Another description of the Berlekamp-Massey algorithm

W. Henkel

Indexing terms: Algorithms, Decoding, Error and error analysis

Abstract: A new explanation of the Berlekamp-Massey algorithm is given using a method that is not based on the usual description as shift register synthesis but in terms of matrices. It has the advantage of being more didactic and it reveals some properties that cannot be deduced so easily from Massey's interpretation.

1 Introduction

The Berlekamp-Massey algorithm (BMA) [1, 2] is a method for solving Toeplitz systems that is often used in particular for decoding Reed-Solomon (RS) and Bose-Chaudhuri–Hocquenghem (BCH) codes. Regarding the BMA as the synthesis of a shift-register [2] made it easier to understand and that is surely one of the main reasons for its wide acceptance and distribution. Nevertheless, it is regarded by many as very confusing. This paper shows that an illustration of the steps of the algorithm using a description based on matrices is a more didactic guide to understanding. It will indeed lead to a deeper knowledge of the structure of the method and to the easy deduction of important, and in some cases hitherto unknown, properties of the algorithm.

First, following Massey's interpretation, the relation with the problem of shift-register synthesis is shown briefly.

2 Short description of the shift-register problem

In the decoding of RS-codes, errors are located from the zeros of the so-called error locator polynomial $C(x)$. These zeros are given as the powers of a primitive element in a Galois field (or, in the complex domain, as powers of $z = e^{j2\pi/N}$, where N is the length of the codeword). If we denote the set of error locations by $\mathbb{F} = \{i \mid C(z^{-i}) = 0\}(|\mathbb{F}| = L)$, the error locator polynomial is given by

$$C(x) = 1 + C_1 x^1 + C_2 x^2 + \cdots + C_L x^L = C_L \prod_{i \in \mathbb{F}} (x - z^{-i}) \quad (1)$$

There is a well-known relation between the coefficients of the error locator polynomial and the components of the syndrome (in the 'frequency domain'), called 'key equation':

$$S_j = -\sum_{i=1}^{L} C_i S_{j-i} \qquad j = L, L+1, \ldots, M-1 \quad (2)$$

Paper 6673I (E8, E16), first received 4th March 1988 and in revised form 4th January 1989

The author is with the Institut für Netzwerk- und Signaltheorie, Technische Hochschule Darmstadt, D-6100 Darmstadt, Merckstraße 25, Federal Republic of Germany

First, let L, the degree of $C(x)$, be unknown. However, L has to be equal to the number of errors e which have actually occurred: $L = e \leq M/2$, where M is the number of parity symbols. Eqn. 2 may be represented by the shift register shown in Fig. 1.

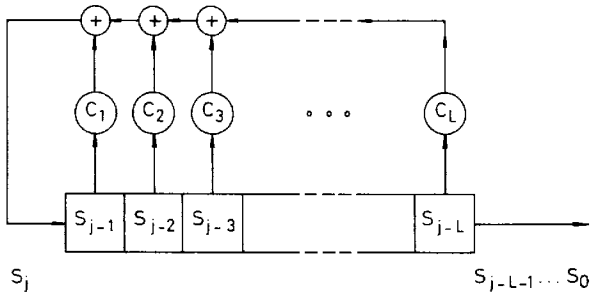

Fig. 1 *Shift-register representation of the 'key equation'*

The BMA leads to the polynomial of smallest possible degree (shortest shift register), as is required to determine the error locations. This is why it increases the length of the shift register successively, increasing L from zero until it equals the number of errors e which have actually occurred. Either a change of the coefficients C_i or a length change is initiated if eqn. 2 is not fulfilled, i.e. if there is a discrepancy

$$d_n = S_n + \sum_{i=1}^{L_n} C_i S_{n-i} \quad (3)$$

not equal to zero. The new shift register coefficients follow by addition of the existing coefficients to those which existed before the last length change. These earlier coefficients are shifted in the direction given by the shift-register by an amount such that it reaches the same part of the syndrome as before. Consequently, the same discrepancy results as that which led to the last length change. It follows that the new error locator polynomial is now given by

$$C_{n+1}(x) = C_n(x) - \frac{d_n}{d_m} x^{n-m} C_m(x) \quad (4)$$

where $n - m$ is the amount (length) of shift, d_n is the new discrepancy and d_m is the discrepancy before the last length change. A length change is only necessary if $d_n \neq 0 \wedge 2L_n \leq n$. For more details concerning this way of describing Berlekamp's algorithm the reader is referred to the paper by Massey [2].

Before presenting the new explanation, all steps of the BMA are listed in detail. For simplicity, Massey's paper, in which the steps are given in the form of a program, is quoted.

(1) $1 \rightarrow C(D) \quad 1 \rightarrow B(D) \quad 1 \rightarrow x$
$0 \rightarrow L \quad 1 \rightarrow b \quad 0 \rightarrow N$
(2) if $N = n$, stop. Otherwise compute
$d = s_N + \sum_{i=1}^{L} c_i s_{N-i}$
(3) if $d = 0$, then $x + 1 \rightarrow x$, and go to (6)
(4) if $d \neq 0$ and $2L > N$, then
$C(D) - db^{-1}D^x B(D) \rightarrow C(D)$
$x + 1 \rightarrow x$
and go to (6)
(5) if $d \neq 0$ and $2L \leqslant N$, then
$C(D) \rightarrow T(D)$
$C(D) - db^{-1}D^x B(D) \rightarrow C(D)$
$N + 1 - L \rightarrow L$
$T(D) \rightarrow B(D)$
$d \rightarrow b$
$1 \rightarrow x$
(6) $N + 1 \rightarrow N$ and return to (2)

In this program $C(D)$ is the actual error locator polynomial, $B(D)$ is the error locator polynomial before the last length change, x is the amount of shift, L is the length of the shift register, d is the actual discrepancy, b is the discrepancy before the last length change, and $N + 1$ is the number of components of the syndrome already used.

