order to allow for an MDS code. In Section 4 many examples of codes are presented reaching the respective bound. Most of these examples are

cyclic convolutional codes, but we also include some other codes with the purpose to exhibit certain features of convolutional codes. For instance, we give examples of MDS codes showing that the lower bounds for the field size as derived in Section 3 are tight. Furthermore, an example is given showing that a code reaching the Griesmer bound may have extreme Forney indices, a phenomenon that does not occur for MDS codes. The paper concludes with a brief account of cyclicity for convolutional codes.

2 Preliminaries

We will make use of the following notation. The symbol $\mathbb{F}$ stands for any finite field while $\mathbb{F}_q$ always denotes a field with q elements. The ring of polynomials and the field of formal Laurent series over $\mathbb{F}$ are given by

$$\mathbb{F}[z] = \Big\{ \sum_{j=0}^{N} f_j z^j \,\Big|\, N \in \mathbb{N}_0,\ f_j \in \mathbb{F} \Big\} \quad \text{and} \quad \mathbb{F}(\!(z)\!) = \Big\{ \sum_{j=l}^{\infty} f_j z^j \,\Big|\, l \in \mathbb{Z},\ f_j \in \mathbb{F} \Big\}.$$

The following definition of a convolutional code is standard.

Definition 2.1 Let $\mathbb{F} = \mathbb{F}_q$ be a field with q elements. An $(n, k, \delta)_q$-*convolutional code* is a k-dimensional subspace $\mathcal{C}$ of the vector space $\mathbb{F}(\!(z)\!)^n$ of the form

$$\mathcal{C} = \operatorname{im} G := \big\{ uG \,\big|\, u \in \mathbb{F}(\!(z)\!)^k \big\}$$

where $G \in \mathbb{F}[z]^{k\times n}$ satisfies
(a) G is *right invertible*, i. e. there exists some matrix $\tilde{G} \in \mathbb{F}[z]^{n\times k}$ such that $G\tilde{G} = I_k$.
(b) $\delta = \max\{\deg \gamma \mid \gamma \text{ is a } k\text{-minor of } G\}$.
We call G a *generator matrix* and δ the *complexity* of the code $\mathcal{C}$.

The complexity is also known as the *overall constraint length* [9, p. 55] or the *degree* [16, Def. 3.5] of the code. Notice that a generator matrix is always polynomial and has a polynomial right inverse. This implies that in the situation of Definition 2.1 the polynomial codewords belong to polynomial messages, i. e.

$$\mathcal{C} \cap \mathbb{F}[z]^n = \big\{ uG \,\big|\, u \in \mathbb{F}[z]^k \big\}. \tag{2.1}$$

In other words, the generator matrix is delay-free and non-catastrophic. As a consequence, a convolutional code is always uniquely determined by its polynomial part. Precisely, if $\mathcal{C} = \operatorname{im} G$ and $\mathcal{C}' = \operatorname{im} G'$ where G, $G' \in \mathbb{F}[z]^{k\times n}$ are right invertible, then

$$\mathcal{C} = \mathcal{C}' \Longleftrightarrow \mathcal{C} \cap \mathbb{F}[z]^n = \mathcal{C}' \cap \mathbb{F}[z]^n. \tag{2.2}$$

This follows from (2.1) and the fact that $\{uG \mid u \in \mathbb{F}[z]^k\} = \{uG' \mid u \in \mathbb{F}[z]^k\}$ is equivalent to $G' = VG$ for some matrix $V \in \mathbb{F}[z]^{k\times k}$ that is invertible over $\mathbb{F}[z]$. This also shows that the complexity of a code does not depend on the choice of the generator matrix. From all this it should have become clear that with respect to code construction there is no difference whether one works in the context of infinite message and codeword sequences (Laurent series) or finite ones (polynomials) as long as one considers right invertible generator matrices. Only for decoding it becomes important whether or not one may assume the sent codeword to be

finite. The issue whether convolutional coding theory should be based on finite or infinite message sequences, has first been raised and discussed in detail in [21, 20].

It is well-known [2, Thm. 5] or [3, p. 495] that each convolutional code has a minimal generator matrix in the sense of the next definition. In the same paper [3, Sec. 4] it has been shown how to derive such a matrix from a given generator matrix in a constructive way.

Definition 2.2 (1) For $v = \sum_{j=0}^{N} v_j z^j \in \mathbb{F}[z]^n$ where $v_j \in \mathbb{F}^n$ and $v_N \neq 0$ let $\deg v := N$ be the *degree* of v. Moreover, put $\deg 0 = -\infty$.

(2) Let $G \in \mathbb{F}[z]^{k\times n}$ be a right invertible matrix with complexity $\delta = \max\{\deg \gamma \mid \gamma \text{ is a } k\text{-minor of } G\}$ and let $\nu_1, \ldots, \nu_k$ be the degrees of the rows of G in the sense of (1). We say that G is *minimal* if $\delta = \sum_{i=1}^{k} \nu_i$. In this case, the row degrees of G are uniquely determined by the code $\mathcal{C} := \operatorname{im} G \subseteq \mathbb{F}(\!(z)\!)^n$. They are called the *Forney indices* of $\mathcal{C}$ and the number $\max\{\nu_1, \ldots, \nu_k\}$ is said to be the *memory* of the code. An $(n, k, \delta)_q$-code with memory m is also called an $(n, k, \delta; m)_q$-code.

From the above it follows that an $(n, k, \delta)_q$-convolutional code has a constant generator matrix if and only if $\delta = 0$. In that case the code can be regarded as an $(n, k)_q$-block code.

The definition of the distance of a convolutional code is straightforward. For a constant vector $w = (w_1, \ldots, w_n) \in \mathbb{F}^n$ we define its *(Hamming) weight* as $\operatorname{wt}(w) = \#\{i \mid w_i \neq 0\}$. For a polynomial vector $v = \sum_{j=0}^{N} v_j z^j \in \mathbb{F}[z]^n$, where $v_j \in \mathbb{F}^n$, the *weight* is defined as $\operatorname{wt}(v) = \sum_{j=0}^{N} \operatorname{wt}(v_j)$. Then the *(free) distance* of a code $\mathcal{C} \subseteq \mathbb{F}(\!(z)\!)^n$ with generator matrix $G \in \mathbb{F}[z]^{k\times n}$ is given as

$$\operatorname{dist}(\mathcal{C}) := \min\left\{\operatorname{wt}(v) \,\middle|\, v \in \mathcal{C} \cap \mathbb{F}[z]^n,\ v \neq 0\right\}.$$

By virtue of (2.1) this can be rephrased as $\operatorname{dist}(\mathcal{C}) = \min\{\operatorname{wt}(uG) \mid u \in \mathbb{F}[z]^k,\, u \neq 0\}$.

When presenting some optimal codes in Section 4 we will also investigate the column distances of the codes. For each $l \in \mathbb{N}_0$ the lth *column distance* of $\mathcal{C}$ is defined as

$$d_l^c = \min\left\{\operatorname{wt}\big((uG)_{[0,l]}\big) \,\middle|\, u \in \mathbb{F}[z]^k, u_0 \neq 0\right\} \tag{2.3}$$

where for a polynomial vector $v = \sum_{j=0}^{N} v_j z^j$ we define $v_{[0,l]} = \sum_{j=0}^{\min\{N,l\}} v_j z^j$. It can easily be shown [9, Thm. 3.4] that for each code $\mathcal{C}$ there exists some $M \in \mathbb{N}_0$ such that

$$d_0^c \leq d_1^c \leq d_2^c \ldots \leq d_M^c = d_{M+1}^c = \ldots = \operatorname{dist}(\mathcal{C}). \tag{2.4}$$

3 Distance Bounds

In this section we want to present some upper bounds for the distance of a convolutional code. These bounds are quite standard for binary convolutional codes and can be found in Chapter 3.5 of the book [9]. The proof for arbitrary fields goes along the same lines of arguments, but for sake of completeness we wish to repeat the arguments in this paper. We will also compare the numerical values of the bounds with each other.

Let us begin with recalling various distance bounds for block codes. The Plotkin bound as given below can be found in [1, 1.4.3], but can also easily be derived from the more familiar formula

$$\text{if } d > \theta n \text{ where } \theta = \frac{q-1}{q}, \text{ then } q^k \leq \frac{d}{d-\theta n}, \tag{3.1}$$

see for instance [13, (5.2.4)]. As for the Singleton and the Griesmer bound we also refer to [13, Ch. 5.2].

Theorem 3.1 *Let $\mathcal{C} \subseteq \mathbb{F}^n$ be an $(n,k)_q$-block code and let $d = \operatorname{dist}(\mathcal{C})$. Then*

$$\begin{aligned} &d \leq n-k+1 && \textit{(Singleton bound)},\\ &d \leq \Big\lfloor \frac{nq^{k-1}(q-1)}{q^k-1} \Big\rfloor && \textit{(Plotkin bound)},\\ &\sum_{l=0}^{k-1} \Big\lceil \frac{d}{q^l} \Big\rceil \leq n && \textit{(Griesmer bound)}. \end{aligned}$$

An $(n,k)_q$-code $\mathcal{C}$ with $\operatorname{dist}(\mathcal{C}) = n-k+1$ is called an MDS code.

Notice that the Singleton bound does not take the field size into account. As a consequence the question arises as to how large the field size q has to be in order to allow the existence of MDS codes and how to construct such codes. Answers in this direction can be found in [14, Ch. 11].

It is certainly well-known that the Griesmer bound is at least as good as the Plotkin bound. The importance of the Plotkin bound, however, is that it also applies to nonlinear block codes, in which case it is usually given as in (3.1) and with $M := |\mathcal{C}|$ instead of q^k. Since we did not find a comparison of the two bounds for linear block codes in the literature we wish to present a short proof of this statement. We also include the relation between the Griesmer and the Singleton bound.

Proposition 3.2 *Given the parameters n, k, d, and $q \in \mathbb{N}$ where $k < n$ and q is a prime power. Assume $\sum_{l=0}^{k-1} \left\lceil \frac{d}{q^l} \right\rceil \leq n$. Then*

(a) $d \leq \Big\lfloor \frac{nq^{k-1}(q-1)}{q^k-1} \Big\rfloor$,

(b) $d \leq n-k+1$.

There is no relation between the Plotkin and the Singleton bound in this generality. Roughly speaking, for relatively large values of q the Singleton bound is better than the Plotkin bound while for small values the Plotkin bound is better.

