Progress in Systems and Control Theory, Vol. 25
©1999 Birkhäuser Verlag Basel/Switzerland

An Algebraic Decoding Algorithm for Convolutional Codes

Joachim Rosenthal[1]

1 Introduction

The class of convolutional codes generalizes the class of linear block codes in a natural way. The construction of convolutional codes which have a large free distance and which come with an efficient decoding algorithm is a major task. Contrary to the situation of linear block codes there exists only very few algebraic construction of convolutional codes.

It is the purpose of this article to introduce a new iterative algebraic decoding algorithm which is capable of decoding convolutional codes which have a certain underlying algebraic structure. The algorithm exploits the algebraic structure of the convolutional code and it achieves its best performance if some naturally associated block codes can be efficiently decoded in an algebraic manner.

In order to achieve this goal we will work with a classical state space description of a so called systematic encoder. Using this description we will derive a general procedure which will allow one to extend known decoding algorithms for block codes (like e.g. the Berlekamp Massey algorithm) to convolutional codes.

In the coding literature there exist several decoding algorithms for convolutional codes. Maybe the most prominent one is the Viterbi decoding algorithm which applies the principle of dynamic programming to compute the transmitted message sequence. It was shown by Forney [6] that this algorithm computes the message sequence in a maximum likelihood fashion. The disadvantage of this algorithm is that it becomes computationally infeasible if the degree of the encoder is larger than 20. On the side of the Viterbi algorithm there are several sub-optimal algorithms and we would like to mention Massey's threshold decoding algorithm [9], the sequential decoding algorithm and the feedback decoding algorithm [7, 8, 12]. More recently there has been a significant interest in some iterative decoding algorithms in connection with the decoding of low density parity check codes and other codes defined on general graphs and we refer to [17, 20].

The iterative decoding algorithm which we will present in this paper seems to be different from above ideas. Indeed the algorithm iteratively computes the state vector x_t inside the trellis diagram (see [7, 8]) by making use of the algebraic structure of the convolutional code.

[1]Supported in part by NSF grant DMS-96-10389.

Once a state x_τ has been correctly computed we will show how to compute in an algebraic manner a new state vector $x_{\tau+\Theta}$, where Θ is a positive integer which depends on the underlying code. Once $x_{\tau+\Theta}$ is computed all code words between time τ and time $\tau+\Theta$ are generally computed through an algebraic decoding scheme.

Similarly to the known algebraic decoding algorithms for block codes it is required that the convolutional code has a certain algebraic structure. In this way the algorithm cannot be applied efficiently to arbitrary convolutional codes. On the other hand if the convolutional code is of Reed Solomon type (see [15]) or of BCH type (see [16, 21]) then the algorithm is capable of decoding convolutional codes in situations where the Viterbi decoding algorithm would not be feasible because of complexity considerations.

The paper is structured as follows: In the next section we summarize some basic notions for block codes and convolutional codes. Emphasis will be on the state space representation for a systematic encoder. In Section 3 we first provide exact conditions on the convolutional code and on the transmitted error pattern which guarantee that the iterative decoding algorithm as presented in Section 4 does compute the transmitted message. In Section 5 we address issues of complexity and we describe two variations where we expect the algorithm to perform very efficiently. In Section 6 we will show that the Berlekamp Massey algorithm or any of its recent improvements (see e.g. [3]) can be invoked to iteratively decode the Reed Solomon and BCH type convolutional codes as presented in [15, 16, 21].

2 Convolutional Codes and their State Space Description

In this section we will provide a short tutorial on block codes and convolutional codes. More details on our state space approach are given in [11, 13, 15, 16, 21]. Comprehensive textbooks on convolutional codes are [7, 8, 12].

Let $\mathbb{F}=\mathbb{F}_q$ be the Galois field of q elements. If

$$\varphi:\ \mathbb{F}^k \longrightarrow \mathbb{F}^n$$

is a monomorphism we say that $\mathcal{C}:=\mathrm{im}(\varphi)\subset\mathbb{F}^n$ is a linear block code and φ is an encoder. Let G be an $n\times k$ matrix representing the linear map φ. The encoding process given by φ is then described by

$$m \longmapsto v = Gm.$$

One says G is a generator matrix for the code $\mathcal{C}$, $m\in\mathbb{F}^k$ is a message vector and $v\in\mathbb{F}^n$ is a code vector. If S is a $k\times k$ invertible matrix then G and

$\tilde{G} := GS$ generate the same code and we say that G and $\tilde{G}$ are equivalent encoders.

We say G is a *systematic encoder* if G has the particular form $\begin{pmatrix} Y \\ I_k \end{pmatrix}$, where I_k is the $k \times k$ identity matrix and Y is a matrix of size $(n-k) \times k$. A systematic encoder has the property that k message symbols $m \in \mathbb{F}^k$ will be transmitted (in an 'unencoded manner') together with $(n-k)$ *parity check symbols* $y = Ym \in \mathbb{F}^{n-k}$.

If the transmitted data has more than k symbols it will be necessary to break down the data into several message blocks. Let $m_0, \ldots, m_\gamma$ be $\gamma + 1$ blocks of messages to be transmitted. If one introduces the polynomial vectors

$$m(z) := \sum_{i=0}^{\gamma} m_i z^i \in \mathbb{F}^k[z] \quad \text{and} \quad v(z) := \sum_{i=0}^{\gamma} v_i z^i \in \mathbb{F}^n[z]$$

then the total encoding scheme

$$m_i \longmapsto v_i = Gm_i, \ i = 0, \ldots, \gamma$$

can be compactly described through the module homomorphism

$$\hat{\varphi} : \ \mathbb{F}^k[z] \longrightarrow \mathbb{F}^n[z], \quad m(z) \longmapsto v(z) = Gm(z).$$

It was the idea of Elias [2] to replace in above encoding scheme the generator matrix G with a generator matrix $G(z)$ and to allow in this way general module homomorphisms as encoding schemes.

