An Application of the FGLM Techniques to Linear Codes

Mijail Borges Quintana[†] Miguel A. Borges Trenard[†]

Franz Winkler[‡]

[†] *Departamento de Matemática, Fac. de Ciencias.*
Universidad de Oriente, Santiago de Cuba 90500, Cuba.
mijail@csd.uo.edu.cu, mborges@csd.uo.edu.cu

[‡] *RISC-Linz, Johannes Kepler University,*
Linz, Austria.
winkler@risc.uni-linz.ac.at

March 5, 2001

Abstract

In this paper we emphasize the main results and algorithms presented in [BWBa] and [BWBb] related with a new approach for decoding linear codes. This approach arrives from a mixture of the syndrome decoding method and the FGLM techniques. We also explain the meaning of "FGLM techniques" and the possibilities for new applications are shown by our concrete situation concerned to linear codes.

Introduction

It is well known how the capabilities of the technology is exponentially growing. Nowadays one can think reasonable in some applications that were very far away to consider, even five years ago. We do not think that the proposed method here can be applied to everyday live of coding theory applications, not even in the future; moreover, when there have been developed some reasonable good algorithms, mainly for the class of algebraic geometric codes (Goppa codes), following Duursma's ideas or Sakata's approach (see [Du, FR, HøPe, SJH, SJMJH, JLJH]). However, when it is possible to fulfill the challenge of the requirements of our approach (see Section 3) for a particular application of an specific linear code, then, one can be sure that the decoding process can be as faster as desired. Despite it is not discarted a possible particular cases of applications of this approach, we rather consider our result from the point of view of coding theory mainly with theoretical interest. From the point of view of the FGLM techniques, it shows extensions of these techniques and new possible applications.

The method presented in this paper for decoding linear codes is obtained by connecting the structure of the linear codes with the FGLM techniques that belongs to the framework of the Gröbner bases theory. This application of the FGLM techniques constitutes a good example of how to apply these tools in a more general way, and with less requirements like the admissibility of the term ordering. For the purpose of decoding, one can avoid the presentation with variables, and to present the algorithms with an approach closer to the usual one in coding theory. However, we want to show that the algorithms arrive in a natural way from an FGLM pattern, after making the corresponding connection with linear codes.

We assume the reader to be familiar with coding theory, particularly with linear codes. For an understanding of coding theory we recommend to have a look to [vL, CLO, PeWe]. As some notations we have: C denotes a linear code as a subspace of the vector space F_q^n; c is a codeword and y a received word; the corresponding error for y is denoted by e; H denotes a parity check matrix for C; we used the well known Hamming distance; t is the error-correcting capability of C; and $B(C,t)$ denotes the set of the balls with center in a codeword c and radio t.

For an understanding of the original FGLM algorithm and its firsts applications it is recommended to take a look to [FGLM, MMM]. In [BBM] a generalized FGLM pattern is presented. In this way, FGLM is not anymore just an algorithm, but a model that can be particularized and applied to many particular structures. For example, as it was shown in [BBM], the algorithms in [FGLM] and [MMM] are inmerse in the pattern algorithm presented in [BBM]. We understood as FGLM techniques any specification and its corresponding applications of this pattern to a particular setting. This proccess involves, as we will show here for the case of linear codes, new theoretical results.

To show the FGLM pattern algorithm would be good in order to see from where algorithm 1.3 comes, but it would need some deffinitions like Gröbner basis and some more explanations and notations. Authors are pleased to help by personal communication with interested readers. In this work we gave key comments which could help to readers for a comprenhensive and oriented reading of the literature related here. Let us suggest the readers; in advance, to read [BWBb] for more details or [BWBa] for a more complete analysis.

1 FGLM and linear codes

In connection with the vector space F_q^n a set X of variables will be introduced so that we can relate later the free commutative monoid in these variables with the structure of the linear code.