Proof: (a) Assume to the contrary that $d > \lfloor \frac{nq^{k-1}(q-1)}{q^k-1} \rfloor$. Since d is an integer this implies that $d > \frac{nq^{k-1}(q-1)}{q^k-1}$. Thus

$$\sum_{l=0}^{k-1} \Big\lceil \frac{d}{q^l} \Big\rceil \geq \sum_{l=0}^{k-1} \frac{d}{q^l} > \sum_{l=0}^{k-1} \frac{n(q-1)}{q^k-1} q^{k-1-l} = \frac{n(q-1)}{q^k-1} \sum_{l=0}^{k-1} q^l = n.$$

(b) follows from $\sum_{l=0}^{k-1} \lceil \frac{d}{q^l} \rceil \geq d+k-1$. □

One should also recall that the Griesmer bound is not tight. An example is given by the parameters $n = 13$, $k = 6$, $q = 2$ in which case the Griesmer bound shows that the distance is upper bounded by 5. But it is known that no $(13,6)_2$-code with distance 5 exists, see [13, p. 69].

We will now present the generalization of these bounds to convolutional codes. Let us begin with the Singleton bound. The following result has been proven in [22, Thm. 2.2].

Theorem 3.3 *Let $\mathcal{C} \subseteq \mathbb{F}(\!(z)\!)^n$ be an (n,k,δ)-code. Then*
(a) The distance of $\mathcal{C}$ satisfies

$$\operatorname{dist}(\mathcal{C}) \leq (n-k)\Big(\Big\lfloor \frac{\delta}{k} \Big\rfloor + 1\Big) + \delta + 1 =: S(n,k,\delta).$$

The number $S(n,k,\delta)$ is called the generalized Singleton bound for the parameters (n,k,δ) and we call the code $\mathcal{C}$ an MDS code if $\operatorname{dist}(\mathcal{C}) = S(n,k,\delta)$.

(b) If $\mathcal{C}$ is an MDS code and $\delta = ak + r$ where $a \in \mathbb{N}_0$ and $0 \leq r \leq k-1$, then the Forney indices of $\mathcal{C}$ are given by

$$\underbrace{a, \ldots, a}_{k-r \text{ times}}, \underbrace{a+1, \ldots, a+1}_{r \text{ times}}.$$

Hence the code is compact in the sense of [16, Cor. 4.3].

Just like for block codes the acronym MDS stands for maximum distance separable. In [22, Thm. 2.10] it has been shown that for all given parameters n, k, δ and all primes p there exists an MDS code over a suitably large field of characteristic p. The proof is non-constructive and, as a consequence, does not give a hint about the field size required. In [23, Thm. 3.3] a construction of (n,k,δ)-MDS codes over fields $\mathbb{F}_{p^r}$ is given under the condition that $n|(p^r-1)$ and $p^r \geq \frac{n\delta^2}{k(n-k)}$. Notice that this requires n and the characteristic p being coprime. This result gives first information about the field size required in order to guarantee the existence of an MDS code. However, many examples of MDS codes over smaller fields are known. We will present some of them in the next section. Although they all have a certain structure in common (they are cyclic in the sense of Section 5) we do not know any general construction for cyclic MDS codes yet.

Now we proceed with a generalization of the Plotkin and Griesmer bound to convolutional codes.

Theorem 3.4 *Let $\mathcal{C}$ be an $(n,k,\delta;m)_q$-convolutional code having distance $\operatorname{dist}(\mathcal{C}) = d$. Moreover, let*

$$\hat{\mathbb{N}} = \begin{cases} \mathbb{N} := \{1,2,\ldots\}, & \text{if } km = \delta \\ \mathbb{N}_0 := \{0,1,2,\ldots\}, & \text{if } km > \delta \end{cases}$$

Then

$$d \leq \min_{i \in \hat{\mathbb{N}}} \Big\lfloor \frac{n(m+i)q^{k(m+i)-\delta-1}(q-1)}{q^{k(m+i)-\delta}-1} \Big\rfloor =: H_q(n,k,\delta;m) \qquad \textit{(Heller bound)}$$

$$\begin{aligned} d &\leq \max\Big\{ d' \in \{1,\ldots,S(n,k,\delta)\} \;\Big|\; \sum_{l=0}^{k(m+i)-\delta-1} \Big\lceil \frac{d'}{q^l} \Big\rceil \leq n(m+i) \text{ for all } i \in \hat{\mathbb{N}} \Big\} \\ &=: G_q(n,k,\delta;m) \qquad \textit{(Griesmer bound)} \end{aligned}$$

Moreover, $G_q(n,k,\delta;m) \leq H_q(n,k,\delta;m)$.

In the binary case ($q=2$) both bounds can be found in [9, 3.17 and 3.22]. In that version the first bound has been proven first by Heller in [7]. The Griesmer bound as given above differs slightly from the one given at [9, 3.22]. We have upper bounded the possible values for d' by the generalized Singleton bound, which is certainly reasonable to do. As a consequence, the Griesmer bound is always less than or equal to the generalized Singleton bound. This

would not have been the case had we taken the maximum over all $d' \in \mathbb{N}$. This can be seen by taking the parameters $(n,k,\delta;m)_q = (5,2,3;3)_8$. In this case the generalized Singleton bound is $S(n,k,\delta) = 10$ but the inequalities of the Griesmer bound are all satisfied for the value $d' = 12$.

The proof of the inequalities above is based on the same idea as in the binary case as we will show now.

PROOF: The last statement follows from Proposition 3.2(a). As for the bounds themselves we will see that they are based on certain block codes which appear as subsets of the given convolutional code $\mathcal{C}$. This will make it possible to apply the block code bounds of Theorem 3.1. The subcodes to be considered are simply the subsets of all codewords corresponding to polynomial messages with an upper bounded degree.
Let $\mathcal{C} = \operatorname{im} G$, where $G \in \mathbb{F}[z]^{k\times n}$ is right-invertible and minimal with Forney indices $\nu_1,\ldots,\nu_k$. Hence $\delta = \sum_{i=1}^{k}\nu_i$ and $m = \max\{\nu_1,\ldots,\nu_k\}$. Notice that $km \geq \delta$ and $km = \delta \Longleftrightarrow \nu_1 = \ldots = \nu_k = m$. For each $i \in \mathbb{N}_0$ define

$$U_i = \{(u_1,\ldots,u_k) \in \mathbb{F}[z]^k \mid \deg u_l \leq m+i-1-\nu_l \text{ for } l = 1,\ldots,k\}.$$

This implies $u_l = 0$ if $\nu_l = m$ and $i = 0$. In particular, $U_i = \{0\} \Longleftrightarrow km = \delta$ and $i = 0$ and this shows that $i = 0$ has to be excluded if $km = \delta$. Obviously, the set U_i is an $\mathbb{F}$-vector space and $\dim_{\mathbb{F}} U_i = \sum_{l=1}^{k}(m+i-\nu_l) = k(m+i)-\delta$. Consider now $\mathcal{C}_i := \{uG \mid u \in U_i\}$ for $i \in \mathbb{N}_0$. Then $\mathcal{C}_i \subseteq \mathcal{C}$ and $\mathcal{C}_i$ is an $\mathbb{F}$-vector space and, by injectivity of G,

$$\dim_{\mathbb{F}} \mathcal{C}_i = \dim_{\mathbb{F}} U_i = k(m+i)-\delta.$$

Furthermore, minimality of the generator matrix G tells us that

$$\deg(uG) = \max_{l=1,\ldots,k}(\deg u_l + \nu_l) \leq m+i-1 \text{ for all } u \in U_i,$$

see [3, p. 495]. Hence $\mathcal{C}_i$ can be regarded as a block code of length $n(m+i)$ and dimension $k(m+i)-\delta$ for all $i \in \hat{\mathbb{N}}$. Since $\operatorname{dist}(\mathcal{C}) \leq \operatorname{dist}(\mathcal{C}_i)$ for all $i \in \hat{\mathbb{N}}$ we obtain the desired results by applying the Plotkin and Griesmer bounds of Theorem 3.1 to the codes $\mathcal{C}_i$. □

The proof shows that the existence of an $(n,k,\delta;m)_q$-code meeting the Griesmer bound implies the existence of $(n(m+i),k(m+i)-\delta)_q$-block codes having at least the same distance for all $i \in \hat{\mathbb{N}}$. The converse, however, is not true, since the block codes have to have some additional structure. We will come back to this at the end of this section.

One should note that these bounds do only take the largest Forney index, the memory, into account. More precisely, the proof shows that codewords having degree smaller than $m-1$ are never taken into consideration. As a consequence, codes with a rather bad distribution of the Forney indices will never attain the bound. For instance, for a code with parameters $(n,k,\delta;m)_q = (5,3,4;2)_2$ the Griesmer bound shows that the distance is upper bounded by 6. This can certainly never be attained if the Forney indices of that code are given by $0,2,2$ since in that case a constant codeword exists. Hence the Forney indices have to be $1,1,2$. In this case a code with distance 6 does indeed exist, see the first code given in Table I of Section 4. But also note that, on the other hand, a code reaching the Griesmer bound need not be compact (see Theorem 3.3(b)); an example is given by the $(5,2,6;4)_2$-code given in Table I of the next section.

The Griesmer bound as given above has the disadvantage that infinitely many inequalities have to be considered. A simple way to reduce this to finitely many inequalities is obtained

by making use of the generalized Singleton bound $S(n,k,\delta)$. Instead of this bound one could equally well use any of the numbers occurring on the right hand side of the Heller bound.

Proposition 3.5 *Given the parameters n, k, m, δ such that $k<n$ and $km\geq\delta$ and let q be any prime power. Define the set $\hat{\mathbb{N}}$ as in Theorem 3.4. Furthermore, let $i_0\in\mathbb{N}$ be such that $q^{k(m+i_0)-\delta}\geq S(n,k,\delta)$ and put $\hat{\mathbb{N}}_{\leq i_0}:=\hat{\mathbb{N}}\cap\{0,1,\ldots,i_0\}$. Then*

$$G_q(n,k,\delta;m)=\max\Big\{d'\in\{1,\ldots,S(n,k,\delta)\}\;\Big|\;\sum_{l=0}^{k(m+i)-\delta-1}\Big\lceil\frac{d'}{q^l}\Big\rceil\leq n(m+i)\text{ for all }i\in\hat{\mathbb{N}}_{\leq i_0}\Big\}. \tag{3.2}$$

Hence the distance of an $(n,k,\delta;m)_q$-code is upper bounded by the number given in (3.2).