Using this point of view we define a convolutional code $\mathcal{C}$ as a $\mathbb{F}[z]$ submodule of $\mathbb{F}^n[z]$. If $V(z)$ is a $k \times k$ unimodular matrix then the encoders $G(z)$ and $\tilde{G}(z) := G(z)V(z)$ define the same convolutional code and we will say that $G(z)$ and $\tilde{G}(z)$ are equivalent encoders. Without loss of generality we can therefore assume that $G(z)$ is in column proper form with column degree $\nu_1 \geq \cdots \geq \nu_k$.

The integer $\delta := \nu_1 + \cdots + \nu_k$ is an invariant of the code (module) $\mathcal{C} \subset \mathbb{F}^n[z]$. We call δ the degree (or complexity) of the code $\mathcal{C}$. Convolutional codes of degree $\delta = 0$ correspond in this way to linear block codes.

Remark 2.1 In the coding literature [5, 7, 12] a convolutional code is often defined as a linear subspace of $\mathcal{F}^n$, where $\mathcal{F} := \mathbb{F}((z))$ is the field of formal Laurent series. If one takes this approach the column degrees are no more invariants of the code and the degree δ is hence also not an invariant of the code but rather a property of the particular encoder. This is one reason why we consider the presented module theoretic approach as appealing. In the same time there seems to exist no practical necessity to have a framework for messages of infinite length.

Remark 2.2 A module theoretic approach to convolutional codes was introduced by Fornasini and Valcher [4, 18] in the context of two dimensional codes. If one works with multidimensional codes then it becomes very difficult if one works with the field of formal Laurent series $\mathbb{F}((z_1, \dots, z_m))$. In [4, 18] Fornasini and Valcher define a code as a submodule of R^n, where $R = \mathbb{F}[z_1, z_2, z_1^{-1}, z_2^{-1}]$ is the ring of Laurent polynomials in the variables z_1, z_2. In this way codes become dual to complete linear behaviors defined on $\mathbb{Z} \times \mathbb{Z}$. Weiner [19] defines a convolutional code as a submodule of R^n, where R is the polynomial ring $R = \mathbb{F}[z_1, \dots, z_m]$. In this framework codes are dual to linear complete behaviors defined on $\mathbb{N}^m$.

Since submodules of $\mathbb{F}^n[z]$, i.e. convolutional codes, are dual to linear complete behaviors they have natural state space descriptions. In the sequel we follow [15, 16] and explain this relation.

Partition the generator matrix $G(z)$ into $G(z) = \begin{pmatrix} Y(z) \\ U(z) \end{pmatrix}$, where $U(z)$ is of size $k \times k$ and $Y(z)$ is of size $(n-k) \times k$. For simplicity assume that $\deg \det U(z) = \delta$, the degree of the encoder $G(z)$. Let $X(z)$ be a basis matrix of size ν (compare with [15, 16]). Then one has the result:

Lemma 2.3 *There exist matrices $A \in \mathbb{F}^{\delta \times \delta}$, $B \in \mathbb{F}^{\delta \times k}$, $C \in \mathbb{F}^{(n-k) \times \delta}$, and $D \in \mathbb{F}^{(n-k) \times k}$ such that*

$$\ker \begin{pmatrix} zI - A & 0 & -B \\ -C & I & -D \end{pmatrix} = \operatorname{im} \begin{pmatrix} X(z) \\ Y(z) \\ U(z) \end{pmatrix}. \tag{2.1}$$

In particular $v(z) = \begin{pmatrix} y(z) \\ u(z) \end{pmatrix} \in \mathbb{F}^n[z]$ is a code word if and only if there is a polynomial vector $x(z) \in \mathbb{F}^\delta[z]$ with

$$\begin{pmatrix} zI - A & 0 & -B \\ -C & I & -D \end{pmatrix} \begin{pmatrix} x(z) \\ y(z) \\ u(z) \end{pmatrix} = 0. \tag{2.2}$$

Remark 2.4 One immediately verifies that the matrices A, B, C, D form a realization for the transfer function $Y(z)U(z)^{-1}$, i.e. one has the relation $Y(z)U(z)^{-1} = C(zI - A)^{-1}B + D$. If the high order coefficient matrix of $U(z)$ is the identity matrix then it is possible to compute the matrices A, B, C, D 'by inspection' [14].

It is possible to give (2.2) a dynamical interpretation. For this let

$$x(z) = x_0 z^\gamma + x_1 z^{\gamma-1} + \ldots + x_\gamma;\ x_t \in \mathbb{F}^\delta, t = 0, \ldots, \gamma,$$

$$u(z) = u_0 z^\gamma + u_1 z^{\gamma-1} + \ldots + u_\gamma;\ u_t \in \mathbb{F}^k, t = 0, \ldots, \gamma,$$

and let

$$y(z) = y_0 z^\gamma + y_1 z^{\gamma-1} + \ldots + y_\gamma;\ y_t \in \mathbb{F}^{n-k}, t = 0, \ldots, \gamma.$$

Then one verifies that (2.2) is equivalent with:

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t, \\ y_t &= Cx_t + Du_t, \\ v_t &= \begin{pmatrix} y_t \\ u_t \end{pmatrix},\ x_0 = 0,\ x_{\gamma+1} = 0. \end{aligned} \tag{2.3}$$

In these equations the sequence of vectors u_t represents the *message vectors*, the sequence of y_t represents the *parity vectors* and the sequence of v_t represents the set of *code vectors.* The equations in (2.3) define the state space realization of the systematic convolutional encoder $\begin{pmatrix} Y(z)U(z)^{-1} \\ I_k \end{pmatrix}$.