Given $(y_1, \ldots, y_n) \in F_q^n$, each component y_i can be represented as $y_i = \sum_{j=1}^{m} \beta_{ij}\alpha^{j-1}$. Based on this representation, the alphabet $X = \{x_{11}, \ldots, x_{1m}, \ldots, x_{n1}, \ldots, x_{nm} \mid i \in [1,n] \text{ and } j \in [1,m]\}$ is introduced and the free commutative monoid $[X]$ is associated with F_q^n in the following way: let ψ be the mapping from X to F_q^n that sends every x_{ij}

to the corresponding vector with zero in all its components up to the position i which is equal to α^{j-1}.

As $[\,X\,]$ is freely generated by X and ψ is a mapping of X onto a generating set of the additive monoid F_q^n, ψ can be extended in a natural way to a linear morphism of $[\,X\,]$ onto F_q^n; moreover, ψ maps $\prod_{i=1}^{n}\prod_{j=1}^{m} x_{ij}^{\beta_{ij}}$ into the vector $(\sum_{j=1}^{m}\beta_{ij}\alpha^{j-1})_i$ modulo p.

Remark 1.1 *Taking into consideration the comments made above, the variables x_{ij} can also be seen as the set of nm variables x_k, where $k=(i-1)m+j$. This last representation of variables will be used in the subsequent sections of this paper. This other representation of variables and the connection between F_q^n with the free commutative monoid with nm generators, is strongly related with the fact that F_q^n is isomorphic to F_p^{mn} as a vector space over F_p. However, we will keep all the time working in F_q^n.*

It is well known that a linear code defines an equivalence relation $R(C)$ in F_q^n. By the linear morphism ψ, the congruence modulo C can be transferred onto $[\,X\,]$ in the following way:

$$u \cong_C w \iff (\psi(u),\psi(w)) \in R(C) \iff \psi(u)H = \psi(w)H.$$

After setting $\xi(u) := \psi(u)H$, we can write this relation as: $u \cong_C w \iff \xi(u) = \xi(w)$ [1].

Definition 1.2 (Standard representation) *The word w is said to be in* standard representation *iff all the exponents of the variables in the representation of w are less than p. Given $y \in F_q^n$, we say that w is the* standard representation *of y iff $\psi(w) = y$ and w is in standard representation.*

Note that the morphism ψ is surjective, but it is not injective. By considering just the standard representations in $[\,X\,]$, then, the correspondence with F_q^n is one-to-one. That is, for every $y \in F_q^n$ there exists a unique standard representation in $[\,X\,]$.

As two more notations we have: $Card(U)$ denotes the cardinal of the set U, and $Ind(w)$ denotes the set of $i \in [1,n]$ s.t. there exists $j \in [1,m]$ for which x_{ij} divides the word w.

There are two important structures in our approach, the set of canonical forms and the function Matphi.

Properties of a set of canonical forms (N).

(i) $1 \in N$ and $N \subset [\,X\,]$.

(ii) $Card(N) = q^{n-k}$.

(iii) Two different words of N determine distinct coset module $R(C)$.

(iv) For all $w \in N$ there exists $x \in X$ such that $w = w'x$ and $w' \in N$.

Properties of the function Matphi:

(i) Matphi is a mapping ϕ from $N \times X$ onto N [2].

[1] Note that $\xi(w)$ is nothing more that the syndrome corresponding to the vector $\psi(w)$ associated to w.

[2] The image of (w,x) will be denoted by $\phi(w,x)$.

(ii) For all $(w, x) \in N \times X \quad \xi(\phi(w, x)) = \xi(wx)$ [3].

The following algorithm builds such a set N and a function Matphi with the required properties. As subroutines of the algorithm we have the ones well known from [FGLM]:

InsertNexts$[t, List]$ inserts properly the products xt (for $x \in X$) in $List$ and sorts it by increasing ordering with respect to the ordering $<$.
NextTerm$[List]$ removes the first element from $List$ and returns it.
Member$[v', \{v_1, \ldots, v_r\}]$ returns $True$ if $v' \in \{v_1, \ldots, v_r\}$ and $False$ otherwise. When the output is True, it also returns an additional output with the position j such that $v' = v_j$.