We will see in the next section that the Griesmer bound is tight for many sets of parameters.

PROOF: Notice that for $a\geq S(n,k,\delta)$ we have $\lceil\frac{d'}{a}\rceil=1$ since $d'\leq S(n,k,\delta)$. As for (3.2) it suffices to show that whenever d' satisfies the inequality $\sum_{l=0}^{k(m+i)-\delta-1}\lceil\frac{d'}{q^l}\rceil\leq n(m+i)$ for some $i\geq i_0$, then it also satisfies the inequality for $i+1$. But this follows easily from

$$\sum_{l=0}^{k(m+i+1)-\delta-1}\Big\lceil\frac{d'}{q^l}\Big\rceil=\sum_{l=0}^{k(m+i)-\delta-1}\Big\lceil\frac{d'}{q^l}\Big\rceil+\sum_{l=k(m+i)-\delta}^{k(m+i+1)\delta-1}\Big\lceil\frac{d'}{q^l}\Big\rceil\leq n(m+i)+k\leq n(m+i+1).\;\square$$

The finite sets for d' and i in (3.2) are not optimized, but they are good enough for our purposes since they allow for a computation of the Griesmer bound in finitely many steps. Unfortunately, (3.2) does not reveal the block code case where only the index $i=1$ has to be considered according to Theorem 3.1. The consistency of the Griesmer bound for $m=\delta=0$ with that case is guaranteed by the following result.

Proposition 3.6 *Given the parameters n, k, and q. Then*

$$\max\Big\{d'\in\mathbb{N}\;\Big|\;\sum_{l=0}^{ki-1}\Big\lceil\frac{d'}{q^l}\Big\rceil\leq ni\text{ for all }i\in\mathbb{N}\Big\}=\max\Big\{d'\in\mathbb{N}\;\Big|\;\sum_{l=0}^{k-1}\Big\lceil\frac{d'}{q^l}\Big\rceil\leq n\Big\}.$$

PROOF: Let d' be any number satisfying $\sum_{l=0}^{k-1}\lceil\frac{d'}{q^l}\rceil\leq n$. We have to show that d' satisfies the inequalities given on the left hand side for all $i\in\mathbb{N}$. In order to do so, notice that according to Proposition 3.2(a)

$$d'\leq\Big\lfloor\frac{nq^{k-1}(q-1)}{q^k-1}\Big\rfloor\leq\frac{nq^{k-1}}{1+q+\ldots+q^{k-1}}\leq\frac{n}{k}q^{k-1}.$$

But this implies $\frac{d'}{q^l}<\frac{n}{k}$ for all $l\geq k$, thus $\lceil\frac{d'}{q^l}\rceil\leq\frac{n}{k}$ and

$$\sum_{l=0}^{ki-1}\Big\lceil\frac{d'}{q^l}\Big\rceil=\sum_{l=0}^{k-1}\Big\lceil\frac{d'}{q^l}\Big\rceil+\sum_{l=k}^{ki-1}\Big\lceil\frac{d'}{q^l}\Big\rceil\leq n+k(i-1)\frac{n}{k}=ni.$$

This proves the assertion. $\square$

Finally we want to investigate as to how big the field size q has to be in order to allow for an MDS code with parameters $(n,k,\delta)_q$. A first estimate can be achieved by using the Griesmer bound in combination with the generalized Singleton bound.

Theorem 3.7 *Let $\mathcal{C} \subseteq \mathbb{F}((z))^n$ be an $(n,k,\delta;m)_q$-MDS code, thus $d := \mathrm{dist}(\mathcal{C}) = S(n,k,\delta) = (n-k)\big(\lfloor \frac{\delta}{k} \rfloor + 1\big) + \delta + 1$. Then the field size q satisfies*

$$q \geq \begin{cases} \frac{d}{n-k+1}, & \text{if } [k=1] \text{ or } [k>1 \text{ and } km = \delta+1] \\ d, & \text{if } [k>1 \text{ and } km \neq \delta+1]. \end{cases}$$

The estimate above also covers the block code case as given in [14, p. 321].

PROOF: We will consider the various cases separately. In each case we will apply the inequality

$$\frac{d}{q} \leq n(m+i) - d - \sum_{l=2}^{k(m+i)-\delta-1} \Big\lceil \frac{d}{q^l} \Big\rceil, \tag{3.3}$$

which is a simple consequence of the Griesmer bound, to the case $d = S(n,k,\delta)$. Moreover we will make use of the fact that $\lceil \frac{d}{q^l} \rceil \geq 1$ for all $l \in \mathbb{N}$.

$\underline{k=1:}$ In this case $m = \delta$ and $d = n(m+1)$. Since $k(m+i)-\delta-1 = i-1$ Inequality (3.3) gives us

$$\frac{d}{q} \leq n(m+i) - n(m+1) - (i-2) = n(i-1) - i + 2$$

for all $i \geq 2$. This shows $q \geq \frac{d}{n}$ as desired. Using $i = 1$ in the Griesmer bound simply leads to $d \leq n(m+1)$. This is true by assumption and gives no further condition on q.
$\underline{k>1 \text{ and } km = \delta:}$ Now $m = \frac{\delta}{k}$ and thus $d = (n-k)(m+1)+mk+1$. Using $k(m+i)-\delta-1 = ki-1$ we obtain from Inequality (3.3)

$$\frac{d}{q} \leq n(m+i) - (n-k)(m+1) - mk - 1 - (ki-2) = (n-k)(i-1) + 1$$

for all $i \geq 1$. Using $i = 1$ leads to $q \geq d$.
$\underline{k>1 \text{ and } km > \delta:}$ In this case $m = \lfloor \frac{\delta}{k} \rfloor + 1$, see Theorem 3.3(b), and $d = (n-k)m + \delta + 1$. Therefore Inequality (3.3) leads to

$$\frac{d}{q} \leq n(m+i) - (n-k)m - \delta - 1 - \big(k(m+i) - \delta - 2\big) = (n-k)i + 1$$

for all $i \geq 1$. This shows $q \geq \frac{d}{n-k+1}$. In order to finish the proof we have to consider also $i = 0$. In the case $km = \delta + 1$ the Griesmer bound applied to $i = 0$ simply leads to $d \leq nm$, which is true anyway, and no additional condition on q arises. If $km - \delta > 1$ a better bound can be achieved. Since $\lfloor \frac{\delta}{k} \rfloor = m - 1$, we obtain after division with remainder of δ by k an identity of the form $\delta = (m-1)k + r$ where $0 \leq r < k-1$. Thus $d = nm - k + r + 1$ and Inequality (3.3) for $i = 0$ leads to

$$\frac{d}{q} \leq nm - d - \sum_{l=2}^{k-r-1} \Big\lceil \frac{d}{q^l} \Big\rceil \leq k - r - 1 - (k-r-2) = 1,$$

hence $q \geq d$.
This covers all cases, since we always have $km \geq \delta$. □

The proof shows that in general the lower bounds on q are not tight since we have estimated $\lceil \frac{d}{q^l} \rceil$ by 1 for $l \geq 2$ in all cases. For instance, if $(n-k+1)^2 > d$, no $(n,k,\delta;m)_q$-MDS code exists for $q = \frac{d}{n-k+1}$ and $k = 1$ or $km = \delta + 1$. But even if $\lceil \frac{d}{q^l} \rceil = 1$ for all $l \geq 2$ there might

not exist an $(n,k,\delta)_q$-MDS code where q attains the lower bound. The obstacle is that for some $i \in \hat{\mathbb{N}}$ there might not exist an $(n(m+i), k(m+i)-\delta)_q$-block code with the appropriate distance as required by the proof of Theorem 3.4. Since these block codes have to produce a convolutional code in a very specific way, they even have to have some additional structure. We wish to illustrate this by the following example.

Example 3.8 Let $(n,k,\delta) = (3,2,3)$. The generalized Singleton bound is $d := S(3,2,3) = 6$ and the memory of a $(3,2,3)$-MDS code is $m = 2$, see Theorem 3.3(b). From Theorem 3.7 we obtain $q \geq 3$ for the field size. Taking $q = 3$ we have $\lceil \frac{d}{q^2} \rceil = 1$ so that indeed the lower bound for the field size cannot be improved. The existence of a $(3,2,3;2)_3$-MDS code requires the existence of $(3(2+i), 1+2i)_3$-block codes with distance at least 6 for all $i \in \mathbb{N}_0$. Such codes do indeed exist[1]. However, the block codes have to have some additional structure in order to be part of a convolutional code. To see this, let $G \in \mathbb{F}_3[z]^{2\times 3}$ be a minimal generator matrix of the desired convolutional code $\mathcal{C}$. Write

$$G = \begin{bmatrix} g_1 \\ g_2 \end{bmatrix} + z \begin{bmatrix} g_3 \\ g_4 \end{bmatrix} + z^2 \begin{bmatrix} g_5 \\ 0 \end{bmatrix} \text{ where } g_i \in \mathbb{F}_3^3.$$

Recall from the proof of Theorem 3.4 that our arguments are based in particular on the block code $\mathcal{C}_1 := \{(u_1, u_2 + u_3 z)G \mid u_1, u_2, u_3 \in \mathbb{F}_3\}$. Comparing like powers of z one observes that this code is isomorphic to

$$\hat{\mathcal{C}}_1 = \operatorname{im} \begin{bmatrix} g_1 & g_3 & g_5 \\ g_2 & g_4 & 0 \\ 0 & g_2 & g_4 \end{bmatrix} \subseteq \mathbb{F}_3^9.$$

Using elementary row operations on the polynomial matrix G we may assume that the entry of G at the position $(1,1)$ is a constant. Furthermore, after rescaling the columns of G we may assume $g_4 = (1,1,1)$. Finally, due to non-catastrophicity, the entries of g_2 are not all the same and because of $\operatorname{dist}(\hat{\mathcal{C}}_1) = 6$, all nonzero. This gives us (up to block code equivalence) the two options

$$\operatorname{im} \begin{bmatrix} a_1 & a_2 & a_3 & 0 & a_4 & a_5 & 0 & a_6 & a_7 \\ 1 & 1 & 2 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 2 & 1 & 1 & 1 \end{bmatrix} \text{ or } \operatorname{im} \begin{bmatrix} a_1 & a_2 & a_3 & 0 & a_4 & a_5 & 0 & a_6 & a_7 \\ 1 & 2 & 2 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 2 & 1 & 1 & 1 \end{bmatrix}$$

for $\hat{\mathcal{C}}_1$. Going through some tedious calculations one can show that no such code in $\mathbb{F}_3^9$ with distance 6 exists. Hence no $(3,2,3)_3$-MDS convolutional code exists.