3 The new matrix description

First, the problem is given in the form of a system of equations. For this purpose, we again start with eqn. 2 and $L = e$ to write down the final set of equations to be solved:

$$S_j = -\sum_{i=1}^{e} C_i S_{j-i} \tag{5}$$

This becomes

$$(1, C_1, C_2, \ldots, C_e)\begin{pmatrix} S_e & S_{e+1} & \cdot & \cdot & S_{2e} & \cdot & S_{M-1} \\ S_{e-1} & S_e & \ddots & & \cdot & & \cdot \\ \cdot & S_{e-1} & \ddots & \ddots & \cdot & \cdot & \cdot \\ \cdot & & \ddots & \ddots & S_{e+1} & & \cdot \\ S_0 & \cdot & & S_{e-1} & S_e & \cdot & S_{M-1-e} \end{pmatrix} = (0, \ldots, 0) \tag{6}$$

The matrix has a Toeplitz-structure.

The recursive procedure starts with the lowest possible degree of the error locator polynomial and the shortest vector $\boldsymbol{C}$, with $L = 0$, using the smallest possible submatrix:

$$(1)(S_0) = (S_0) =: (d_0^1) \tag{7}$$

which means that the first discrepancy is equal to S_0. Considering now the next step, it is obvious that a length change is necessary, i.e. the next greater submatrix has to be used and the additional component of the vector $\boldsymbol{C}$ is set to zero.

$$(1, 0)\begin{pmatrix} S_1 & S_2 \\ S_0 & S_1 \end{pmatrix} = (d_0^2, d_1^2) \tag{8}$$

Of course, this is a quite reasonable procedure, because it means that no (other) error has occurred. In particular, it should be borne in mind that the polynomial $C(x)$ of smallest degree has to be found. On the right-hand side of the set of linear equations there follow components of the vector which may differ from zero. However, the solution should generate zeros. Therefore, initially we try to force the first component of the right-hand side to zero. The system before the last length change (eqn. 7) is already known. If one uses the vector (0, 1) instead of (1), S_0 would again appear at the first location of the right-hand side. To reach zero in the first component, there follows for $\boldsymbol{C}$:

$$\left[(1, 0) - \frac{d_0^2}{S_0}(0, 1)\right]\begin{pmatrix} S_1 & S_2 \\ S_0 & S_1 \end{pmatrix} = (0, d_1^3) \tag{9}$$

If $d_1^3 \neq 0$, the submatrix is not singular [$\det(\cdot) \neq 0$] and a length change is necessary. Even if $d_1^3 = 0$, $\boldsymbol{C}$ has to be enlarged to check the next greater submatrix for singularity. It should be noticed that an enlargement of the vector $\boldsymbol{C}$ is to be regarded as a length change in the sense of the BMA only if the corresponding discrepancy is not equal to zero, i.e. only if the next operation step is really a lengthening with components different from zero.

Having described the beginning of the algorithm, the main operations of the BMA that make use of the special structure of the matrix will now be pointed out.

As could be seen from the initialisation, the lengthening of vectors by appending zeros together with an enlargement of the corresponding sub-Toeplitz matrix tance. An explanation of the effects of adding zeros to one end of the vector $\boldsymbol{C}$ follows.

Adding zeros to the right: The effect of lengthening the vector $\boldsymbol{C}$ by appending a zero to the right together with an enlargement of the corresponding sub-Toeplitz matrix is that the right-hand side of the equation is shifted to the left by one step and two new components are added at the rightmost location:

$$(1, C_1, C_2, \ldots, C_l)\begin{pmatrix} S_l & S_{2l} \\ & \ddots \\ S_0 & S_l \end{pmatrix} = (\rho_0, \rho_1, \rho_2, \ldots, \rho_l) \tag{10}$$

$$(1, C_1, C_2, \ldots, C_l, 0)\begin{pmatrix} S_{l+1} & S_{2(l+1)} \\ & \ddots \\ S_0 & S_{l+1} \end{pmatrix} = (\rho_1, \rho_2, \ldots, \rho_l, \rho_{l+1}, \rho_{l+2}) \tag{11}$$

Adding zeros to the left: A lengthening by filling up with zero at the left leaves the right-hand side unchanged and adds one component to the rightmost location:

$$(1, C_1, C_2, \ldots, C_l)\begin{pmatrix} S_l & S_{2l} \\ & \ddots \\ S_0 & S_l \end{pmatrix} = (\rho_0, \rho_1, \rho_2, \ldots, \rho_l) \tag{12}$$

$$(0, 1, C_1, C_2, \ldots, C_l)\begin{pmatrix} S_{l+1} & S_{2(l+1)} \\ & \ddots \\ S_0 & S_{l+1} \end{pmatrix} = (\rho_0, \rho_1, \rho_2, \ldots, \rho_l, \rho_{l+1}) \tag{13}$$

The vector before the last length change $\boldsymbol{C}_v$ (that is appended by zeros on the left) reveals the special characteristic of generating zeros from one component down to the leftmost location.