Algorithm 1.3 (FGLM for linear codes)

Input: *p, n, m, H the defining parameters for a given linear code.*

Output: N, ϕ.

1. $List := \{1\}$; $N := \emptyset$; $r := 0$;
2. **While** $List \neq \emptyset$ **do**
3. $w :=$ **NextTerm**$[List]$;
4. $v' := \xi(w)$;
5. $(\Lambda,\ j) :=$ **Member**$[v', \{v_1, \ldots, v_r\}]$;
6. **If** $\Lambda =$ **True**
7. **then** *for each k such that $w = ux_k$ with $u \in N$ do*
8. $\phi(u, x_k) := w_j$;
9. **else** $r := r + 1$;
10. $v_r := v'$;
11. $w_r := w$; $N := N \cup \{w_r\}$;
12. $List :=$ **InsertNexts**$[w_r, List]$;
13. *for each k such that $w = ux_k$ with $u \in N$ do*
14. $\phi(u, x_k) := w$;
15. **Return**$[N,\ \phi]$.

For our porpuse, it is necessary to keep a certain order in $List$. We defined $<_e$, a term ordering on $[X]$ called the error vector term ordering (see [BWBa]). This term ordering is used inside InsertNexts, which take care of keeping the elements in $List$ ordered by $<_e$. The reader can see in [BWBa] the justification of this algorithm.

2 Decoding linear codes

In this section we show the basis of an easy to understand and fast decoding algorithm which takes as its input the output of Algorithm 1.3. Previously, we have to introduce a

[3] it means that $\phi(w, x)$ is the representative element in N for the coset determined by $\xi(wx)$.

canonical form function that will be denoted by CF. In fact CF is defined in a recursive way:

$$\begin{array}{l} CF: \ [\,X\,] \ \longrightarrow \ N \\ \\ CF(1) := 1; \\ CF(w) := \phi(CF(u), x_k), \\ \text{where } w = ux_k \text{ and } u \in [\,X_k\,] = [\,x_1, \ldots, x_k\,]. \end{array} \tag{1}$$

Now we will procceed to give the results that show the way of the application to the decoding process.

Theorem 2.1 (Every element has a canonical form) *For every element w in $[\,X\,]$, $CF(w)$ is the unique element in N with the same syndrome as w, that is:*

$$\xi(w) = \xi(CF(w)).$$

As a conclusion of this theorem each syndrome ($\xi(w)$ for the corresponding $w \in [\,X\,]$) of the linear code has a representative element in N ($CF(w)$).

Theorem 2.2 (Decoding) *Let w be such that $\psi(w) \in B(C,t)$. Then $\psi(CF(w))$ is the error vector corresponding to $\psi(w)$.*

Note that from Theorem 2.2 we can already extract a decoding algorithm. It is only necessary to know the error-correcting capability of the code.

Theorem 2.3 (error-correcting capability) *Let List be the list of words in Step 3 of Algorithm 1.3 and let w be the first element analyzed by* NextTerm[*List*] *such that w does not belong to N and w is in standard representation. Then*

$$t = Card(Ind(w)) - 1.$$

The algorithm for decoding given below is a direct consequence of the former three theorems. It consists just in computing for a received vector y the corresponding canonical form $CF(w)$, where $y = \psi(w)$. Thus, if the weight of the corresponding vector e to $CF(w)$ ($e = \psi(CF(w))$) is at most t, then $y - e$ is the codeword; otherwise, we can not decode properly the received word. We recommend the reader to see the example presented in in [BWBa, BWBb]. In [BWBa] the example and the section devoted to the complexity contains a deeper analysis and more details, while in [BWBb] the reader can get faster the key points. First let us explain some procedures used by the algorithm.

Read: Reads the vector y and returns its standard representation in $[\,X\,]$.
NextVar: Returns the first index k of a variable such that $x_k \in$ *Supp(w)* and then computes $w := w/x_k$. If $w = 1$, $NextVar$ returns 0.