In the next section we will give examples of MDS codes over fields $\mathbb{F}_q$ where q attains the lower bound in all cases except for the case $km = \delta + 1$.

4 Examples of Some Optimal Convolutional Codes

In this section we present some convolutional codes with distance reaching the Griesmer bound. To the best of our knowledge it was for most of the parameters, if not all, not known before whether such codes existed.

[1] For small i these codes can be found in tables listing ternary codes. For the general case we wish to thank H.-G. Quebbemann who pointed out to us a construction of such codes for sufficiently large i using direct products of finitely many "short" MDS-codes over $\mathbb{F}_{3^3}$ and mapping them into ternary codes.

In the first column of the tables below the parameters of the given code are listed. In the second column we give the Griesmer bound $g := G_q(n, k, \delta; m)$ for these parameters. The third column gives a code reaching this bound. In all examples the distance of the code has been computed via a program. In each case the code is given by a minimal generator matrix. Thus, in particular all matrices given below are right invertible. In the forth column we present the index of the first column distance that reaches the free distance, cf. (2.4). In the last column we indicate whether the code is a cyclic convolutional code in the sense of Section 5. At the moment this additional structure is not important. We only want to mention that cyclic convolutional codes do not exist for all sets of parameters, in particular the length and the characteristic of the field have to be coprime (just like for block codes). Moreover, the shortest *binary* cyclic convolutional codes with complexity $\delta > 0$ have length $n = 7$ or $n = 15$.

The fields being used in the tables are $\mathbb{F}_2 = \{0, 1\}$, $\mathbb{F}_4 = \{0, 1, \alpha, \alpha^2\}$ where $\alpha^2 + \alpha + 1 = 0$, $\mathbb{F}_8 = \{0, 1, \beta, \ldots, \beta^6\}$ where $\beta^3 + \beta + 1 = 0$, and $\mathbb{F}_{16} = \{0, 1, \gamma, \ldots, \gamma^{14}\}$ where $\gamma^4 + \gamma + 1 = 0$.

The generator matrix $\hat{G}_3$ of the $(15, 4, 12; 3)_2$-code in Table I is given by

$$\hat{G}_3^{\mathsf{T}} = \begin{bmatrix} 1+z^2 & 1+z+z^3 & z+z^2 & 1+z+z^3 \\ 1+z+z^2 & 1+z+z^2+z^3 & 1+z+z^2+z^3 & z \\ 1+z+z^3 & 1+z+z^2 & 1+z+z^2 & 1+z^2+z^3 \\ z & 1+z+z^3 & 1 & 1+z+z^2 \\ z & z^2 & 1+z & 1+z^3 \\ z^2 & z+z^3 & z^3 & 1+z+z^2+z^3 \\ 1+z+z^3 & z^2+z^3 & z+z^2+z^3 & z \\ z^3 & 1+z+z^2 & z+z^3 & z^2 \\ z+z^2+z^3 & z+z^2 & 1+z^3 & z^2+z^3 \\ 1+z+z^2+z^3 & z^2+z^3 & z^2 & 1+z+z^2 \\ 1 & 1 & z+z^2+z^3 & z^2 \\ z^2+z^3 & 1+z & 1 & 0 \\ 1+z & 0 & 1+z^2+z^3 & 1+z^3 \\ z^2+z^3 & 1+z^2+z^3 & z^3 & 1+z+z^3 \\ 1+z^2+z^3 & z^3 & 1+z+z^2 & z+z^2+z^3 \end{bmatrix}.$$

Some additional explanations and remarks will follow the tables.

Table I

$(n,k,\delta;m)_q$	g	code meeting the Griesmer bound	d_z^c cy
$(5,3,4;2)_2$	6	$\begin{bmatrix} 1+z^2 & 1+z & z & 1+z^2 & z+z^2 \\ 1+z & z & 1+z & 1 & z \\ z & 1 & 1+z & 1+z & 1 \end{bmatrix}$ (not even)	7
$(5,2,6;3)_2$	12	$\begin{bmatrix} z^3+z^2+1 & z^2+z & z^3+z+1 & z^2+z & z^3+1 \\ z+1 & z^3+z^2+1 & z^3+z^2 & z^3+z+1 & z^2+z \end{bmatrix}$ (even)	10
$(5,2,6;4)_2$	12	$\begin{bmatrix} 1+z^3+z^4 & 1+z+z^4 & 1+z^3 & 1+z^2+z^3 & z+z^3+z^4 \\ 1+z^2 & 1+z & z^2+z & z^2+z+1 & z^2+z+1 \end{bmatrix}$ (even)	10
$(9,3,1;1)_8$	$8^\bullet$	$\begin{bmatrix} z+1 & z+\beta & z & z+\beta^2 & z+\beta^3 & z+\beta^6 & z+1 & z & z+\beta \\ 1 & \beta^2 & \beta^5 & \beta^6 & \beta^6 & \beta^5 & \beta^2 & 1 & 0 \\ 0 & 1 & \beta^2 & \beta^5 & \beta^6 & \beta^6 & \beta^5 & \beta^2 & 1 \end{bmatrix}$	1
$(3,2,2;1)_5$	$5^\bullet$	$\begin{bmatrix} 2+3z & 3z & 4+4z \\ 4+2z & 1+3z & 2z \end{bmatrix}$	5
$(7,3,3;1)_2$	8	$G_1 = \begin{bmatrix} 1 & z & 1+z & 1+z & 1 & z & 0 \\ z & 1+z & 0 & 1+z & 1 & 1 & z \\ 0 & z & 1 & 0 & 1+z & 1+z & 1+z \end{bmatrix}$ (even)	2 ×
$(7,3,6;2)_2$	12	$G_2 = \begin{bmatrix} 1+z^2 & z+z^2 & 1+z & 1+z & 1+z^2 & z & z^2 \\ z & 1+z+z^2 & 0 & 1+z+z^2 & 1+z^2 & 1+z^2 & z \\ z^2 & z+z^2 & 1+z^2 & 0 & 1+z & 1+z+z^2 & 1+z \end{bmatrix}$ (even)	5 ×
$(7,3,9;3)_2$	16	$\begin{bmatrix} 1+z^2+z^3 & z+z^2 & 1+z+z^3 & 1+z & 1+z^2 & z+z^3 & z^2+z^3 \\ z & 1+z+z^2+z^3 & 0 & 1+z+z^2 & 1+z^2+z^3 & 1+z^2+z^3 & z+z^3 \\ z^2+z^3 & z+z^2 & 1+z^2 & z^3 & 1+z+z^3 & 1+z+z^2+z^3 & 1+z \end{bmatrix}$ (even?)	9 ×
$(7,3,12;4)_2$	20	$\begin{bmatrix} 1+z+z^3+z^4 & 1+z^3+z^4 & 1+z^2 & z+z^2+z^4 & 1+z^2+z^3 & z & z+z^2+z^3+z^4 \\ z^2+z^3 & 1+z+z^2+z^4 & 1+z^4 & 1+z+z^2+z^3+z^4 & z & 1+z+z^3+z^4 & z^2+z^3 \\ z^2+z^4 & z & 1+z+z^3 & 1+z+z^2+z^4 & 1+z^2+z^3+z^4 & z^2+z^3+z^4 & 1+z+z^3 \end{bmatrix}$ (doubly even?)	14×
$(15,4,4;1)_2$	16	$\hat{G}_1 = \begin{bmatrix} z & 0 & z & 1+z & 0 & 0 & 1+z & 1 & 0 & 1 & z & 1+z & 1+z & 1+z & 1 \\ 1 & 0 & z & 0 & 1 & 0 & z & 1+z & 1+z & z & 1 & z & 1 & 1+z & 1+z \\ 1 & 1 & z & z & z & 1+z & 0 & z & 1 & 1+z & z & 1 & 0 & 1+z & 1 \\ 1+z & 1+z & 1 & z & 0 & z & 1+z & 0 & 0 & 1+z & 1 & 0 & 1 & z & 1+z \end{bmatrix}$ (even)	2 ×
$(15,4,8;2)_2$	24	$\hat{G}_2 = \begin{bmatrix} 1+z^2 & 1+z+z^2 & 1+z & z & z^2 & 1+z & 0 & z+z^2 & 1+z+z^2 & 1 & z^2 & 1+z & z^2 & 1+z^2 \\ 1+z & 1+z+z^2 & 1+z+z^2 & 1+z & z^2 & z & z^2 & 1+z+z^2 & z+z^2 & z^2 & 1 & 1+z & 0 & 1+z^2 & 0 \\ z+z^2 & 1+z+z^2 & 1+z+z^2 & 1 & 1+z & 0 & z+z^2 & z & 1 & z^2 & z+z^2 & 1 & 1+z^2 & 0 & 1+z+z^2 \\ 1+z & z & 1+z^2 & 1+z+z^2 & 1 & 1+z+z^2 & z & z^2 & z^2 & 1+z+z^2 & z^2 & 0 & 1 & 1+z & z+z^2 \end{bmatrix}$ (even?)	5 ×
$(15,4,12;3)_2$	32	see $\hat{G}_3$ above, (even?)	×

