

In this file you find the English version starting on the page numbered E1.

The Berlekamp-Massey Algorithm revisited

Then the French version begins on the page numbered F1.

L'algorithme de Berlekamp-Massey revisité

The English version of this paper appeared in AAECC **17** 1 (2006), 75–82. Received: 14 September 2005 / Published online: 9 March 2006.

Note. The paper was submitted to *Mathematics of Computations* in 2004. The referee wrote that our algorithm was well known but did not give any reference in the literature.

The Berlekamp-Massey Algorithm revisited

Nadia Ben Atti (*), Gema M. Diaz-Toca (†) Henri Lombardi (‡)

2005

Abstract

We propose a slight modification of the Berlekamp-Massey Algorithm for obtaining the minimal polynomial of a given linearly recurrent sequence. Such a modification enables to explain it in a simpler way and to adapt it to lazy evaluation.

MSC 2000: 68W30, 15A03

Key words: Berlekamp-Massey Algorithm. Linearly recurrent sequences.

Note: This paper appeared in AAECC **17** 1 (2006), 75–82. Received: 14 September 2005 / Published online: 9 March 2006

1 Introduction: The usual Berlekamp-Massey algorithm

Let \mathbb{K} be an arbitrary field. Given a linearly recurrent sequence, denoted by $\mathcal{S}(x) = \sum_{i=0}^{\infty} a_i x^i$, $a_i \in \mathbb{K}$, we wish to compute its minimal polynomial, denoted by $P(x)$. Recall that if $P(x)$ is given by $P(x) = \sum_{i=0}^d p_i x^i$ denotes such polynomial, then $P(x)$ is the polynomial of the smallest degree such that $\sum_{i=0}^d p_i a_{j+i} = 0$, for all j in \mathbb{N} .

Let suppose that the minimal polynomial of $\mathcal{S}(x)$ has degree bound n . Under such hypothesis, the Berlekamp-Massey Algorithm only requires the first $2n$ coefficients of $\mathcal{S}(x)$ in order to compute the minimal polynomial. Such coefficients define the polynomial $S = \sum_{i=0}^{2n-1} a_i x^i$.

A large literature can be consulted nowadays in relation to the Berlekamp's Algorithm. The (original) Berlekamp's Algorithm was created for decoding Bose-Chaudhuri-Hocquenghem (BCH) codes in 1968 (see [Berlekamp \(1984\)](#)). One year later, the original version of this algorithm has been simplified by Massey (see [Massey \(1969\)](#)). The similarity of the algorithm to the extended Euclidean Algorithm can be found in several articles, for instance, in [Cheng \(1984\)](#), [Dornstetter \(1987\)](#), [Mills \(1975\)](#), [Sugiyama et al. \(1975\)](#) and [Welch and Scholtz \(1979\)](#). Some more recent interpretations of the Berlekamp-Massey Algorithm in terms of Hankel Matrices and Padé approximations can be found in [Jonckheere and Ma \(1989\)](#) and [Pan \(2000\)](#).

The usual interpretation of the Berlekamp-Massey Algorithm for obtaining $P(x)$ is expressed in pseudocode in Algorithm 1.

* Equipe de Mathématiques, CNRS UMR 6623, UFR des Sciences et Techniques, Université de Franche-Comté, 25 030 Besançon cedex, France. nadia.benatti@ensi.rnu.tn

† Dpto. de Matemáticas Aplicada. Universidad de Murcia, Spain. gema.diaz@um.es, partially supported by the Galois Theory and Explicit Methods in Arithmetic Project HPRN-CT-2000-00114

‡ Equipe de Mathématiques, CNRS UMR 6623, UFR des Sciences et Techniques, Université de Franche-Comté, 25 030 Besançon cedex, France, henri.lombardi@univ-fcomte.fr, partially supported by the European Union funded project RAAG CT-2001-00271

Algorithm 1. The Usual Berlekamp-Massey Algorithm

Input: $n \in \mathbb{N}$. The first $2n$ coefficients of a linearly recurrent sequence defined over \mathbb{K} , given by the list $[a_0, a_1, \dots, a_{2n-1}]$. The minimal polynomial has degree bound n .