Remember that x_{ij} corresponds to $x_{(i-1)m+j}$ when we use only one index instead of two (see Remark 1.1).

Algorithm 2.4 (The algorithm for decoding)

Input: *A received vector y.*

Output: *The codeword corresponding to y, if $y \in B(C,t)$, and the error message "more than t errors" otherwise.*

1. $w :=$ **Read**(y); $w_e := 1$;
2. $i :=$ **NextVar**$[w]$;
3. **While** $i \neq 0$ **do**
4. $\quad w_e := \phi(w_e, x_i)$;
5. $\quad i :=$ **NextVar**$[w]$;
6. **If** $weight(\psi(w_e)) > t$ **then** *Return["more than t errors"]*
7. **else** *Return*$[y - \psi(w_e)]$

Steps from 3 to 5 correspond to the computation of $CF(w)$, the correctness of this algorithm follows directly from Theorems 2.2 and 2.3.

3 Some comments about complexity

It is necessary to say that we consider operations not only given by computation, but every kind of simple steps for getting the codeword.

The algorithm 2.4 for decoding computes the corresponding codeword of the received word in $\mathcal{O}((p-1)mn)$ operations. Notes that if the field F_q is fixed, the algorithms takes linear time. Under certain bounds, easy to get, our algorithm is less than $\mathcal{O}(n^2)$, while the well known algorithms for the class of Goppa codes, following Duursma's ideas and Sakata's approach take $\mathcal{O}(n^3)$ operations. In certain situations, it can be possible to get a complexity less than $\mathcal{O}(n^3)$ but never less than or equal to $\mathcal{O}(n^2)$. However, our approach could not be applied to many Goppa codes for which it is possible to apply the others algorithms. The reason is the requirements of our method discussed in [BWBa]. Nevertheless, it is theoretical interesting that there exists an algorithm almost linear for the process of decoding, if one assumes the requirements fulfilled. Also note that in special situations of application of linear codes, where the technology available provides the needed condition in order to carry out our method for a particular code, then one can use this way to obtain a faster decoding process.

The following parragraph clarifies the requirements. In [BWBa] all the formulas given below are proved and commented.

The memory required for Algorithm 1.3 is $q^{n-k}(cmn^2 + c_1 n - c_1 k)$, the number of operations performed by Algorithm 1.3 is $\mathcal{O}(mn^2 q^{n-k})$ operations. In order to use Algorithm 2.4 for decoding we need to keep the information contained in Matphi, whose size is given by $q^{n-k}(nc + nmc' + c'')$.

The complexity of Algorithm 1.3 is reasonable if one keeps in mind that this algorithm computes among all the q^n vectors in F_q^n the q^{n-k} representative vectors for the cosets determined by the code. Moreover, these representative elements are not arbitrarily chosen.

The algorithm also computes the Matphi, which gives us a way of multiplying cosets. We only need a computer powerful enough for computing the Matphi for the code. Then, if the code is really big, at least the same computer that was able to execute Algorithm 1.3 will be able to decode by Algorithm 2.4.

If one considers n and d fixed, it is an interesting problem in Coding Theory to find k as big as possible such that such a $[n, k, d]$ linear code exists. We know from the bounds given before that the better the code is (which means, if n and d are fixed, the bigger k is) the less information one needs to store and the fewer operations are needed in Algorithm 1.3.

References

[BBM] M. Borges-Trenard, M. Borges-Quintana, T. Mora. Computing Gröbner Bases by FGLM Techniques in a Noncommutative Setting. *J. Symbolic Computation* (to appear 2000).

[BWBa] M. Borges-Quintana, F. Winkler, M. Borges-Trenard. FGLM Techniques Applied to Linear Codes – An Algorithm for Decoding Linear Codes. Techn. Rep. *RISC-Linz, RISC - 00-14*, J. Kepler Univ., Linz, Austria (2000).