Table II

$(n,k,\delta;m)_q$	g	code meeting the Griesmer bound	d_i^c	cy
$(3,1,1;1)_4$	6^*	$[\alpha+\alpha z,\ \alpha^2+\alpha z,\ 1+\alpha z]$	2^{**}	×
$(3,1,2;2)_4$	9^*	$[\alpha+\alpha z+z^2,\ \alpha^2+\alpha z+\alpha^2 z^2,\ 1+\alpha z+\alpha z^2]$	5	×
$(3,1,3;3)_4$	$12^{*\bullet}$	$[\alpha+\alpha z+z^2+\alpha^2 z^3,\ \alpha^2+\alpha z+\alpha^2 z^2+z^3,\ 1+\alpha z+\alpha z^2+\alpha z^3]$	7	×
$(3,1,4;4)_4$	14	$[\alpha+\alpha z+z^2+\alpha^2 z^3+\alpha z^4,\ \alpha^2+\alpha z+\alpha^2 z^2+z^3+\alpha z^4,\ 1+\alpha z+\alpha z^2+\alpha z^3+\alpha z^4]$	10	×
$(3,1,5;5)_4$	16	$[\alpha+\alpha z+z^2+\alpha^2 z^3+\alpha z^4+\alpha z^5,\ \alpha^2+\alpha z+\alpha^2 z^2+z^3+\alpha z^4+z^5,\ 1+\alpha z+\alpha z^2+\alpha z^3+\alpha z^4+\alpha^2 z^5]$	11	×
$(5,2,2;1)_4$	8	$\begin{bmatrix} 0 & \alpha+z & \alpha^2+\alpha^2 z & \alpha^2+\alpha^2 z & \alpha+z \\ \alpha+\alpha^2 z & z & \alpha & \alpha^2+z & \alpha^2+\alpha^2 z \end{bmatrix}$	2	×
$(5,2,4;2)_4$	12	$\begin{bmatrix} 0 & \alpha+z+\alpha z^2 & \alpha^2+\alpha^2 z+\alpha^2 z^2 & \alpha^2+\alpha^2 z+\alpha^2 z^2 & \alpha+z+\alpha z^2 \\ \alpha+\alpha^2 z+\alpha z^2 & z+\alpha^2 z^2 & \alpha+\alpha^2 z^2 & \alpha^2+z+\alpha z^2 & \alpha^2+\alpha^2 z \end{bmatrix}$	5	×
$(5,2,6;3)_4$	16	$\begin{bmatrix} 0 & \alpha^2+\alpha^2 z+\alpha z^2+z^3 & 1+\alpha z+\alpha^2 z^2+\alpha^2 z^3 & 1+\alpha z+\alpha^2 z^2+\alpha^2 z^3 & \alpha^2+\alpha^2 z+\alpha z^2+z^3 \\ \alpha^2+\alpha z+\alpha z^2+\alpha^2 z^3 & \alpha^2 z+\alpha^2 z^2+\alpha^2 z^3 & \alpha^2+\alpha^2 z^2+z^3 & 1+\alpha^2 z+\alpha z^2 & 1+\alpha z+z^3 \end{bmatrix}$	9	×
$(3,2,2;1)_{16}$	5^*	$\begin{bmatrix} \gamma^5+\gamma^4 z & \gamma^3+\gamma^8 z & \gamma^9+\gamma^2 z \\ \gamma^9+\gamma^{12} z & \gamma^5+\gamma^{14} z & \gamma^3+\gamma^3 z \end{bmatrix}$	3^{**}	×
$(3,2,3;2)_{16}$	6^*	$\begin{bmatrix} \gamma+\gamma z+z^2 & \gamma^6+\gamma z+\gamma^{10} z^2 & \gamma^{11}+\gamma z+\gamma^5 z^2 \\ 1+z & \gamma^{10}+\gamma^5 z & \gamma^5+\gamma^{10} z \end{bmatrix}$	5	×
$(5,1,1;1)_{16}$	10^*	$[\gamma+\gamma z,\gamma^{13}+\gamma^{10} z,\gamma^{10}+\gamma^4 z,\gamma^7+\gamma^{13} z,\gamma^4+\gamma^7 z]$	2^{**}	×
$(5,1,2;2)_{16}$	15^*	$[\gamma+\gamma^4 z+\gamma z^2,\gamma^7+\gamma z+\gamma^{10} z^2,\gamma^{13}+\gamma^{13} z+\gamma^4 z^2,\gamma^4+\gamma^{10} z+\gamma^{13} z^2,\gamma^{10}+\gamma^7 z+\gamma^7 z^2]$	3^{**}	×
$(5,1,3;3)_{16}$	20^*	$[\gamma+z+\gamma^2 z^2+z^3,\gamma^7+\gamma^{12} z+\gamma^{11} z^2+\gamma^3 z^3,\gamma^{13}+\gamma^9 z+\gamma^5 z^2+\gamma^6 z^3,\gamma^4+\gamma^6 z+\gamma^{14} z^2+\gamma^9 z^3,\gamma^{10}+\gamma^3 z+\gamma^8 z^2+\gamma^{12} z^3]$	5	×
$(5,2,2;1)_{16}$	9^*	$\begin{bmatrix} \gamma+\gamma z & \gamma^{13}+\gamma^{10} z & \gamma^{10}+\gamma^4 z & \gamma^7+\gamma^{13} z & \gamma^4+\gamma^7 z \\ 1+\gamma^5 z & \gamma^3+\gamma^{11} z & \gamma^6+\gamma^2 z & \gamma^9+\gamma^8 z & \gamma^{12}+\gamma^{14} z \end{bmatrix}$	2^{**}	×
$(7,1,1;1)_8$	14^*	$[\beta+\beta z,\beta^3+z,\beta^5+\beta^6 z,1+\beta^5 z,\beta^2+\beta^4 z,\beta^4+\beta^3 z,\beta^6+\beta^2 z]$	2^{**}	×
$(7,1,2;2)_8$	21^*	$[\beta^2+\beta z+z^2,\beta^5+\beta^3 z+\beta^6 z^2,\beta+\beta^5 z+\beta^5 z^2,\beta^4+z+\beta^4 z^2,1+\beta^2 z+\beta^3 z^2,\beta^3+\beta^4 z+\beta^2 z^2,\beta^6+\beta^6 z+\beta z^2]$	3^{**}	×
$(7,1,3;3)_8$	28^*	$[1+\beta z+\beta^6 z^2+z^3,1+\beta^5 z+\beta^5 z^2+\beta^5 z^3,1+\beta^2 z+\beta^4 z^2+\beta^3 z^3,1+\beta^6 z+\beta^3 z^2+\beta z^3,1+\beta^3 z+\beta^2 z^2+\beta^6 z^3,1+z+\beta z^2+\beta^4 z^3,1+\beta^4 z+z^2+\beta^2 z^3]$	5	×
$(7,2,3;2)_8$	14^*	$\begin{bmatrix} 1+z+\beta^4 z^2 & \beta^4+\beta^5 z+\beta^5 z^2 & \beta+\beta^3 z+\beta^6 z^2 & \beta^5+\beta z+z^2 & \beta^2+\beta^6 z+\beta z^2 & \beta^6+\beta^4 z+\beta^2 z^2 & \beta^3+\beta^2 z+\beta^3 z^2 \\ \beta+\beta z & \beta^3+z & \beta^5+\beta^6 z & 1+\beta^5 z & \beta^2+\beta^4 z & \beta^4+\beta^3 z & \beta^6+\beta^2 z \end{bmatrix}$	3	×

Table III

$(n,k,\delta;m)_q$	g	code meeting the Griesmer bound	d_i^c	cy
$(6,3,3;1)_2$	6	columns $1,2,3,5,6,7$ of G_1 (even)	3	
$(6,3,6;2)_2$	10	columns $1,2,4,5,6,7$ of G_2 (even)	3	
$(14,4,4;1)_2$	14	columns 1 – 14 of $\hat{G}_1$ (not even)	3	
$(13,4,4;1)_2$	13	columns $1,2,4$ – 14 of $\hat{G}_1$ (not even)	3	
$(12,4,4;1)_2$	12	columns $1,2,4$ – $12,14$ of $\hat{G}_1$ (even)	3	
$(10,4,4;1)_2$	10	columns $1,2,4,6$ – $11,14$ of $\hat{G}_1$ (even)	4	
$(8,4,4;1)_2$	8	columns $1,2,4,5,8,11,13,14$ of $\hat{G}_1$ (not even)	4	
$(14,4,8;2)_2$	22	columns 2 – 15 of $\hat{G}_2$ (even?)	6	
$(13,4,8;2)_2$	20	columns 1 – $4,7$ – 15 of $\hat{G}_2$ (even?)	6	
$(12,4,8;2)_2$	18	columns $1,2,4,7$ – 15 of $\hat{G}_2$ (not even)	6	
$(10,4,8;2)_2$	16	columns $1,2,4,5,7,8,10,11,13,14$ of $\hat{G}_2$ (even?)	7	
$(8,4,8;2)_2$	12	columns $1,2,6,9,12$ – 15 of $\hat{G}_2$ (even?)	9	

It remains to explain some additional notation of the tables. We also make some further comments illustrating the contents of the tables.

Remark 4.1 (a) A $*$ attached to the bounds in the second column indicate that these numbers are identical to the generalized Singleton bound. Hence the corresponding codes are even MDS codes.

(b) An additional supscript $\bullet$ attached to the bound g indicates that the code is an MDS code where the field size reaches the lower bound of Theorem 3.7. This gives us examples for the three cases $k=1$, $km>\delta+1$, and $km=\delta$. We did not find an example of an $(n,k,\delta)_q$-MDS code where $km=\delta+1$ and $q=\frac{d}{n-k+1}$.

(c) In [4, Prop. 2.3] it has been shown that the jth column distance of an $(n,k,\delta)_q$-code satisfies $d_j^c \le (n-k)(j+1)+1$. From this it follows that the earliest column distance of an MDS code that can reach the free distance has index $M := \lfloor \frac{\delta}{k} \rfloor + \lceil \frac{\delta}{n-k} \rceil$, see [4, Prop. 2.6]. In the same paper an MDS code is called strongly MDS if the Mth column distance is equal to the free distance. We attached a ** to the index of the column distance in the second last column of the tables in order to indicate the strongly MDS codes. As far as we know no upper bound for the column distances is known that also takes the field sizes into account. However, using the estimate $d_j^c \le (n-k)(j+1)+1$ one observes that the $(5,2,2;1)_4$- and the $(9,3,1;1)_8$-code are also optimal in the sense that no code with the same parameters exists where an earlier column distance reaches the free distance. We did not investigate whether any of the other codes is optimal in this sense.