Remark 2.5 In the coding literature [5, 10, 11] one often finds a state space description, where the message words m_i are the inputs and the code words v_i are the outputs. Such an A, B, C, D representation is related to the generator matrix $G(z)$ via the relation $G(z^{-1}) = C(zI - A)^{-1}B + D$. The state space realization (2.3) is different.

Assume that the encoder is at state x_τ. Using (2.3) we immediately derive an algebraic dependence between the message vectors u_t and the parity check vectors y_t (compare with [15, 16, 21]):

Proposition 2.6 (Local Description of Trajectories) *Let $\tau, \gamma \in \mathbb{Z}_+$ be positive integers with $\tau < \gamma$. Assume that the encoder is at state x_τ at time $t = \tau$. Then any code sequence $\left\{ \binom{y_t}{u_t} \right\}_{t \geq 0}$ governed by the dynamical system* (2.3) *must satisfy:*

$$\begin{pmatrix} y_\tau \\ y_{\tau+1} \\ \vdots \\ \vdots \\ y_\gamma \end{pmatrix} = \begin{pmatrix} C \\ CA \\ \vdots \\ \vdots \\ CA^{\gamma-\tau} \end{pmatrix} x_\tau$$

$$+ \begin{pmatrix} D & 0 & \cdots & & 0 \\ CB & D & \ddots & & \vdots \\ CAB & CB & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & 0 \\ CA^{\gamma-\tau-1}B & CA^{\gamma-\tau-2}B & \cdots & CB & D \end{pmatrix} \begin{pmatrix} u_\tau \\ u_{\tau+1} \\ \vdots \\ \vdots \\ u_\gamma \end{pmatrix}.$$

Moreover the evolution of the state vector x_t is given over time as:

$$x_t = A^{t-\tau}x_\tau + \left(A^{t-\tau-1}B \ \dots \ B\right)\begin{pmatrix} u_\tau \\ \vdots \\ u_{t-1} \end{pmatrix}; \ t = \tau+1, \tau+2, \dots, \gamma+1. \tag{2.4}$$

In this paper we will mainly use the code description of Proposition 2.6 to arrive at an iterative decoding algorithm of the convolutional code. As it turns out it is possible to construct the matrices A, B, C in a way which will allow one to use iteratively known decoding algorithms for block codes to arrive at the decoding of the convolutional code.

3 Basic Assumptions and Main Results

The decoding task is as follows: Assume a sequence of code words $\{v_t\}_{t\geq 0} = \left\{\binom{y_t}{u_t}\right\}_{t\geq 0}$ was sent and the sequence

$$\{\hat{v}_t\}_{t\geq 0} = \left\{\begin{pmatrix} \hat{y}_t \\ \hat{u}_t \end{pmatrix}\right\}_{t\geq 0}$$

has been received. The decoding problem then asks for the minimization of the error

$$\text{error} := \min_{\{v_t\}\in\mathcal{C}} \sum_{t=0}^{\infty} \text{dist}\,(v_t, \hat{v}_t) = \min\left(\sum_{t=0}^{\infty}(\text{dist}\,(u_t,\hat{u}_t) + \text{dist}\,(y_t,\hat{y}_t))\right), \tag{3.1}$$

where 'dist' does denote the usual Hamming distance between two vectors, i.e. dist $(v, \hat{v})$ is equal to the number of coordinates where v and $\hat{v}$ differ. If no transmission error did occur then $\{\hat{v}_t\}_{t\geq 0}$ is a valid trajectory and the error value in (3.1) is zero. If $\{\hat{v}_t\}_{t\geq 0}$ is not a valid trajectory then the decoding task asks for the computation of the 'nearest trajectory' with respect to the Hamming metric. The decoding problem has therefore the characteristic of a discrete tracking problem. The difficulty lies in the fact that the Hamming metric is not induced by a positive quadratic form and it is therefore not possible to apply standard techniques from LQ theory immediately.

Remark 3.1 If the transmission is done over the 'Gaussian channel' then the natural metric on $\mathbb{F}^n$ is not the Hamming metric but rather a metric which is induced by the Euclidean metric through some modulation scheme. The received signals are in this situation some points in Euclidean space and the decoding task asks for the minimization of the error (3.1) which can be any positive real number. Even in this situation standard methods used in the study of the linear quadratic regulator problem cannot be applied. The basic difficulty comes this time from the fact that the set of code trajectories is $\mathbb{F}$ linear but not $\mathbb{R}$ linear. In either case it seems that decoding of general convolutional codes is in terms of computational complexity a 'hard' problem.

In the sequel we will work with the Hamming metric and we will approach the decoding task by combining ideas used in the decoding of linear block codes and systems theoretic descriptions such as the one given in Proposition 2.6. Let

$$\left\{ \begin{pmatrix} f_t \\ e_t \end{pmatrix} \right\}_{t \geq 0} := \left\{ \begin{pmatrix} \hat{y}_t - y_t \\ \hat{u}_t - u_t \end{pmatrix} \right\}_{t \geq 0} \tag{3.2}$$

be the sequence of errors.

Assumption 3.2 Consider a convolutional code $\mathcal{C}$ described by the matrices A, B, C, D having sizes $\delta \times \delta$, $\delta \times k$, $(n-k) \times \delta$ and $(n-k) \times k$ respectively and let $T > \Theta$ be integers satisfying:

1. A is invertible, the matrix
$$\begin{pmatrix} B & AB & \ldots & A^{T-1}B \end{pmatrix} \tag{3.3}$$
has full row rank δ and its rows form the parity check matrix of a block code of distance at least d_1.

2. The matrix
$$\begin{pmatrix} C \\ CA \\ \vdots \\ CA^{\Theta-1} \end{pmatrix} \tag{3.4}$$
has full column rank δ and its columns define the generator matrix of a block code of distance d_2.