The vector C has to be enlarged by zeros on the right, i.e. the right side has to be shifted to the left as well until the first component not equal to zero is located at the position given by the right side using C_v. The following two systems of equations result:

$$(0, \ldots, 0, 1, C_{1_v}, C_{2_v}, \ldots)(\cdot) = \underbrace{(0, \ldots, 0}, d_{i_v}, d_{i+1_v}, d_{i+2_v}, \ldots) \quad (14)$$

$$(1, C_1, C_2, \ldots, 0, \ldots, 0)(\cdot) = \overbrace{(0, \ldots, 0}, d_i, d_{i+1}, d_{i+2}, \ldots) \quad (15)$$

Combining the two eqns. 14 and 15 so that the first component d_i different from zero vanishes, we get the new vector C:

$$C =: (1, C_1, C_2, \ldots, 0, \ldots, 0) - \frac{d_i}{d_{i_v}} (0, \ldots, 0, 1, C_{1_v}, C_{2_v}, \ldots) \quad (16)$$

This exactly equals the recursion eqn. 4.

One could interpret the right-hand side of the set of equations before the last length change as a 'mark' that has to be reached by enlargement of the vector C. The 'cancellation' of this component then follows.

Two examples may provide a deeper insight into the procedure. Example 2 examines the case of singular submatrices. There, one recognises that it may even be necessary to further enlarge the Toeplitz matrix with unknown components to carry on. However, only known components of the matrix are used in subsequent operations. For both examples, an odd number of syndromes has been chosen. However, it will be noticed that the last component of the syndrome is not necessary at all. It merely serves as an additional check if all errors have been found.

The reader should pay special attention to the fact that length changes occur if and only if $d_n \neq 0 \wedge 2L_n \leqslant n$.

Example 1:
GF(7); primitive element: 5; length of the syndrome: 5
Error vector: (1, 0, 0, 0, 1, 0)
Syndrome: (5, 2, 4, 5, 2)

$$(1, C_1, C_2)\begin{pmatrix} 4 & 5 & 2 \\ 2 & 4 & 5 \\ 5 & 2 & 4 \end{pmatrix} \triangleq (0, 0, 0)$$

$$(1) \cdot (5) = (5)$$

$$(1, 0)\begin{pmatrix} 2 & 4 \\ 5 & 2 \end{pmatrix} = (2, 4)$$

$$(1, 0) - \tfrac{2}{5}(0, 1) = (1, 1)$$

$$(1, 1)\begin{pmatrix} 2 & 4 \\ 5 & 2 \end{pmatrix} = (0, 6)$$

$$(1, 1, 0)\begin{pmatrix} 4 & 5 & 2 \\ 2 & 4 & 5 \\ 5 & 2 & 4 \end{pmatrix} = (6, 2, 0)$$

$$(1, 1, 0) - \tfrac{6}{5}(0, 0, 1) = (1, 1, 3)$$

$$(1, 1, 3)\begin{pmatrix} 4 & 5 & 2 \\ 2 & 4 & 5 \\ 5 & 2 & 4 \end{pmatrix} = (0, 1, 5)$$

$$(1, 1, 3) - \tfrac{1}{6}(0, 1, 1) = (1, 2, 4)$$

$$(1, 2, 4)\begin{pmatrix} 4 & 5 & 2 \\ 2 & 4 & 5 \\ 5 & 2 & 4 \end{pmatrix} = (0, 0, 0)$$

$$C(x) = 1 + 2x + 4x^2$$

The positions of errors are 0 and 4.

$$C(5^0) = 1 + 2 + 4 = 0$$

$$C(5^4) = 1 + 2 \cdot 2 + 4 \cdot 4 = 0$$

Example 2:
GF(11); primitive element: 6; length of the syndrome: 7
Error vector: (1, 0, 1, 0, 4, 0, 0, 0, 0, 0)
Syndrome: (5, 8, 4, 7, 4, 5, 8)

$$(1, C_1, C_2, C_3)\begin{pmatrix} 7 & 4 & 5 & 8 \\ 4 & 7 & 4 & 5 \\ 8 & 4 & 7 & 4 \\ 5 & 8 & 4 & 7 \end{pmatrix} \triangleq (0, 0, 0, 0)$$

$$(1) \cdot (5) = (5)$$

$$(1, 0)\begin{pmatrix} 8 & 4 \\ 5 & 8 \end{pmatrix} = (8, 4)$$

$$(1, 0) - \tfrac{8}{5}(0, 1) = (1, 5)$$

$$(1, 5)\begin{pmatrix} 8 & 4 \\ 5 & 8 \end{pmatrix} = (0, 0)$$

$$(1, 5, 0)\begin{pmatrix} 4 & 7 & 4 \\ 8 & 4 & 7 \\ 5 & 8 & 4 \end{pmatrix} = (0, 5, 6)$$

$$(1, 5, 0, 0)\begin{pmatrix} 7 & 4 & 5 & 8 \\ 4 & 7 & 4 & 5 \\ 8 & 4 & 7 & 4 \\ 5 & 8 & 4 & 7 \end{pmatrix} = (5, 6, 3, 0)$$

$$(1, 5, 0, 0) - \tfrac{5}{5}(0, 0, 0, 1) = (1, 5, 0, 10)$$

$$(1, 5, 0, 10)\begin{pmatrix} 7 & 4 & 5 & 8 \\ 4 & 7 & 4 & 5 \\ 8 & 4 & 7 & 4 \\ 5 & 8 & 4 & 7 \end{pmatrix} = (0, 9, 10, 4)$$

$$(1, 5, 0, 10) - \tfrac{9}{5}(0, 1, 5, 0) = (1, 1, 2, 10)$$

$$(1, 1, 2, 10)\begin{pmatrix} 7 & 4 & 5 & 8 \\ 4 & 7 & 4 & 5 \\ 8 & 4 & 7 & 4 \\ 5 & 8 & 4 & 7 \end{pmatrix} = (0, 0, 8, 3)$$