(d) We investigated the binary codes with respect of being even, that is, whether all codewords have even weight. This can be done by computing the weight distribution (see [17] or [9, Sec. 3.10]). Evenness of a code is indicated by an (even) attached to the generator matrix. Since the computation of the full weight distribution is very complex for larger complexity, we did not fully check the binary codes having complexity bigger than 6. In

those cases we checked the weight of codewords associated with message words of small degree. In case this weight is always even we think there is strong evidence that the code is even and attached an (even?) to the generator matrix. In this sense there is also evidence that the $(7, 3, 12; 4)_2$-code is doubly even, that is, all codewords have weight divisible by 4. Further investigation is necessary in order to understand whether (and why) all the binary cyclic convolutional codes of length 7 and 15 are even.

(e) The second and third code of Table I show that a code meeting the Griesmer bound need not have evenly distributed Forney indices. In other words, such a code need not be compact in the sense of Theorem 3.3(b). For both codes in Table I the free distance is attained by the 10th column distance. Only the full weight distribution shows that the code with Forney indices 3, 3 is better than the code with indices 4, 2. The first one has weight distribution

$$W_1(T) = 10T^{12} + 12T^{14} + 71T^{16} + 248T^{18} + 873T^{20} + \ldots,$$

saying that there are 10 molecular codewords of weight 12 and 12 molecular codewords of weight 14, etc. (for the definition of molecular codewords, see [17]; for weight distributions see also [9, Sec. 3.10]). The weight distribution of the second code is

$$W_2(T) = 10T^{12} + 27T^{14} + 99T^{16} + 350T^{18} + 1280T^{20} + \ldots.$$

(f) It is worth being mentioned that the codes with parameters $(7, 3, 3; 1)_2$, $(7, 3, 6; 2)_2$, and $(7, 3, 9; 3)_2$ form a sequence in the sense that if one deletes z^3 (resp. z^2) in the last (resp. second) of the according generator matrices then one obtains the previous code. The same applies to the codes with parameters $(3, 1, 1; 1)_4$, ..., $(3, 1, 5; 5)_4$ as well as to the $(5, 2, 2; 1)_4$- and $(5, 2, 4; 2)_4$-codes.

(g) The codes with parameters $(7, 3, 3; 1)_2$, $(7, 3, 6; 2)_2$, $(15, 4, 4; 1)_2$ and $(15, 4, 8; 2)_2$ are extremely robust against puncturing in the sense of cutting columns of the according generator matrix (this is not puncturing in the sense of [16, Sec. 8]). This way we do not only obtain right invertible matrices again, but even minimal matrices and, by doing this appropriately, codes reaching the Griesmer bound. We have cut one column of the codes of length 7 and up to 7 columns of the codes of length 15. The results are given in Table III. The only cases where we did not get codes reaching the Griesmer bound are for $(11, 4, 4; 1)_2$ and for $(9, 4, 8; 2)_2$. We do not know if for these parameters there exist any codes at all that reach the bound. Since $G_2(9, 4, 4; 1) = 8 = G_2(8, 4, 4; 1)$ and $G_2(11, 4, 8; 2) = 16 = G_2(10, 4, 8; 2)$ we skipped in both cases the bigger length. Puncturing the code of length 7 and memory bigger than 2 did not result in a code meeting the Griesmer bound. We did not puncture the code of length 15 and memory 3.

(h) Consider the $(8, 4, 4; 1)_2$-code given in Table III. There are other codes with exactly these parameters given in the literature. Indeed, in [8] some (doubly-even self-dual) $(8, 4, 4; 1)_2$-codes are presented. Our code is not even, which can easily be seen by writing down the generator matrix. We also computed the weight distribution and obtained

$$\begin{aligned} W(T) =& 11T^8 + 28T^9 + 39T^{10} + 101T^{11} + 206T^{12} + 565T^{13} + 1374T^{14} + 3033T^{15} \\ &+ 7366T^{16} + 16984T^{17} + 40510T^{18} + 95617T^{19} + 22348T^{20} + \ldots, \end{aligned}$$

which is better than the weight distribution of the self-dual code given in [8, Eq. (10)].

5 Cyclic Convolutional Codes

The first two tables of the last section list plenty of optimal codes that we have declared as cyclic. Moreover, they gave rise to further sets of optimal codes as listed in Table III. In this section we want to briefly describe the notion of cyclicity for convolutional codes. The first investigations in this direction have been made in the seventies by Piret [18] and Roos [19]. In both papers it has been shown (with different methods and in different contexts) that cyclicity of convolutional codes must not be understood in the usual sense, i. e. invariance under the cyclic shift, if one wants to go beyond the theory of cyclic block codes (see Theorem 5.2 below). As a consequence, Piret suggested a more complex notion of cyclicity which then has been further generalized by Roos. In both papers some nontrivial examples of cyclic convolutional codes in this new sense are presented along with their distances. All this indicates that the new notion of cyclicity seems to be the appropriate one in the convolutional case. Unfortunately, the papers [18, 19] did not get much attention at that time and the topic came to a halt. Only recently it has been resumed in [5]. Therein, an algebraic theory of cyclic convolutional codes has been established which goes well beyond the results of the seventies. On the one hand it leads to a nice, yet nontrivial, generalization of the theory of cyclic block codes, on the other hand it gives a very powerful toolbox for constructing cyclic convolutional codes. We will now give a very brief description of these results and refer to [5] for the details.

Just like for cyclic block codes we assume from now on that the length n and the field size q are coprime. Let $\mathbb{F} = \mathbb{F}_q$ be a field of size q. Recall that a block code $\mathcal{C} \subseteq \mathbb{F}^n$ is called cyclic if it is invariant under the cyclic shift, i. e.

$$(v_0, \ldots, v_{n-1}) \in \mathcal{C} \Longrightarrow (v_{n-1}, v_0, \ldots, v_{n-2}) \in \mathcal{C} \tag{5.1}$$

for all $(v_0, \ldots, v_{n-1}) \in \mathbb{F}^n$. It is well-known that this is the case if and only if $\mathcal{C}$ is an ideal in the quotient ring

$$A := \mathbb{F}[x]/\langle x^n - 1\rangle = \Big\{ \sum_{i=0}^{n-1} f_i x^i \bmod (x^n - 1) \,\Big|\, f_0, \ldots, f_{n-1} \in \mathbb{F} \Big\}, \tag{5.2}$$

identified with $\mathbb{F}^n$ in the canonical way via

$$\mathfrak{p} : \mathbb{F}^n \longrightarrow A, \quad (v_0, \ldots, v_{n-1}) \longmapsto \sum_{i=0}^{n-1} v_i x^i.$$

At this point it is important to recall that the cyclic shift in $\mathbb{F}^n$ translates into multiplication by x in A, i. e.

$$\mathfrak{p}(v_{n-1}, v_0, \ldots, v_{n-2}) = x\mathfrak{p}(v_0, \ldots, v_{n-1}) \tag{5.3}$$

for all $(v_0, \ldots, v_{n-1}) \in \mathbb{F}^n$. Furthermore, it is well-known that each ideal $I \subseteq A$ is principal, hence there exists some $g \in A$ such that $I = \langle g \rangle$. One can even choose g as a monic divisor of $x^n - 1$, in which case it is usually called the *generator polynomial* of the code $\mathfrak{p}^{-1}(I) \subseteq \mathbb{F}^n$.

It is our aim to extend this structure to the convolutional setting. The most convenient way to do so is by using only the polynomial part $\mathcal{C} \cap \mathbb{F}[z]^n$ of the convolutional code $\mathcal{C} \subseteq \mathbb{F}(\!(z)\!)^n$. Recall from (2.2) that this uniquely determines the full code. Hence imposing some additional structure on the polynomial part (that is, on the generator matrix) will also impose some additional structure on the full code. In Remark 5.6 below we will see from hindsight that one can just as well proceed directly with the full code. The polynomial part of a convolutional

code is always a submodule of the free module $\mathbb{F}[z]^n$. Due to the right invertibility of the generator matrix not every submodule of $\mathbb{F}[z]^n$ arises as polynomial part of a convolutional code. It is easy to see [5, Prop. 2.2] that we have

Remark 5.1 A submodule $\mathcal{S} \subseteq \mathbb{F}[z]^n$ is the polynomial part of some convolutional code if and only if $\mathcal{S}$ is a direct summand of $\mathbb{F}[z]^n$, i.e. $\mathcal{S} \oplus \mathcal{S}' = \mathbb{F}[z]^n$ for some submodule $\mathcal{S}' \subseteq \mathbb{F}[z]^n$.

In order to extend the situation of cyclic block codes to the convolutional setting, we have to replace the vector space $\mathbb{F}^n$ by the free module $\mathbb{F}[z]^n$ and, consequently, the ring A by the polynomial ring

$$A[z] := \Big\{ \sum_{j=0}^{N} z^j a_j \,\Big|\, N \in \mathbb{N}_0,\, a_j \in A \Big\}$$

over A. Then we can extend the map $\mathfrak{p}$ above coefficientwise to polynomials, thus

$$\mathfrak{p} : \mathbb{F}[z]^n \longrightarrow A[z], \quad \sum_{j=0}^{N} z^j v_j \longmapsto \sum_{j=0}^{N} z^j \mathfrak{p}(v_j), \tag{5.4}$$

where, of course, $v_j \in \mathbb{F}^n$ and thus $\mathfrak{p}(v_j) \in A$ for all j. This map is an isomorphism of $\mathbb{F}[z]$-modules. Again, by construction the cyclic shift in $\mathbb{F}[z]^n$ corresponds to multiplication by x in $A[z]$, that is, we have (5.3) for all $(v_0, \ldots, v_{n-1}) \in \mathbb{F}[z]^n$. At this point it is quite natural to call a convolutional code $\mathcal{C} \subseteq \mathbb{F}(\!(z)\!)^n$ cyclic if it is invariant under the cyclic shift, i. e. if (5.1) holds true for all $(v_0, \ldots, v_{n-1}) \in \mathbb{F}(\!(z)\!)^n$. This, however, does not result in any codes other than block codes due to the following result, see [18, Thm. 3.12] and [19, Thm. 6]. An elementary proof can be found at [5, Prop. 2.7].

Theorem 5.2 *Let $\mathcal{C} \subseteq \mathbb{F}(\!(z)\!)^n$ be an (n, k, δ)-convolutional code such that* (5.1) *holds true for all $(v_0, \ldots, v_{n-1}) \in \mathbb{F}[z]^n$. Then $\delta = 0$, hence $\mathcal{C}$ is a block code.*

This result has led Piret [18] to suggest a different notion of cyclicity for convolutional codes. We will present this notion in the slightly more general version as it has been introduced by Roos [19].