Assumption 3.2 implies that (A, B) forms a controllable pair and (A, C) forms an observable pair, in particular (2.3) forms a minimal state space representation of a non-catastrophic encoder (see [16] for details). The integer Θ appearing in (3.4) is necessarily larger than the observability index of the matrix pair (A, C). The observability index describes the maximal number of consecutive zero code words $v_t = \binom{y_t}{u_t}$ starting from a nonzero state. In the coding literature [5] this number appears as the solution for the zero run problem.

Conditions (3.3) and (3.4) are stronger than the simple controllability and observability requirement and they imply that valid code trajectories have necessarily certain distance properties. The following lemma makes this precise.

Lemma 3.3 *Assume the matrices A, B, C and the integers T, Θ satisfy the conditions of Assumption 3.2. Assume that*

$$\left\{\begin{pmatrix} y_t \\ u_t \end{pmatrix}\right\}_{t\geq 0} \text{ and } \left\{\begin{pmatrix} \tilde{y}_t \\ \tilde{u}_t \end{pmatrix}\right\}_{t\geq 0}$$

are two set of codewords both satisfying (2.3). *Let $\{x_t\}_{t\geq 0}$ and $\{\tilde{x}_t\}_{t\geq 0}$ be the corresponding set of state vectors. If there is a $\tau \geq 0$ with*

$$x_\tau = \tilde{x}_\tau \text{ and } x_{\tau+1} \neq \tilde{x}_{\tau+1}$$

then for any γ satisfying $\tau + T > \gamma \geq \tau$ one has that

$$\sum_{t=\tau}^{\gamma} (\text{dist}\,(u_t, \tilde{u}_t) + \text{dist}\,(y_t, \tilde{y}_t)) \geq \min\left(d_1, \left\lfloor \frac{\gamma - \tau}{\Theta} \right\rfloor + 1\right). \tag{3.5}$$

Proof: The proof is established by induction over the integer $\eta := \left\lfloor \frac{\gamma-\tau}{\Theta} \right\rfloor$. Since $x_{\tau+1} \neq \tilde{x}_{\tau+1}$ it follows that $u_\tau \neq \tilde{u}_\tau$ and the result is therefore true for $\eta = 0$. Assume now that the result has already been proved for $\eta = k$ and let $\gamma = k\Theta + \tau$. By induction hypothesis we can assume that $\sum_{t=\tau}^{\gamma} (\text{dist}\,(u_t, \tilde{u}_t) + \text{dist}\,(y_t, \tilde{y}_t)) \geq \min\,(d_1, k+1)$. If $x_{\gamma+1} = \tilde{x}_{\gamma+1}$ then necessarily one has

$$\begin{pmatrix} B & AB & \ldots & A^{\gamma-\tau}B \end{pmatrix} \begin{pmatrix} u_\gamma - \tilde{u}_\gamma \\ \vdots \\ u_\tau - \tilde{u}_\tau \end{pmatrix} = 0. \tag{3.6}$$

By Assumption 3.2 it follows that $\sum_{t=\tau}^{\gamma} \text{dist}\,(u_t, \tilde{u}_t) \geq d_1$. In this situation the proof would be complete. If $x_{\gamma+1} \neq \tilde{x}_{\gamma+1}$ then either

$$\sum_{t=\gamma+1}^{\gamma+\Theta} \text{dist}\,(u_t, \tilde{u}_t) \geq 1$$

(in this case the induction step would be complete as well) or alternatively $u_t = \tilde{u}_t$ for $t = \gamma+1, \ldots, \gamma+\Theta$. In the latter situation it follows from Proposition 2.6 and from the second condition of Assumption 3.2 that $\sum_{t=\gamma+1}^{\gamma+\Theta} \text{dist}\,(y_t, \tilde{y}_t) \geq d_2 \geq 1$. In either case we did show the claim for $\gamma + \Theta = (k+1)\Theta + \tau$, i.e. for $\eta = k+1$. □

The main result of this section is formulated in the following Theorem. The result shows that under certain assumptions on the weight distribution of the errors $\{e_t, f_t\}$ it is possible to decode the received message uniquely. The proof of this theorem will be established in the next section through an explicit iterative decoding algorithm.

Theorem 3.4 *Let A, B, C, D be matrices satisfying the conditions of Assumption 3.2. Consider a received message*

$$\left\{ \begin{pmatrix} \hat{y}_t \\ \hat{u}_t \end{pmatrix} \right\}_{t \geq 0} = \left\{ \begin{pmatrix} y_t + f_t \\ u_t + e_t \end{pmatrix} \right\}_{t \geq 0}. \tag{3.7}$$

Assume that for any $\tau \geq 0$ the error sequence

$$\begin{pmatrix} f_\tau \\ e_\tau \end{pmatrix}, \ldots, \begin{pmatrix} f_{\tau+T-1} \\ e_{\tau+T-1} \end{pmatrix}$$

has weight at most

$$\lambda := \min \left(\left\lfloor \frac{d_1 - 1}{2} \right\rfloor, \left\lfloor \frac{T}{2\Theta} \right\rfloor \right).$$

In this situation it is possible to uniquely compute the transmitted sequence $\left\{ \binom{y_t}{u_t} \right\}_{t \geq 0}$.

Remark 3.5 The major task in the decoding procedure will be the decoding of the block codes defined in (3.3) and (3.4). If the matrices A, B, C are chosen in a way which allows one to decode the block codes defined in (3.3) and (3.4) through an efficient algebraic decoding algorithm then one is led to an efficient algebraic decoding algorithm of the associated convolutional code.

Remark 3.6 If $\lfloor \frac{d_1-1}{2} \rfloor = \lambda$ it follows from [16, Theorem 3.1] that the free distance of the convolutional code defined by the matrices A, B, C, D is at least d_1. Theorem 3.4 essentially states that decoding is possible if no more than 'half the free distance' errors occur in any time interval of length T.