$$(1, 1, 2, 10, 0)\begin{pmatrix} 4 & 5 & 8 & ? & ? \\ 7 & 4 & 5 & 8 & ? \\ 4 & 7 & 4 & 5 & 8 \\ 8 & 4 & 7 & 4 & 5 \\ 5 & 8 & 4 & 7 & 4 \end{pmatrix} = (0, 8, 3, ?, ?)$$

$$(1, 1, 2, 10, 0) - \tfrac{8}{5}(0, 0, 1, 5, 0) = (1, 1, 7, 2, 0)$$

$$(1, 1, 7, 2, 0)\begin{pmatrix} 4 & 5 & 8 & ? & ? \\ 7 & 4 & 5 & 8 & ? \\ 4 & 7 & 4 & 5 & 8 \\ 8 & 4 & 7 & 4 & 5 \\ 5 & 8 & 4 & 7 & 4 \end{pmatrix} = (0, 0, 0, ?, ?)$$

$$(1, 1, 7, 2)\begin{pmatrix} 7 & 4 & 5 & 8 \\ 4 & 7 & 4 & 5 \\ 8 & 4 & 7 & 4 \\ 5 & 8 & 4 & 7 \end{pmatrix} = (0, 0, 0, 0)$$

$$C(x) = 1 + x + 7x^2 + 2x^3$$

The positions of errors are 0, 2 and 4.

$$C(6^0) = C(1) = 0$$

$$C(6^2) = C(3) = 0$$

$$C(6^4) = C(9) = 0$$

4 Initialisation of the BMA

Readers who are familiar with Massey's paper may possibly have noticed that the initial part of the algorithm, as described here, is not identical to his presentation in all details. The first steps of the BMA of example 1 will therefore be listed again in accordance with the quoted paper:

$C(D) = 1$ $\quad x = 1$

$B(D) = 1$ $\quad L = 0$

$b = 1$

$N = 0$

$(1) \cdot (5) = (5)$

$(1, 0) - \frac{5}{1}(0, 1) = (1, 2) \quad (*)$

$C(D) = 1 + 2D$ $\quad x = 1$

$B(D) = 1$ $\quad L = 1$

$b = 5$

$N = 1$

$$(1, 2)\begin{pmatrix} 2 & 4 \\ 5 & 2 \end{pmatrix} = (5, 1)$$

$(1, 2) - \frac{5}{5}(0, 1) = (1, 1)$

$C(D) = 1 + D$ $\quad x = 2$

$B(D) = 1$ $\quad L = 1$

$b = 5$

$N = 2$

$\vdots$

By comparing both presentations it will be realised that operation (*) is not really necessary. Nevertheless, it has the advantage that L always gives the length of the corresponding shift-register, which would no longer be the case if this step were to be eliminated as shown below:

$\vdots$

(5) if $d \neq 0$ and $2L \leqslant N$, then
$C(D) \rightarrow T(D)$
if $N \neq 0$ then $C(D) - db^{-1}D^x B(D) \rightarrow C(D)$
$N + 1 - L \rightarrow L$
$T(D) \rightarrow B(D)$
$d \rightarrow b$
$1 \rightarrow x$

$\vdots$

To demonstrate the clarity of the new description, the case of erasure decoding is considered briefly in the following section.

5 The BMA in the case of erasure decoding

Erasures are symbols that are received with insufficient reliability or not received at all. From the point of view of decoding, they may be regarded as errors at known locations. By analogy with the error locator polynomial, an erasure locator polynomial $\Lambda(x)$ is defined as

$$\Lambda(x) = \prod_{i \in \mathbb{E}} (x - z^{-i}) \Big/ \prod_{i \in \mathbb{E}} - z^{-i} \tag{17}$$

where $\mathbb{E}$ is the set of erasure locations ($|\mathbb{E}| = a_E$).

With this new description, it at once becomes obvious that the algorithm now starts with the subsystem

$$(1, \Lambda_1, \Lambda_2, \ldots, \Lambda_{a_E}) \cdot \begin{pmatrix} S_{a_E} & S_{2a_E} \\ & \ddots \\ S_0 & S_{a_E} \end{pmatrix} = (\rho_0, \rho_1, \rho_2, \ldots, \rho_{a_E}) \tag{18}$$

which means that $\boldsymbol{C}$ is initialised with $\boldsymbol{\Lambda}$, retaining it as a linear factor. Lengthening by appending a zero to the right side then follows, etc. The first usable discrepancy for determining the error locations is ρ_0. The syndrome length available for error location is $M - a_E$, and from this the number of correctable errors results in $\lfloor M - a_E/2 \rfloor$. Here, the shortening of the available syndrome can be realised without further consideration.

Remark

It should be noted that, with the new description, the BMA leads to a so-called 'triangular square-root factorisation' of the inverse syndrome matrix (if all discrepancies are not equal to zero), consisting of a triangular matrix out of the recursive error locators $\boldsymbol{C}$ and its transpose, and a diagonal matrix out of the corresponding discrepancies [3].

6 Summary

The operation steps of the BMA have been illustrated by studying the corresponding sub-Toeplitz matrices. The advantage of this description is due especially to the fact that the main operations of the algorithm appear as a consequence of the matrix structure. The illustration corresponds to the complete set of equations; this cannot be achieved with the same clarity when the BMA is described as a shift-register synthesis. It is a more didactic way of explaining the algorithm, and some properties of the algorithm can be deduced more easily.