In order to do so notice that $\mathbb{F}$ can be regarded as a subfield of the ring A in a natural way. As a consequence, A is an $\mathbb{F}$-algebra, i. e., a ring and a vector space over the field $\mathbb{F}$ and the two structures are compatible. In the sequel the automorphisms of A with respect to this algebra structure will play an important role. Therefore we define

$$\mathrm{Aut}_{\mathbb{F}}(A) := \big\{ \sigma : A \to A \,\big|\, \sigma|_{\mathbb{F}} = \mathrm{id}_{\mathbb{F}},\, \sigma \text{ is bijective, } \sigma(a \dot{+} b) = \sigma(a) \dot{+} \sigma(b) \text{ for all } a, b \in A \big\}.$$

It is clear that each automorphism $\sigma \in \mathrm{Aut}_{\mathbb{F}}(A)$ is uniquely determined by the single value $\sigma(x) \in A$. But not every choice for $\sigma(x)$ determines an automorphism on A. Since x generates the $\mathbb{F}$-algebra A, the same has to be true for $\sigma(x)$ and, more precisely, we obtain for $a \in A$

$$\left.\begin{array}{l} \sigma(x) = a \text{ determines an} \\ \text{automorphism on } A \end{array}\right\} \Longleftrightarrow \left\{\begin{array}{l} 1, a, \ldots, a^{n-1} \text{ are linearly independent over } \mathbb{F} \\ \text{and } a^n = 1. \end{array}\right. \tag{5.5}$$

Of course, $\sigma(x) = x$ determines the identity map on A. It should be mentioned that there is a better way to determine the automorphism group of A by using the fact that the ring is direct product of fields. This is explained in [5, Sec. 3].

The main idea of Piret was to impose a new ring structure on $A[z]$ and to call a code cyclic if it is a left ideal with respect to that ring structure. The new structure is non-commutative and based on an (arbitrarily chosen) automorphism on A. In detail, this looks as follows.

Definition 5.3 Let $\sigma \in \mathrm{Aut}_{\mathbb{F}}(A)$.

(1) On the set $A[z]$ we define addition as usual and multiplication via

$$\sum_{j=0}^{N} z^j a_j \cdot \sum_{l=0}^{M} z^l b_l = \sum_{t=0}^{N+M} z^t \sum_{j+l=t} \sigma^l(a_j) b_l \text{ for all } N,\, M \in \mathbb{N}_0 \text{ and } a_j,\, b_l \in A.$$

This turns $A[z]$ into a non-commutative ring which is denoted by $A[z;\sigma]$.

(2) Consider the map $\mathfrak{p} : \mathbb{F}[z]^n \to A[z;\sigma]$ as in (5.4), where now the images $\mathfrak{p}(v) = \sum_{j=0}^{N} z^j \mathfrak{p}(v_j)$ are regarded as elements of $A[z;\sigma]$. A direct summand $\mathcal{S} \subseteq \mathbb{F}[z]^n$ is said to be *σ-cyclic* if $\mathfrak{p}(\mathcal{S})$ is a left ideal in $A[z;\sigma]$.

(3) A convolutional code $\mathcal{C} \subseteq \mathbb{F}(\!(z)\!)^n$ is said to be *σ-cyclic* if $\mathcal{C} \cap \mathbb{F}[z]^n$ is a σ-cyclic direct summand.

A few comments are in order. First of all, notice that multiplication is determined by the rule

$$az = z\sigma(a) \text{ for all } a \in A \tag{5.6}$$

along with the rules of a (non-commutative) ring. Hence, unless σ is the identity, the indeterminate z does not commute with its coefficients. Consequently, it becomes important to distinguish between left and right coefficients of z. Of course, the coefficients can be moved to either side by applying the rule (5.6) since σ is invertible. Multiplication inside A remains the same as before. Hence A is a commutative subring of $A[z;\sigma]$. Moreover, since $\sigma|_{\mathbb{F}} = \mathrm{id}_{\mathbb{F}}$, the classical polynomial ring $\mathbb{F}[z]$ is a commutative subring of $A[z;\sigma]$, too. As a consequence, $A[z;\sigma]$ is a left and right $\mathbb{F}[z]$-module and the map $\mathfrak{p} : \mathbb{F}[z]^n \to A[z;\sigma]$ is an isomorphism of left $\mathbb{F}[z]$-modules (but not of right $\mathbb{F}[z]$-modules). In the special case where $\sigma = \mathrm{id}_A$ the ring $A[z;\sigma]$ is the classical commutative polynomial ring and we know from Theorem 5.2 that no σ-cyclic convolutional codes with nonzero complexity exist.

Example 5.4 Let us consider the case where $\mathbb{F} = \mathbb{F}_2$ and $n = 7$. Thus $A = \mathbb{F}[x]/\langle x^7 - 1 \rangle$. Using (5.5) one obtains 18 automorphisms, also listed at [19, p. 680, Table II] (containing one typo: the last element of that table has to be $x^2 + x^3 + x^4 + x^5 + x^6$ rather than $x + x^3 + x^4 + x^5 + x^6$).

Let us choose the automorphism $\sigma \in \mathrm{Aut}_{\mathbb{F}}(A)$ defined by $\sigma(x) = x^5$. Furthermore, we consider the polynomial

$$g := 1 + x^2 + x^3 + x^4 + z(x + x^2 + x^3 + x^5) \in A[z;\sigma]$$

and denote by ${}^{\bullet}\langle\, g \,\rangle := \{fg \mid f \in A[z;\sigma]\}$ the left ideal generated by g in $A[z;\sigma]$. Moreover, put $\mathcal{S} := \mathfrak{p}^{-1}({}^{\bullet}\langle\, g \,\rangle) \subseteq \mathbb{F}[z]^7$. We will show now that $\mathcal{S}$ is a direct summand of $\mathbb{F}[z]^7$, hence $\mathcal{S} = \mathcal{C} \cap \mathbb{F}[z]^7$ for some convolutional code $\mathcal{C} \subseteq \mathbb{F}(\!(z)\!)^7$, see Remark 5.1. In order to do so we first notice that

$${}^{\bullet}\langle\, g \,\rangle = \mathrm{span}_{\mathbb{F}[z]}\{g,\, xg, \ldots, x^6 g\}$$

and therefore

$$\mathcal{S} = \left\{uM \,\middle|\, u \in \mathbb{F}[z]^7\right\} \text{ where } M = \begin{bmatrix} \mathfrak{p}^{-1}(g) \\ \mathfrak{p}^{-1}(xg) \\ \vdots \\ \mathfrak{p}^{-1}(x^6 g) \end{bmatrix}.$$

Thus we have to compute $x^i g$ for $i = 1, \ldots, 6$. Using the multiplication rule in (5.6) we obtain

$$\begin{aligned} xg &= x + x^3 + x^4 + x^5 + z(1 + x + x^3 + x^6), \\ x^2 g &= x^2 + x^4 + x^5 + x^6 + z(x + x^4 + x^5 + x^6), \\ x^3 g &= 1 + x^3 + x^5 + x^6 + z(x^2 + x^3 + x^4 + x^6) \\ &= g + x^2 g. \end{aligned}$$

Since $x^3 g$ is in the $\mathbb{F}$-span of the previous elements, we obtain ${}^{\bullet}\langle\, g \,\rangle = \operatorname{span}_{\mathbb{F}[z]}\{g, xg, x^2 g\}$ and, since $\mathfrak{p}$ is an isomorphism,

$$\mathcal{S} = \{uG \,|\, u \in \mathbb{F}[z]^3\},$$

where

$$G = \begin{bmatrix} \mathfrak{p}^{-1}(g) \\ \mathfrak{p}^{-1}(xg) \\ \mathfrak{p}^{-1}(x^2 g) \end{bmatrix} = \begin{bmatrix} 1 & z & 1+z & 1+z & 1 & z & 0 \\ z & 1+z & 0 & 1+z & 1 & 1 & z \\ 0 & z & 1 & 0 & 1+z & 1+z & 1+z \end{bmatrix}.$$

One can easily check that the matrix G is right invertible. Hence $\mathcal{S}$ is indeed a direct summand of $\mathbb{F}[z]^7$ and thus we have obtained a σ-cyclic convolutional code $\mathcal{C} = \operatorname{im} G \subseteq \mathbb{F}(\!(z)\!)^7$. This is exactly the $(7, 3, 3; 1)_2$-code given in Table I of the last section.

The other cyclic convolutional codes in Tables I and II are obtained in a similar way. Since the underlying automorphism cannot easily be read off from the generator matrix of a cyclic convolutional code we will, for sake of completeness, present them explicitly in the following table. All those codes come from principal left ideals in $A[z; \sigma]$ and, except for the codes with parameters $(3, 2, 3; 2)_{16}$, $(5, 2, 2; 1)_{16}$, $(7, 2, 3; 2)_8$, the generator polynomial can be recovered from the given data by applying the map $\mathfrak{p}$ to the first row of the respective generator matrix. The generator matrices of the remaining three codes are built in a slightly different way. In those cases each row of the given matrix generates a 1-dimensional cyclic code and thus each of those three codes is the direct sum of two 1-dimensional cyclic codes. In each case a generator polynomial of the associated principal left ideal is obtained by applying $\mathfrak{p}$ to the sum of the two rows of the respective generator matrix.

Table IV

$(n, k, \delta; m)_q$-code of Tables I and II	automorphism given by
$(7, 3, 3m; m)_2$, $m = 1, \ldots, 4$	$\sigma(x) = x^5$
$(15, 4, 4; 1)_2$	$\sigma(x) = x + x^7 + x^{10}$
$(15, 4, 4m; m)_2$, $m = 2, 3$	$\sigma(x) = x^3 + x^5 + x^7 + x^{10} + x^{12} + x^{13} + x^{14}$
$(3, 1, \delta; \delta)_4$, $\delta = 1, \ldots, 5$	$\sigma(x) = \alpha^2 x$
$(5, 2, 2m; m)_4$, $m = 1, 2, 3$	$\sigma(x) = x^2$
$(3, 2, 2; 1)_{16}$ and $(3, 2, 3; 2)_{16}$	$\sigma(x) = \gamma^{10} x$
$(5, 1, \delta; \delta)_{16}$, $\delta = 1, 2, 3$ and $(5, 2, 2; 1)_{16}$	$\sigma(x) = x^3$
$(7, 1, \delta; \delta)_8$, $\delta = 1, 2$ and $(7, 2, 3; 2)_8$	$\sigma(x) = x^5$
$(7, 1, 3; 3)_8$	$\sigma(x) = \beta x + \beta x^2 + \beta^3 x^3 + \beta^3 x^4 + \beta^3 x^5 + \beta^2 x^6$

The fact that all the cyclic convolutional codes above come from principal left ideals in $A[z; \sigma]$ is not a restriction since we have the following important result.