4 The Decoding Algorithm

The presented decoding algorithm is an iterative algorithm. We hence will assume that the received message has already been correctly decoded up to time τ. In other words we will assume that the code words

$$\begin{pmatrix} y_0 \\ u_0 \end{pmatrix}, \begin{pmatrix} y_1 \\ u_1 \end{pmatrix}, \ldots, \begin{pmatrix} y_{\tau-1} \\ u_{\tau-1} \end{pmatrix}$$

have been computed correctly and that the state x_τ is known. Under these conditions we will show how to decode at least another Θ code vectors.

We start with some preliminary remarks: Assume for a moment that the message vectors

$$\hat{u}_{\tau+T-\Theta+1}, \hat{u}_{\tau+T-\Theta+2}, \ldots, \hat{u}_{\tau+T} \tag{4.1}$$

have been correctly received. Assume also that the error sequence

$$f_{\tau+T-\Theta+1}, f_{\tau+T-\Theta+2}, \ldots, f_{\tau+T} \tag{4.2}$$

has weight at most $\lfloor \frac{d_2-1}{2} \rfloor$ where d_2 is the distance of the block code introduced in Assumption 3.2. From Proposition 2.6 it follows that

$$\begin{pmatrix} y_{\tau+T-\Theta+1} \\ y_{\tau+T-\Theta+2} \\ \vdots \\ \vdots \\ y_{\tau+T} \end{pmatrix} - \begin{pmatrix} D & 0 & \cdots & & 0 \\ CB & D & \ddots & & \vdots \\ CAB & CB & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & 0 \\ CA^{\Theta-2}B & CA^{\Theta-3}B & \cdots & CB & D \end{pmatrix} \begin{pmatrix} u_{\tau+T-\Theta+1} \\ u_{\tau+T-\Theta+2} \\ \vdots \\ \vdots \\ u_{\tau+T} \end{pmatrix}$$
$$= \begin{pmatrix} C \\ CA \\ \vdots \\ \vdots \\ CA^{\Theta-1} \end{pmatrix} x_{\tau+T-\Theta+1}. \tag{4.3}$$

In particular the left hand side of the previous equation is in the column space of the block code generated by the columns of

$$\begin{pmatrix} C \\ CA \\ \vdots \\ CA^{\Theta-1} \end{pmatrix}. \tag{4.4}$$

Since this last code has distance d_2 it is possible to both compute the errors appearing in (4.2) and the state vector $x_{\tau+T-\Theta+1}$. Since the state vector $x_{\tau+T-\Theta+1}$ is also equal to

$$x_{\tau+T-\Theta+1} = A^{T-\Theta+1}x_\tau + \left(A^{T-\Theta}B \ \ldots \ B\right) \begin{pmatrix} u_\tau \\ \vdots \\ u_{\tau+T-\Theta} \end{pmatrix} \tag{4.5}$$

it is possible to compute the error sequence $e_\tau, \ldots, e_{\tau+T-\Theta}$ from the syndrome vector

$$\left(A^{T-\Theta}B \ \ldots \ B\right) \begin{pmatrix} \hat{u}_\tau \\ \vdots \\ \hat{u}_{\tau+T-\Theta} \end{pmatrix} - x_{\tau+T-\Theta+1} - A^{T-\Theta+1}x_\tau. \tag{4.6}$$

Once the error sequence $e_\tau, \ldots, e_{\tau+T-\Theta}$ has been computed we can compute the sequence of states $x_{\tau+1}, \ldots, x_{\tau+T-\Theta+1}$ and the sequence of parity vectors $y_\tau, \dots, y_{\tau+T-\Theta}$ using the defining equation (2.3). Because of the assumptions the error sequence

$$\begin{pmatrix} f_\tau = y_\tau - \tilde{y}_\tau \\ e_\tau = u_\tau - \tilde{u}_\tau \end{pmatrix}, \dots, \begin{pmatrix} f_{\tau+T-\Theta} = y_{\tau+T-\Theta} - \tilde{y}_{\tau+T-\Theta} \\ e_{\tau+T-\Theta} = u_{\tau+T-\Theta} - \tilde{u}_{\tau+T-\Theta} \end{pmatrix} \tag{4.7}$$

must have weight at most λ.

After these preliminary remarks we explain the algorithm.

In a first step we attempt to compute the state vector $x_{\tau+T-\Theta+1}$ from identity (4.3) and the error sequence (4.7) from identities (4.6) and (2.3). Several things might happen then.

A) It is possible that we cannot compute the state vector $x_{\tau+T-\Theta+1}$ from identity (4.3).

B) It is possible that we did compute a state vector $x_{\tau+T-\Theta+1}$ and the resulting error sequence has weight

$$\sum_{t=\tau}^{\tau+T-\Theta} (\mathrm{wt}(f_t) + \mathrm{wt}(e_t)) > \lambda. \tag{4.8}$$

C) It is possible that we compute a state vector $x_{\tau+T-\Theta+1}$ and the weight of the error sequence appearing in (4.8) is less than or equal to λ.

In the sequel we will show that after some possible iterations we will always end up with the situation C).

In situations A) and B) we can conclude that either sequence (4.1) was wrong or the weight of the sequence appearing in (4.2) is larger than $\lfloor \frac{d_2-1}{2} \rfloor$. In both these cases we will attempt to compute the state vector $x_{\tau+T-2\Theta+1}$ using the new sequence of message vectors

$$\hat{u}_{\tau+T-2\Theta+1}, \hat{u}_{\tau+T-2\Theta+2}, \dots, \hat{u}_{\tau+T-\Theta} \tag{4.9}$$

and parity check vectors

$$\hat{y}_{\tau+T-2\Theta+1}, \hat{y}_{\tau+T-2\Theta+2}, \dots, \hat{y}_{\tau+T-\Theta}. \tag{4.10}$$

Since there were mistakes in the last Θ code words, i.e. the weight of the sequences appearing in (4.1) and (4.2) were nonzero we can assume that there were at most $\lambda - 1$ errors among the received sequence

$$\begin{pmatrix} \hat{y}_\tau \\ \hat{u}_\tau \end{pmatrix}, \dots, \begin{pmatrix} \hat{y}_{\tau+T-\Theta} \\ \hat{u}_{\tau+T-\Theta} \end{pmatrix}.$$

Again if we cannot compute either $x_{\tau+T-2\Theta+1}$ or if the weight

$$\sum_{t=\tau}^{\tau+T-2\Theta} (\mathrm{wt}(f_t) + \mathrm{wt}(e_t)) > \lambda - 1$$

we conclude that either sequence (4.9) was wrong or there were more than $\lfloor \frac{d_2-1}{2} \rfloor$ mistakes in the sequence appearing in (4.10).