7 References

1 BERLEKAMP, E.R.: 'Algebraic coding theory' (McGraw-Hill, New York, 1968)

2 MASSEY, J.L.: 'Shift-register synthesis and BCH decoding', *IEEE Trans.*, 1969, **IT-15**, (1), pp. 122–127

3 HENKEL, W.: 'Multiple error correction with analog codes'. 6th Int. Conference on Applied algebra, algebraic algorithms and error correcting codes (AAECC-6), Rome, 4th–8th July 1988

Another description of the Berlekamp-Massey algorithm

W. Henkel

Indexing terms: Algorithms, Decoding, Error and error analysis

Abstract: A new explanation of the Berlekamp-Massey algorithm is given using a method that is not based on the usual description as shift register synthesis but in terms of matrices. It has the advantage of being more didactic and it reveals some properties that cannot be deduced so easily from Massey's interpretation.

1 Introduction

The Berlekamp-Massey algorithm (BMA) [1, 2] is a method for solving Toeplitz systems that is often used in particular for decoding Reed-Solomon (RS) and Bose-Chaudhuri–Hocquenghem (BCH) codes. Regarding the BMA as the synthesis of a shift-register [2] made it easier to understand and that is surely one of the main reasons for its wide acceptance and distribution. Nevertheless, it is regarded by many as very confusing. This paper shows that an illustration of the steps of the algorithm using a description based on matrices is a more didactic guide to understanding. It will indeed lead to a deeper knowledge of the structure of the method and to the easy deduction of important, and in some cases hitherto unknown, properties of the algorithm.

First, following Massey's interpretation, the relation with the problem of shift-register synthesis is shown briefly.

2 Short description of the shift-register problem

In the decoding of RS-codes, errors are located from the zeros of the so-called error locator polynomial $C(x)$. These zeros are given as the powers of a primitive element in a Galois field (or, in the complex domain, as powers of $z = e^{j2\pi/N}$, where N is the length of the codeword). If we denote the set of error locations by $\mathbb{F} = \{i \mid C(z^{-i}) = 0\}(|\mathbb{F}| = L)$, the error locator polynomial is given by

$$C(x) = 1 + C_1x^1 + C_2x^2 + \cdots + C_Lx^L = C_L \prod_{i \in \mathbb{F}} (x - z^{-i}) \quad (1)$$

There is a well-known relation between the coefficients of the error locator polynomial and the components of the syndrome (in the 'frequency domain'), called 'key equation':

$$S_j = -\sum_{i=1}^{L} C_i S_{j-i} \qquad j = L, L+1, \ldots, M-1 \quad (2)$$

First, let L, the degree of $C(x)$, be unknown. However, L has to be equal to the number of errors e which have actually occurred: $L = e \leq M/2$, where M is the number of parity symbols. Eqn. 2 may be represented by the shift register shown in Fig. 1.

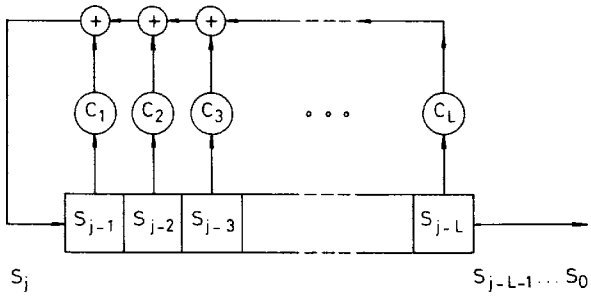

Fig. 1 *Shift-register representation of the 'key equation'*

The BMA leads to the polynomial of smallest possible degree (shortest shift register), as is required to determine the error locations. This is why it increases the length of the shift register successively, increasing L from zero until it equals the number of errors e which have actually occurred. Either a change of the coefficients C_i or a length change is initiated if eqn. 2 is not fulfilled, i.e. if there is a discrepancy

$$d_n = S_n + \sum_{i=1}^{L_n} C_i S_{n-i} \quad (3)$$

not equal to zero. The new shift register coefficients follow by addition of the existing coefficients to those which existed before the last length change. These earlier coefficients are shifted in the direction given by the shift-register by an amount such that it reaches the same part of the syndrome as before. Consequently, the same discrepancy results as that which led to the last length change. It follows that the new error locator polynomial is now given by

$$C_{n+1}(x) = C_n(x) - \frac{d_n}{d_m} x^{n-m} C_m(x) \quad (4)$$

where $n - m$ is the amount (length) of shift, d_n is the new discrepancy and d_m is the discrepancy before the last length change. A length change is only necessary if $d_n \neq 0 \wedge 2L_n \leq n$. For more details concerning this way of describing Berlekamp's algorithm the reader is referred to the paper by Massey [2].

Before presenting the new explanation, all steps of the BMA are listed in detail. For simplicity, Massey's paper, in which the steps are given in the form of a program, is quoted.