Theorem 5.5 *Let $\sigma \in \mathrm{Aut}_{\mathbb{F}}(A)$. If $\mathcal{S} \subseteq \mathbb{F}[z]^n$ is a σ-cyclic direct summand, then $\mathfrak{p}(\mathcal{S})$ is a principal left ideal of $A[z;\sigma]$, that is, there exists some polynomial $g \in A[z;\sigma]$ such that $\mathfrak{p}(\mathcal{S}) = {}^{\bullet}\langle\, g \,\rangle$. We call g a generator polynomial of both $\mathcal{S}$ and the σ-cyclic convolutional code $\mathcal{C} \subseteq \mathbb{F}(\!(z)\!)^n$ determined by $\mathcal{S}$, see Remark 5.1 and* (2.2).

The generator polynomial of a σ-cyclic convolutional code can be translated into vector notation and leads to a generalized circulant matrix. This looks as follows. Let $\mathcal{S} \subseteq \mathbb{F}[z]^n$ be a σ-cyclic direct summand and let $\mathfrak{p}(\mathcal{S}) = {}^{\bullet}\langle\, g \,\rangle$. Define

$$\mathcal{M}^{\sigma}(g) = \begin{bmatrix} \mathfrak{p}^{-1}(g) \\ \mathfrak{p}^{-1}(xg) \\ \vdots \\ \mathfrak{p}^{-1}(x^{n-1}g) \end{bmatrix} \in \mathbb{F}[z]^{n\times n}.$$

Then it is easy to see that $\mathfrak{p}\big(u\mathcal{M}^{\sigma}(g)\big) = \mathfrak{p}(u)g$ for all $u \in \mathbb{F}[z]^n$ (see [5, Prop. 6.8(b)]) and therefore, $\mathcal{S} = \big\{u\mathcal{M}^{\sigma}(g) \,\big|\, u \in \mathbb{F}[z]^n\big\}$. We call $\mathcal{M}^{\sigma}(g)$ the σ-circulant associated with g.

Remark 5.6 Using the identities above we can now easily see that σ-cyclic structure can also be considered without restricting to the polynomial part. Just like the polynomial ring $A[z]$ we can turn the set $A(\!(z)\!)$ of formal Laurent series over A into a non-commutative ring by defining addition as usual and multiplication via (5.6). We will denote the ring obtained this way by $A(\!(z;\sigma)\!)$. Furthermore, we can extend the map $\mathfrak{p}$ to Laurent series in the canonical way, see also (5.4). Then one can easily show that just like in the polynomial case

$$\mathfrak{p}\big(u\mathcal{M}^{\sigma}(g)\big) = \mathfrak{p}(u)g \text{ for all } u \in \mathbb{F}(\!(z)\!)^n$$

for each $g \in A[z;\sigma]$. Using the fact that a code $\mathcal{C} \subseteq \mathbb{F}(\!(z)\!)^n$ is uniquely determined by its polynomial part (see (2.2)), and that the latter is a principal left ideal in $A[z;\sigma]$ due to Theorem 5.5, one can now derive the equivalence

$$\mathcal{C} \subseteq \mathbb{F}(\!(z)\!)^n \text{ is } \sigma\text{-cyclic} \iff \mathfrak{p}(\mathcal{C}) \text{ is a left ideal in } A(\!(z;\sigma)\!).$$

Moreover, if $\mathcal{C}$ is σ-cyclic, a generator polynomial of the ideal $\mathfrak{p}(\mathcal{C} \cap \mathbb{F}[z]^n)$ in $A[z;\sigma]$ is also a principal generator of the ideal $\mathfrak{p}(\mathcal{C})$ in $A(\!(z;\sigma)\!)$. This justifies to call g a generator polynomial of the full code $\mathcal{C}$ as we did in Theorem 5.5.

At this point the question arises as to how a (right invertible) generator matrix can be obtained from the σ-circulant $\mathcal{M}^{\sigma}(g)$. Notice that in Example 5.4 the generator matrix of the code is simply given by the first three rows of the circulant. This is indeed in general the case, but requires a careful choice of the generator polynomial g of the code. Recall that, due to zero divisors in $A[z;\sigma]$, the generators of a principal left ideal, are highly non unique. The careful choice of the generator polynomial is based on a Gröbner basis theory that can be established in the non-commutative polynomial ring $A[z;\sigma]$. This is a type of reduction procedure resulting in unique generating sets of left ideals which in turn produce very powerful σ-circulants. The details of this theory goes beyond the scope of this paper and we refer the reader to [5] for the details, in particular to [5, Thm. 7.8, Thm. 7.18]. Therein it has been shown that a reduced generator polynomial also reflects the parameters of the code, i. e., the dimension and the complexity, and even leads to a minimal generator matrix through σ-circulants. Only with these results it becomes clear that cyclic convolutional codes can have only very specific parameters (length, dimension, and complexity) depending on the

chosen field $\mathbb{F}_q$. Furthermore, the notions of parity check polynomial and associated parity check matrix have been discussed in detail in [5], leading to a generalization of the block code situation.

As for the cyclic codes of the last section we only would like to mention that their generator polynomials obtained as explained right before Table IV are all reduced in the sense above.

So far we do not have any estimates for the distance of a cyclic convolutional code in terms of its (reduced) generator polynomial and the chosen automorphism. The examples given in the last section have been found simply by trying some promising reduced generator polynomials (using the algebraic theory of [5]). Except for the puncturing in Table III we did not perform a systematic search for optimal codes.

Conclusion

In this paper we gave many examples of cyclic convolutional codes that all reach the Griesmer bound. The examples indicate that this class of convolutional codes promises to contain many excellent codes and therefore deserves further investigation. As one of the next steps the relation between the (reduced) generator polynomial and the automorphism on the one hand and the distance on the other hand should be investigated in detail.

References

[1] A. Betten and other. *Codierungstheorie: Konstruktion und Anwendung linearer Codes*. Springer, Berlin, 1998.

[2] G. D. Forney Jr. Convolutional codes I: Algebraic structure. *IEEE Trans. Inform. Theory*, 16:720–738, 1970. (see also corrections in *IEEE Trans. Inf. Theory*, vol. 17,1971, p. 360).

[3] G. D. Forney Jr. Minimal bases of rational vector spaces, with applications to multivariable linear systems. *SIAM J. on Contr.*, 13:493–520, 1975.

[4] H. Gluesing-Luerssen, J. Rosenthal, and R. Smarandache. Strongly MDS convolutional codes. 2003. Submitted. Available at http://front.math.ucdavis.edu/ with ID-number RA/0303254.

[5] H. Gluesing-Luerssen and W. Schmale. On cyclic convolutional codes. Preprint 2002. Submitted. Available at http://front.math.ucdavis.edu/ with ID-number RA/0211040.

[6] H. Gluesing-Luerssen, W. Schmale, and M. Striha. Some small cyclic convolutional codes. In *Electronic Proceedings of the 15th International Symposium on the Mathematical Theory of Networks and Systems*, Notre Dame, IN (USA), 2002. (8 pages).

[7] J. A. Heller. Short constraint length convolutional codes. Jet Propulsion Lab., California Inst. Technol., Pasadena, Space Programs Summary 37–54, 3:171–177.

[8] R. Johannesson, P. Ståhl, and E. Wittenmark. A note on type II convolutional codes. *IEEE Trans. Inform. Theory*, IT-46:1510–1514, 2000.

[9] R. Johannesson and K. S. Zigangirov. *Fundamentals of Convolutional Coding*. IEEE Press, New York, 1999.

[10] J. Justesen. New convolutional code constructions and a class of asymptotically good time-varying codes. *IEEE Trans. Inform. Theory*, IT-19:220–225, 1973.

[11] J. Justesen. Algebraic construction of rate $1/\nu$ convolutional codes. *IEEE Trans. Inform. Theory*, IT-21:577–580, 1975.

[12] K. J. Larsen. Short convolutional codes with maximal free distance for rates 1/2, 1/3, and 1/4. *IEEE Trans. Inform. Theory*, IT-19:371–372, 1973.

[13] J. Lint. *Introduction to Coding Theory*. Springer, 3. edition, 1999.

[14] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland, 1977.

[15] J. L. Massey, D. J. Costello, and J. Justesen. Polynomial weights and code constructions. *IEEE Trans. Inform. Theory*, IT-19:101–110, 1973.

[16] R. J. McEliece. The algebraic theory of convolutional codes. In V. Pless and W. Huffman, editors, *Handbook of Coding Theory, Vol. 1*, pages 1065–1138. Elsevier, Amsterdam, 1998.

[17] R. J. McEliece. How to compute weight enumerators for convolutional codes. In M. Darnell and B. Honory, editors, *Communications and Coding (P. G. Farrell 60th birthday celebration)*, pages 121–141. Wiley, New York, 1998.

[18] P. Piret. Structure and constructions of cyclic convolutional codes. *IEEE Trans. Inform. Theory*, 22:147–155, 1976.

[19] C. Roos. On the structure of convolutional and cyclic convolutional codes. *IEEE Trans. Inform. Theory*, 25:676–683, 1979.

[20] J. Rosenthal. Connections between linear systems and convolutional codes. In B. Marcus and J. Rosenthal, editors, *Codes, Systems, and Graphical Models*, pages 39–66. Springer, Berlin, 2001.

[21] J. Rosenthal, J. M. Schumacher, and E. V. York. On behaviors and convolutional codes. *IEEE Trans. Inform. Theory*, 42:1881–1891, 1996.

[22] J. Rosenthal and R. Smarandache. Maximum distance separable convolutional codes. *Appl. Algebra Engrg. Comm. Comput.*, 10:15–32, 1999.

[23] R. Smarandache, H. Gluesing-Luerssen, and J. Rosenthal. Constructions of MDS-convolutional codes. *IEEE Trans. Inform. Theory*, 47(5):2045–2049, 2001.