Proceeding in this way iteratively we will find after h iterations that the state vector $x_{\tau+T-h\Theta+1}$ can be computed from the data

$$\hat{u}_{\tau+T-h\Theta+1}, \hat{u}_{\tau+T-h\Theta+2}, \ldots, \hat{u}_{\tau+T-(h-1)\Theta}$$

and

$$\hat{y}_{\tau+T-h\Theta+1}, \hat{y}_{\tau+T-h\Theta+2}, \ldots, \hat{y}_{\tau+T-(h-1)\Theta}$$

and in addition we have that the weight

$$\sum_{t=\tau}^{\tau+T-h\Theta} (\mathrm{wt}(f_t) + \mathrm{wt}(e_t)) \leq \lambda - h + 1.$$

In other words we did arrive at situation C) after h iterations. In general it would be wrong to assume that the state $x_{\tau+T-h\Theta}$ is a correct state. However Lemma 3.3 and the assumption on the error pattern as formulated in Theorem 3.4 will guarantee that the state $x_{\tau+\Theta}$ is correctly computed. Indeed if the computed state $x_{\tau+\Theta}$ would be different from the true state $x_{\tau+\Theta}$ then the computed code sequence

$$\begin{pmatrix} y_\tau \\ u_\tau \end{pmatrix}, \ldots, \begin{pmatrix} y_{\tau+T} \\ u_{\tau+T} \end{pmatrix}$$

would be in distance more than min $\left(d_1, \lfloor \frac{T}{\Theta} \rfloor + 1\right)$ apart from the true code sequence. This is not possible since we assumed that at most λ errors did occur. The computed state $x_{\tau+\Theta}$ has therefore to be correct.

Under the given assumptions we also conclude that

$$\begin{pmatrix} y_\tau \\ u_\tau \end{pmatrix}, \ldots, \begin{pmatrix} y_{\tau+\Theta-1} \\ u_{\tau+\Theta-1} \end{pmatrix}.$$

has been decoded correctly. In this way Θ additional time units were decoded.

The algorithm proceeds now again from the beginning by replacing the initial state x_τ with the state $x_{\tau+\Theta}$.

Remark 4.1 Crucial for the algorithm was the computation of a new state vector $x_{\tau+\Theta}$. Due to the fact that we have been working with a systematic encoder one has a certain asymmetry between the assumption on the error patterns among the input sequence $\{u_t\}$ and the output sequence $\{y_t\}$. In [1] a new method for computing the state vector $x_{\tau+\Theta}$ was announced for codes having rate $1/n$. This method seems to have advantages over the one presented here and it will be addressed in upcoming research.

In the next section we will show that under certain probabilistic assumptions one can speed up the algorithm considerably.

5 Complexity Considerations and Variations of the Algorithm

It is clear from the described algorithm that the block codes described in Assumption 3.2 have to be decoded over and over again. It is hence desirable to construct (A, B, C) matrices where these codes have both good distance properties and come with efficient decoding algorithms.

Even in these cases up to $\lfloor \frac{T}{\Theta} \rfloor$ vectors have to be decoded by the two block codes described in Assumption 3.2 to decode in the worst case just Θ time units. The algorithm takes into consideration many unlikely events and it 'cautiously' assumes that $x_{\tau+\Theta}$ is correct despite the fact that there is already a preliminary estimate for $x_{\tau+T-h\Theta-1}$.

We see in principle two ways how to speed up the algorithm considerably:

Variation 1: One way which will guarantee that the algorithm performs much quicker is to assume that the number of errors λ which are permitted in every time interval of length T is less than the number specified in Theorem 3.4. For this assume e.g. that for any $\tau \geq 0$ the error sequence

$$\begin{pmatrix} f_\tau \\ e_\tau \end{pmatrix}, \ldots, \begin{pmatrix} f_{\tau+T-1} \\ e_{\tau+T-1} \end{pmatrix}$$

has weight at most

$$\tilde{\lambda} := \min \left(\left\lfloor \frac{d_1 - 1}{2} \right\rfloor, \left\lfloor \frac{T}{4\Theta} \right\rfloor \right).$$

In contrast with Theorem 3.4 we assume that at most half the errors do occur over any time period T if $\lfloor \frac{T}{2\Theta} \rfloor < \lfloor \frac{d_1-1}{2} \rfloor$. In this situation one verifies with the help of Lemma 3.3 that not only the state vector $x_{\tau+\Theta}$ is correct but that even the state vector $x_{\tau+\frac{T}{2}}$ has been correctly computed. In this variation up to $\lfloor \frac{T}{2\Theta} \rfloor$ vectors have to be decoded by the two block codes described in Assumption 3.2 in order to decode $\frac{T}{2}$ time units.