Paper 6673I (E8, E16), first received 4th March 1988 and in revised form 4th January 1989

The author is with the Institut für Netzwerk- und Signaltheorie, Technische Hochschule Darmstadt, D-6100 Darmstadt, Merckstraße 25, Federal Republic of Germany

(1) $1 \rightarrow C(D) \quad 1 \rightarrow B(D) \quad 1 \rightarrow x$
$0 \rightarrow L \quad 1 \rightarrow b \quad 0 \rightarrow N$

(2) if $N = n$, stop. Otherwise compute
$d = s_N + \sum_{i=1}^{L} c_i s_{N-i}$

(3) if $d = 0$, then $x + 1 \rightarrow x$, and go to (6)

(4) if $d \neq 0$ and $2L > N$, then
$C(D) - db^{-1}D^x B(D) \rightarrow C(D)$
$x + 1 \rightarrow x$
and go to (6)

(5) if $d \neq 0$ and $2L \leqslant N$, then
$C(D) \rightarrow T(D)$
$C(D) - db^{-1}D^x B(D) \rightarrow C(D)$
$N + 1 - L \rightarrow L$
$T(D) \rightarrow B(D)$
$d \rightarrow b$
$1 \rightarrow x$

(6) $N + 1 \rightarrow N$ and return to (2)

In this program $C(D)$ is the actual error locator polynomial, $B(D)$ is the error locator polynomial before the last length change, x is the amount of shift, L is the length of the shift register, d is the actual discrepancy, b is the discrepancy before the last length change, and $N + 1$ is the number of components of the syndrome already used.

3 The new matrix description

First, the problem is given in the form of a system of equations. For this purpose, we again start with eqn. 2 and $L = e$ to write down the final set of equations to be solved:

$$S_j = -\sum_{i=1}^{e} C_i S_{j-i} \tag{5}$$

This becomes

$$(1, C_1, C_2, \ldots, C_e)\begin{pmatrix} S_e & S_{e+1} & \cdot & \cdot & S_{2e} & \cdot & S_{M-1} \\ S_{e-1} & S_e & \ddots & & \cdot & & \cdot \\ \cdot & S_{e-1} & \ddots & \ddots & \cdot & & \cdot \\ \cdot & & \ddots & \ddots & S_{e+1} & & \cdot \\ S_0 & \cdot & & S_{e-1} & S_e & \cdot & S_{M-1-e} \end{pmatrix} = (0, \ldots, 0) \tag{6}$$

The matrix has a Toeplitz-structure.

The recursive procedure starts with the lowest possible degree of the error locator polynomial and the shortest vector $\boldsymbol{C}$, with $L = 0$, using the smallest possible submatrix:

$$(1)(S_0) = (S_0) =: (d_0^1) \tag{7}$$

which means that the first discrepancy is equal to S_0. Considering now the next step, it is obvious that a length change is necessary, i.e. the next greater submatrix has to be used and the additional component of the vector $\boldsymbol{C}$ is set to zero.

$$(1, 0)\begin{pmatrix} S_1 & S_2 \\ S_0 & S_1 \end{pmatrix} = (d_0^2, d_1^2) \tag{8}$$

Of course, this is a quite reasonable procedure, because it means that no (other) error has occurred. In particular, it should be borne in mind that the polynomial $C(x)$ of smallest degree has to be found. On the right-hand side of the set of linear equations there follow components of the vector which may differ from zero. However, the solution should generate zeros. Therefore, initially we try to force the first component of the right-hand side to zero. The system before the last length change (eqn. 7) is already known. If one uses the vector (0, 1) instead of (1), S_0 would again appear at the first location of the right-hand side. To reach zero in the first component, there follows for $\boldsymbol{C}$:

$$\left[(1, 0) - \frac{d_0^2}{S_0}(0, 1)\right]\begin{pmatrix} S_1 & S_2 \\ S_0 & S_1 \end{pmatrix} = (0, d_1^3) \tag{9}$$

If $d_1^3 \neq 0$, the submatrix is not singular [$\det(\cdot) \neq 0$] and a length change is necessary. Even if $d_1^3 = 0$, $\boldsymbol{C}$ has to be enlarged to check the next greater submatrix for singularity. It should be noticed that an enlargement of the vector $\boldsymbol{C}$ is to be regarded as a length change in the sense of the BMA only if the corresponding discrepancy is not equal to zero, i.e. only if the next operation step is really a lengthening with components different from zero.

Having described the beginning of the algorithm, the main operations of the BMA that make use of the special structure of the matrix will now be pointed out.

As could be seen from the initialisation, the lengthening of vectors by appending zeros together with an enlargement of the corresponding sub-Toeplitz matrix tance. An explanation of the effects of adding zeros to one end of the vector $\boldsymbol{C}$ follows.

Adding zeros to the right: The effect of lengthening the vector $\boldsymbol{C}$ by appending a zero to the right together with an enlargement of the corresponding sub-Toeplitz matrix is that the right-hand side of the equation is shifted to the left by one step and two new components are added at the rightmost location:

$$(1, C_1, C_2, \ldots, C_l)\begin{pmatrix} S_l & S_{2l} \\ & \ddots \\ S_0 & S_l \end{pmatrix} = (\rho_0, \rho_1, \rho_2, \ldots, \rho_l) \tag{10}$$

$$(1, C_1, C_2, \ldots, C_l, 0)\begin{pmatrix} S_{l+1} & S_{2(l+1)} \\ & \ddots \\ S_0 & S_{l+1} \end{pmatrix} = (\rho_1, \rho_2, \ldots, \rho_l, \rho_{l+1}, \rho_{l+2}) \tag{11}$$

Adding zeros to the left: A lengthening by filling up with zero at the left leaves the right-hand side unchanged and adds one component to the rightmost location:

$$(1, C_1, C_2, \ldots, C_l)\begin{pmatrix} S_l & S_{2l} \\ & \ddots \\ S_0 & S_l \end{pmatrix} = (\rho_0, \rho_1, \rho_2, \ldots, \rho_l) \tag{12}$$

$$(0, 1, C_1, C_2, \ldots, C_l)\begin{pmatrix} S_{l+1} & S_{2(l+1)} \\ & \ddots \\ S_0 & S_{l+1} \end{pmatrix} = (\rho_0, \rho_1, \rho_2, \ldots, \rho_l, \rho_{l+1}) \tag{13}$$

The vector before the last length change $\boldsymbol{C}_v$ (that is appended by zeros on the left) reveals the special characteristic of generating zeros from one component down to the leftmost location.