[BWBb] M. Borges-Quintana, F. Winkler, M. Borges-Trenard. An FGLM Method for Decoding Linear Codes. In *Proceedings of the Conference EACA-2000.* Barcelona, Spain, pp. 117-128 (2000).

[CCS] A. M. Cohen, H. Cuypers, H. Sterk (Eds). *Some Tapas of Computer Algebra.* Springer-Verlag, Berlin (1999).

[CLO] D. Cox, J. Little, D. O'Shea. *Using Algebraic Geometry.* Springer-Verlag, New York (1998).

[Du] I. M. Duursma. Majority Coset Decoding. *IEEE. Trans. On Inf. Theory*, vol. 39/3, pp. 1067-1070 (1993).

[FR] G.-L. Feng, T.R.N. Rao. Decoding Algebraic-Geometric Codes up to the Designed Minimum Distance. *IEEE Trans. On Inf. Theory*, vol. 39/1, pp. 37-45 (1993).

[FGLM] J.C. Faugere, P. Gianni, D. Lazard, T. Mora. Efficient Computation of Zero-dimensional Gröbner Bases by Change of Ordering. *J. Symbolic Computation*, 16, pp. 329-344 (1993).

[HøPe] T. Høholdt, R. Pellikaan. On the Decoding of Algebraic-Geometric Codes. *IEEE Trans. On Inf. Theory*, vol. 41/6, pp. 1589-1614 (1995).

[MMM] M. G. Marinari, H. M. Möller, T. Mora. Gröbner Bases of Ideals Defined by Functionals with an Application to Ideals of Projective Points. *Applicable Algebra in Engeneering, Communication and Computing*, vol. 4, pp. 103-145 (1993).

[MR1] K. Madlener, B. Reinert. String Rewriting and Gröbner bases – A General Approach to Monoid and Group Rings. In *Proceedings of the Workshop on Symbolic Rewriting Techniques*, Monte Verita, Birkhäuser, pp 127-180, 1995 (printed 1998).

[MR2] K. Madlener, B. Reinert. Relating Rewriting Techniques on Monoids and Rings: Congruences on Monoids and Ideals in Monoid Rings. *Theoretical Computer Sciences*, 208, pp. 3-31 (1998).

[MRM] K. Madlener, B. Reinert, T. Mora. A note on Nielsen Reduction and Coset Enumeration. *Proc. ISSAC 98*, pp. 171-178 (1998).

[PeWe] W. W. Peterson, E. J. Jr. Weldon. *Error-Correcting Codes (2nd ed.)*. MIT Press, Cambridge, Massachusetts, London. England (1972).

[Pre] O. Pretzel. *Codes and Algebraic Curves*. Clarendon Press. Oxford (1998).

[SJH] S. Sakata, H. E. Jensen, T. Høholdt. Generalized berlekamp-Massey Decoding of Algebraic-Geometric Codes up to Half the Feng-Rao Bound. *IEEE Trans. On Inf. Theory*, vol. 41/6, pp. 1762-1768 (1995).

[SJMJH] S. Sakata, J. Justesen, Y. Madelung, H. E. Jensen, T. Høholdt. Fast Decoding of Algebraic-Geometric Codes up to the designed Minimum Distance. *IEEE Trans. On Inf. Theory*, vol 41/5, pp. 1672-1677 (1995).

[Sak] Shojiro Sakata. Gröbner Bases and Coding Theory. In *Gröbner Bases and Applications (Proc. of the Conference 33 Years of Gröbner Bases)*. B. Buchberger, F. Winkler (eds.). Cambridge University Press, London Mathematical Society Lecture Notes Series, vol. 251, pp. 205-220 (1998).

[JLJH] J. Justesen, J. Larsen, H. E. Jensen, T. Høholdt. Fast Decoding of Codes from Algebraic Plane Curves. *IEEE Trans. On Inf. Theory*, vol. 38/1, pp. 111-119 (1992).

[vL] J. H. van Lint. *Introduction to Coding Theory (2nd ed.)*. Springer-Verlag, Berlin (1992).