Variation 2: Assume that the distance d_2 of the block code

$$\begin{pmatrix} C \\ CA \\ \vdots \\ CA^{\Theta-1} \end{pmatrix}$$

is relatively large, i.e. $d_2 >> 1$. Under certain probabilistic assumptions it is then possible to proceed directly with the state vector $x_{\tau+T-h\Theta-1}$ after h iterations of the algorithm. This works particularly well if one rejects the computation of the state vector $x_{\tau+T-j\Theta-1}$ in the jth iteration of the algorithm as soon as the weight of the error sequence

$$f_{\tau+T-(j+1)\Theta+1}, f_{\tau+T-j\Theta+2}, \ldots, f_{\tau+T-j\Theta} \tag{5.1}$$

is more than a certain fraction of d_2. By doing this it becomes highly unlikely that a wrong state vector $x_{\tau+T-h\Theta-1}$ was computed. Indeed the computation of a wrong state vector $x_{\tau+T-h\Theta-1}$ where the weight of the parity check sequence appearing in (5.1) was low can only happen if in the local description of the code as provided in Proposition 2.6 the errors among the message vectors and parity check vectors do cancel in a very particular way. In this way one can assume with high probability that after h iterations we actually will find a correct state vector $x_{\tau+T-h\Theta-1}$. The probability will depend on the fraction of d_2 which one uses for rejecting the error sequence appearing in (5.1).

During the decoding process of

$$\begin{pmatrix} \hat{y}_\tau \\ \hat{u}_\tau \end{pmatrix}, \begin{pmatrix} \hat{y}_{\tau+1} \\ \hat{u}_{\tau+1} \end{pmatrix}, \ldots, \begin{pmatrix} \hat{y}_{\tau+T-(h+1)\Theta-2} \\ \hat{u}_{\tau+T-(h+1)\Theta-2} \end{pmatrix}$$

one has one more verification that $x_{\tau+T-h\Theta-1}$ was actually correctly computed.

We conclude the section with some complexity estimates for the second variation of the algorithm. For this assume that in average the error sequence

$$\begin{pmatrix} f_\tau \\ e_\tau \end{pmatrix}, \ldots, \begin{pmatrix} f_{\tau+\Theta-1} \\ e_{\tau+\Theta-1} \end{pmatrix}$$

has weight one. In other words we do assume that over Θ time units the expected number of errors is one. In average there are therefore at most $h = 2$ iterations needed until it is possible to compute the new state vector $x_{\tau+T-h\Theta-1}$. But this means that for the decoding of T message vectors we need to decode in average up to two block codes of rate $\frac{\delta}{(n-k)\Theta}$ having the form (3.4) and in average one block code of rate $\frac{\delta}{kT}$ having the form (3.3).

The described decoding algorithm seems to be particularly well suited in situations where once in a while there is a burst error and where in the remaining time the transmission is error free.

In the next section we explain a situation where both the block codes appearing in (3.3) and in (3.4) are Reed Solomon type block codes.

6 Decoding of Reed Solomon and BCH type Convolutional Codes

In order that Assumption 3.2 is satisfied it will be necessary that the parity check matrix appearing in (3.3) has some good distance properties. Moreover the presented algorithm works best if the associated block code can be decoded efficiently.

In [15, 16, 21] convolutional codes were presented where the block code defined by (3.3) is a BCH code. In the sequel we will illustrate the decoding algorithm using these codes. For the sake of presentation we will illustrate the algorithm only for Reed Solomon convolutional codes as presented in [15] and we leave the extension to the general BCH situation to the reader.

Let α be a primitive of the field $\mathbb{F}_q$ and assume that $n > k$ are positive integers with $k \geq n - k$. Consider the matrices

$$A := \begin{pmatrix} \alpha^k & 0 & \cdots & 0 \\ 0 & \alpha^{2k} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \alpha^{\delta k} \end{pmatrix},$$

$$B := \begin{pmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^{k-1} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(k-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \alpha^\delta & \alpha^{2\delta} & \cdots & \alpha^{\delta(k-1)} \end{pmatrix},$$

$$C := \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha & \alpha^2 & \cdots & \alpha^\delta \\ \alpha^2 & \alpha^4 & \cdots & \alpha^{2\delta} \\ \vdots & \vdots & & \vdots \\ \alpha^{n-k-1} & \alpha^{2(n-k-1)} & \cdots & \alpha^{\delta(n-k-1)} \end{pmatrix},$$

$$D := \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha & \alpha^2 & \cdots & \alpha^k \\ \vdots & \vdots & & \vdots \\ \alpha^{(n-k-1)} & \alpha^{2(n-k-1)} & \cdots & \alpha^{k(n-k-1)} \end{pmatrix}.$$

It has been shown in [15]:

Theorem 6.1 *Assume $|\mathbb{F}_q| = q > \delta k \left\lceil \frac{\delta}{n-k} \right\rceil$. Then the convolutional code $\mathcal{C}$ defined by the matrices A, B, C, D represents an observable, rate k/n convolutional code with degree δ and free distance*

$$d_f(\mathcal{C}) \geq \delta + 1. \tag{6.1}$$

Since the free distance of this code is at least $\delta+1$ it should be possible to decode up to $\lfloor \frac{\delta}{2} \rfloor$ errors. Actually we will show (compare with Remark 3.6) that the decoding algorithm presented in Section 4 is capable of decoding the received message if at most $\lfloor \frac{\delta}{2} \rfloor$ errors do occur in any time interval of length T where T is defined as

$$T := \delta \left\lceil \frac{\delta}{n-k} \right\rceil . \tag{6.2}$$

In order to apply Theorem 3.4 we define

$$\Theta := \left\lceil \frac{\delta}{n-k} \right\rceil . \tag{6.3}$$

Because of the assumed number of field elements we have $q > kT$. The block code described in (3.3) defines therefore a Reed Solomon code of distance $\delta + 1$. This code is in particular a maximum distance separable (MDS) code.

The block code appearing in (3.4) describes a MDS block code as well, although the distance is small since we did choose the smallest possible value for Θ. Assumption 3.2 is therefore in place with $d_1 = \delta + 1$ and $d_2 \geq 1$. Finally note that the number λ appearing in Theorem 3.4 is equal to

$$\lambda = \left\lfloor \frac{d_1 - 1}{2} \right\rfloor = \left\lfloor \frac{T}{2\Theta} \right\rfloor = \left\lfloor \frac{\delta}{2} \right\rfloor .$$

According to Theorem 3.4 it is possible to decode a message word if at most λ mistakes did occur over any time interval of length T. The decoding algorithm of Section 4 does therefore fully apply.