The vector C has to be enlarged by zeros on the right, i.e. the right side has to be shifted to the left as well until the first component not equal to zero is located at the position given by the right side using C_v. The following two systems of equations result:

$$(0, \ldots, 0, 1, C_{1_v}, C_{2_v}, \ldots)(\cdot) = \underbrace{(0, \ldots, 0}, d_{i_v}, d_{i+1_v}, d_{i+2_v}, \ldots) \quad (14)$$

$$(1, C_1, C_2, \ldots, 0, \ldots, 0)(\cdot) = \overbrace{(0, \ldots, 0}, d_i, d_{i+1}, d_{i+2}, \ldots) \quad (15)$$

Combining the two eqns. 14 and 15 so that the first component d_i different from zero vanishes, we get the new vector C:

$$C = (1, C_1, C_2, \ldots, 0, \ldots, 0) - \frac{d_i}{d_{i_v}}(0, \ldots, 0, 1, C_{1_v}, C_{2_v}, \ldots) \quad (16)$$

This exactly equals the recursion eqn. 4.

One could interpret the right-hand side of the set of equations before the last length change as a 'mark' that has to be reached by enlargement of the vector C. The 'cancellation' of this component then follows.

Two examples may provide a deeper insight into the procedure. Example 2 examines the case of singular submatrices. There, one recognises that it may even be necessary to further enlarge the Toeplitz matrix with unknown components to carry on. However, only known components of the matrix are used in subsequent operations. For both examples, an odd number of syndromes has been chosen. However, it will be noticed that the last component of the syndrome is not necessary at all. It merely serves as an additional check if all errors have been found.

The reader should pay special attention to the fact that length changes occur if and only if $d_n \neq 0 \wedge 2L_n \leqslant n$.

Example 1:
GF(7); primitive element: 5; length of the syndrome: 5
Error vector: (1, 0, 0, 0, 1, 0)
Syndrome: (5, 2, 4, 5, 2)

$$(1, C_1, C_2)\begin{pmatrix}4&5&2\\2&4&5\\5&2&4\end{pmatrix} \stackrel{!}{=} (0, 0, 0)$$

$$(1) \cdot (5) = (5)$$

$$(1, 0)\begin{pmatrix}2&4\\5&2\end{pmatrix} = (2, 4)$$

$$(1, 0) - \tfrac{2}{5}(0, 1) = (1, 1)$$

$$(1, 1)\begin{pmatrix}2&4\\5&2\end{pmatrix} = (0, 6)$$

$$(1, 1, 0)\begin{pmatrix}4&5&2\\2&4&5\\5&2&4\end{pmatrix} = (6, 2, 0)$$

$$(1, 1, 0) - \tfrac{6}{5}(0, 0, 1) = (1, 1, 3)$$

$$(1, 1, 3)\begin{pmatrix}4&5&2\\2&4&5\\5&2&4\end{pmatrix} = (0, 1, 5)$$

$$(1, 1, 3) - \tfrac{1}{6}(0, 1, 1) = (1, 2, 4)$$

$$(1, 2, 4)\begin{pmatrix}4&5&2\\2&4&5\\5&2&4\end{pmatrix} = (0, 0, 0)$$

$$C(x) = 1 + 2x + 4x^2$$

The positions of errors are 0 and 4.

$$C(5^0) = 1 + 2 + 4 = 0$$

$$C(5^4) = 1 + 2 \cdot 2 + 4 \cdot 4 = 0$$

Example 2:
GF(11); primitive element: 6; length of the syndrome: 7
Error vector: (1, 0, 1, 0, 4, 0, 0, 0, 0, 0)
Syndrome: (5, 8, 4, 7, 4, 5, 8)

$$(1, C_1, C_2, C_3)\begin{pmatrix}7&4&5&8\\4&7&4&5\\8&4&7&4\\5&8&4&7\end{pmatrix} \stackrel{!}{=} (0, 0, 0, 0)$$

$$(1) \cdot (5) = (5)$$

$$(1, 0)\begin{pmatrix}8&4\\5&8\end{pmatrix} = (8, 4)$$

$$(1, 0) - \tfrac{8}{5}(0, 1) = (1, 5)$$

$$(1, 5)\begin{pmatrix}8&4\\5&8\end{pmatrix} = (0, 0)$$

$$(1, 5, 0)\begin{pmatrix}4&7&4\\8&4&7\\5&8&4\end{pmatrix} = (0, 5, 6)$$

$$(1, 5, 0, 0)\begin{pmatrix}7&4&5&8\\4&7&4&5\\8&4&7&4\\5&8&4&7\end{pmatrix} = (5, 6, 3, 0)$$

$$(1, 5, 0, 0) - \tfrac{5}{5}(0, 0, 0, 1) = (1, 5, 0, 10)$$

$$(1, 5, 0, 10)\begin{pmatrix}7&4&5&8\\4&7&4&5\\8&4&7&4\\5&8&4&7\end{pmatrix} = (0, 9, 10, 4)$$

$$(1, 5, 0, 10) - \tfrac{9}{5}(0, 1, 5, 0) = (1, 1, 2, 10)$$

$$(1, 1, 2, 10)\begin{pmatrix}7&4&5&8\\4&7&4&5\\8&4&7&4\\5&8&4&7\end{pmatrix} = (0, 0, 8, 3)$$