In order to illustrate the second variation of the algorithm as presented in Section 5 let

$$\tilde{T} := 2T = 2\delta \left\lceil \frac{\delta}{n-k} \right\rceil \quad \text{and} \quad \tilde{\Theta} := 2\Theta = 2 \left\lceil \frac{\delta}{n-k} \right\rceil . \tag{6.4}$$

Assume that the field size $q > k\tilde{T} = 2kT$. With these choices the block code described in (3.4) has distance at least δ. If in the iteration of the decoding algorithm we require that the weight of the computed error sequence described in (5.1) is e.g. at most $\frac{1}{10}\delta$ then the likelihood that an accepted state vector $x_{\tau+T-h\Theta-1}$ is actually a correct state is very high. (The balls centered around the code words and having radius $\frac{1}{10}\delta$ are a small fraction inside the total configuration space).

Conclusions

We presented an iterative decoding algorithm for convolutional codes whose performance mainly depends on the availability of good algorithms to decode the block codes appearing in (3.3) and (3.4). If these block codes are of Reed Solomon type (as described in [15]) or of BCH type (as described in [16, 21]) then the major decoding task can be accomplished by iteratively applying e.g. the Berlekamp Massey algorithm.

Acknowledgments.
I am grateful to Brian Allen, Dan Costello, Dave Forney, Heide Glüsing-Lüerssen, Hans Schumacher, Roxana Smarandache, Paul Weiner and Eric York for helpful discussions in connection with the presented research.

References

[1] B. M. Allen and J. Rosenthal (1998). Parity-Check decoding of convolutional codes whose systems parameters have desirable algebraic properties, in *Proceedings of the 1998 IEEE International Symposium on Information Theory*, 307, Boston, MA.

[2] P. Elias (1955). Coding for Noisy Channels. *IRE Conv. Rec.* 4, 37–46.

[3] P. Fitzpatrick (1995). On the Key Equation. *IEEE Trans. Inform. Theory* IT-41, No. 5, 1290–1302.

[4] E. Fornasini and M.E. Valcher (1994). Algebraic Aspects of 2D Convolutional Codes. *IEEE Trans. Inform. Theory* IT-40, No. 4, 1068–1082.

[5] G. D. Forney (1973). Structural Analysis of Convolutional Codes via Dual Codes. *IEEE Trans. Inform. Theory* IT-19, No. 5, 512–518.

[6] G. D. Forney (1974). Convolutional Codes II: Maximum Likelihood Decoding. *Inform. Control* 25, 222–266.

[7] R. Johannesson and K. Sh. Zigangirov (1999). *Fundamentals of Convolutional Coding.* IEEE Press, New York.

[8] S. Lin and D. Costello (1983). *Error Control Coding: Fundamentals and Applications.* Prentice-Hall, Englewood Cliffs, NJ.

[9] J. L. Massey (1963). *Threshold decoding.* MIT Press, Cambridge, Massachusetts.

[10] J. L. Massey and M. K. Sain (1967). Codes, Automata, and Continuous Systems: Explicit Interconnections. *IEEE Trans. Automat. Contr.* AC-12, No. 6, 644–650.

[11] R.J. McEliece. The Algebraic Theory of Convolutional Codes. In *Handbook of Coding Theory*, R. Brualdi, W.C. Huffman and V. Pless (eds.). Elsevier Science Publishers, Amsterdam, The Netherlands, 1998, To appear.

[12] Ph. Piret (1988). *Convolutional Codes, an Algebraic Approach.* MIT Press, Cambridge, MA.

[13] J. Rosenthal (1997). Some Interesting Problems in Systems Theory which are of Fundamental Importance in Coding Theory, in *Proc. of the 36th IEEE Conference on Decision and Control*, 4574–4579, San Diego, California.

[14] J. Rosenthal and J. M. Schumacher (1997). Realization by Inspection. *IEEE Trans. Automat. Contr.* AC-42, No. 9, 1257–1263.

[15] J. Rosenthal, J. M. Schumacher and E.V. York (1996). On Behaviors and Convolutional Codes. *IEEE Trans. Inform. Theory* 42, No. 6, 1881–1891.

[16] J. Rosenthal and E.V. York (Oct. 1997). BCH Convolutional Codes. Tech. rep., University of Notre Dame, Dept. of Mathematics, Preprint # 271. Available at http://www.nd.edu/˜rosen/preprints.html.

[17] R. M. Tanner (1983). A recursive approach to low complexity codes. *IEEE Trans. Inform. Theory* 27, No. 5, 533–547.

[18] M.E. Valcher and E. Fornasini (1994). On 2D Finite Support Convolutional Codes: an Algebraic Approach. *Multidim. Sys. and Sign. Proc.* 5, 231–243.

[19] P. Weiner (1998). *Multidimensional Convolutional Codes.* Ph.D. thesis, University of Notre Dame, Available at http://www.nd.edu/˜rosen/preprints.html.

[20] N. Wiberg, H.A. Loeliger and R. Koetter (1995). Codes and Iterative Decoding on General Graphs. *European Trans. on Telecommunications* 6, No. 5, 513–525.

[21] E.V. York (1997). *Algebraic Description and Construction of Error Correcting Codes, a Systems Theory Point of View.* Ph.D. thesis, University of Notre Dame, Available at http://www.nd.edu/˜rosen/preprints.html.

Department of Mathematics, University of Notre Dame,
Notre Dame, Indiana 46556, USA.
e-mail: Rosenthal.1@nd.edu, *URL:* http://www.nd.edu/˜rosen/