$$(1, 1, 2, 10, 0)\begin{pmatrix}4&5&8&?&?\\7&4&5&8&?\\4&7&4&5&8\\8&4&7&4&5\\5&8&4&7&4\end{pmatrix} = (0, 8, 3, ?, ?)$$

$$(1, 1, 2, 10, 0) - \tfrac{8}{5}(0, 0, 1, 5, 0) = (1, 1, 7, 2, 0)$$

$$(1, 1, 7, 2, 0)\begin{pmatrix}4&5&8&?&?\\7&4&5&8&?\\4&7&4&5&8\\8&4&7&4&5\\5&8&4&7&4\end{pmatrix} = (0, 0, 0, ?, ?)$$

$$(1, 1, 7, 2)\begin{pmatrix} 7 & 4 & 5 & 8 \\ 4 & 7 & 4 & 5 \\ 8 & 4 & 7 & 4 \\ 5 & 8 & 4 & 7 \end{pmatrix} = (0, 0, 0, 0)$$

$$C(x) = 1 + x + 7x^2 + 2x^3$$

The positions of errors are 0, 2 and 4.

$$C(6^0) = C(1) = 0$$

$$C(6^2) = C(3) = 0$$

$$C(6^4) = C(9) = 0$$

4 Initialisation of the BMA

Readers who are familiar with Massey's paper may possibly have noticed that the initial part of the algorithm, as described here, is not identical to his presentation in all details. The first steps of the BMA of example 1 will therefore be listed again in accordance with the quoted paper:

$C(D) = 1$ $\quad x = 1$

$B(D) = 1$ $\quad L = 0$

$b = 1$

$N = 0$

$(1) \cdot (5) = (5)$

$(1, 0) - \frac{5}{1}(0, 1) = (1, 2)$ (*)

$C(D) = 1 + 2D$ $\quad x = 1$

$B(D) = 1$ $\quad L = 1$

$b = 5$

$N = 1$

$$(1, 2)\begin{pmatrix} 2 & 4 \\ 5 & 2 \end{pmatrix} = (5, 1)$$

$(1, 2) - \frac{5}{5}(0, 1) = (1, 1)$

$C(D) = 1 + D$ $\quad x = 2$

$B(D) = 1$ $\quad L = 1$

$b = 5$

$N = 2$

$\vdots$

By comparing both presentations it will be realised that operation (*) is not really necessary. Nevertheless, it has the advantage that L always gives the length of the corresponding shift-register, which would no longer be the case if this step were to be eliminated as shown below:

$\vdots$

(5) if $d \neq 0$ and $2L \leqslant N$, then
$C(D) \rightarrow T(D)$
if $N \neq 0$ then $C(D) - db^{-1}D^x B(D) \rightarrow C(D)$
$N + 1 - L \rightarrow L$
$T(D) \rightarrow B(D)$
$d \rightarrow b$
$1 \rightarrow x$

$\vdots$

To demonstrate the clarity of the new description, the case of erasure decoding is considered briefly in the following section.

5 The BMA in the case of erasure decoding

Erasures are symbols that are received with insufficient reliability or not received at all. From the point of view of decoding, they may be regarded as errors at known locations. By analogy with the error locator polynomial, an erasure locator polynomial $\Lambda(x)$ is defined as

$$\Lambda(x) = \prod_{i \in \mathbb{E}} (x - z^{-i}) \Big/ \prod_{i \in \mathbb{E}} - z^{-i} \tag{17}$$

where $\mathbb{E}$ is the set of erasure locations ($|\mathbb{E}| = a_E$).

With this new description, it at once becomes obvious that the algorithm now starts with the subsystem

$$(1, \Lambda_1, \Lambda_2, \ldots, \Lambda_{a_E}) \cdot \begin{pmatrix} S_{a_E} & S_{2a_E} \\ & \ddots \\ S_0 & S_{a_E} \end{pmatrix} = (\rho_0, \rho_1, \rho_2, \ldots, \rho_{a_E}) \tag{18}$$

which means that C is initialised with Λ, retaining it as a linear factor. Lengthening by appending a zero to the right side then follows, etc. The first usable discrepancy for determining the error locations is ρ_0. The syndrome length available for error location is $M - a_E$, and from this the number of correctable errors results in $\lfloor M - a_E/2 \rfloor$. Here, the shortening of the available syndrome can be realised without further consideration.

Remark

It should be noted that, with the new description, the BMA leads to a so-called 'triangular square-root factorisation' of the inverse syndrome matrix (if all discrepancies are not equal to zero), consisting of a triangular matrix out of the recursive error locators C and its transpose, and a diagonal matrix out of the corresponding discrepancies [3].

6 Summary

The operation steps of the BMA have been illustrated by studying the corresponding sub-Toeplitz matrices. The advantage of this description is due especially to the fact that the main operations of the algorithm appear as a consequence of the matrix structure. The illustration corresponds to the complete set of equations; this cannot be achieved with the same clarity when the BMA is described as a shift-register synthesis. It is a more didactic way of explaining the algorithm, and some properties of the algorithm can be deduced more easily.

7 References

1 BERLEKAMP, E.R.: 'Algebraic coding theory' (McGraw-Hill, New York, 1968)

2 MASSEY, J.L.: 'Shift-register synthesis and BCH decoding', *IEEE Trans.*, 1969, **IT-15**, (1), pp. 122–127

3 HENKEL, W.: 'Multiple error correction with analog codes'. 6th Int. Conference on Applied algebra, algebraic algorithms and error correcting codes (AAECC-6), Rome, 4th–8th July 1988