Output : The minimal polynomial P of the sequence.

Start

Local variables : $R, R_0, R_1, V, V_0, V_1, Q$: polynomials in x

initialization

$R_0 := x^{2n} ; R_1 := \sum_{i=0}^{2n-1} a_i x^i ; V_0 = 0 ; V_1 = 1 ;$

loop

while $n \leq \deg(R_1)$ **do**

$(Q, R) :=$ quotient and remainder of R_0 divided by R_1 ;

$V := V_0 - Q V_1$;

$V_0 := V_1$; $V_1 := V$; $R_0 := R_1$; $R_1 := R$;

end while

exit

$d := \max(\deg(V_1), 1 + \deg(R_1))$; $P := x^d V_1(1/x)$; Return $P := P/\text{leadcoeff}(P)$.

End.

In practice, we must apply the simplification of the extended Euclidean Algorithm given in Dornstetter (1987), to find exactly the Berlekamp-Massey Algorithm. Such simplification is based on the fact that initial R_0 is equal to x^{2n} .

Although Algorithm 1 is not complicated, it seems to be no easy to find a direct and transparent explanation for the determination of the degree of P . In the literature, we think there is a little confusion with the different definitions of minimal polynomial and with the different ways of defining the sequence. Here, we introduce a slight modification of the algorithm which makes it more comprehensible and natural. We did not find in the literature such a modification before the first submission of this article (May 2004). However, we would like to add that you can also find it in Shoup (2005), published in 2005, without any reference.

2 Some good reasons to modify the usual algorithm

By the one hand, as it can be observed at the end of Algorithm 1, we have to compute the (nearly) reverse polynomial of V_1 , in order to obtain the right polynomial. The following example helps us to understand what happens:

$$n = d = 3,$$

$$S = a_0 + a_1x + a_2x^2 + a_3x^3 + a_4x^4 + a_5x^5 = 1 + 2x + 7x^2 - 9x^3 + 2x^4 + 7x^5,$$

$$\text{Algorithm 1}(3, [1, 2, 7, -9, 2, 7]) \Rightarrow P = x + x^2 + x^3,$$

$$\text{with } V_1 = v_0 + v_1x + v_2x^2 = 49/67(1 + x + x^2),$$

$$\text{and } R \text{ such that } SV_1 = R \bmod x^6, \deg(R) = 2$$

which implies that

$$\text{coeff}(SV_1, x, 3) = a_1v_2 + a_2v_1 + a_3v_0 = 2v_2 + 7v_1 - 9v_0 = 0,$$

$$\text{coeff}(SV_1, x, 4) = a_2v_2 + a_3v_1 + a_4v_0 = 7v_2 - 9v_1 + 2v_0 = 0,$$

$$\text{coeff}(SV_1, x, 5) = a_3v_2 + a_4v_1 + a_5v_0 = -9v_2 + 2v_1 + 7v_0 = 0.$$

Hence, the right degree of P is given by the degree of the last R_1 plus one because x divides P . Observe that $a_0v_2 + a_1v_1 + a_2v_0 = 490/67 \neq 0$. We would like to obtain directly the desired polynomial from V_1 .

Moreover, by the other hand, in Algorithm 1 all the first $2n$ coefficients are required to start the usual algorithm, where n only provides a degree bound for the minimal polynomial. Consequently, it may be possible that the true degree of P is much smaller than n and so, less coefficients of the sequence are required to obtain the wanted polynomial.

So, we suggest a more natural, efficient and direct way to obtain P . Our idea is to consider the polynomial $\widehat{S} = \sum_{i=0}^{2n-1} a_i x^{2n-1-i}$ as the initial R_1 . Observe that in this case, using the same notation as in Algorithm 1, the same example shows that it is not necessary to reverse the polynomial V_1 at the end of the algorithm.

$$\begin{aligned} n &= d = 3, \\ \widehat{S} &= a_0x^5 + a_1x^4 + a_2x^3 + a_3x^2 + a_4x + a_5 = x^5 + 2x^4 + 7x^3 - 9x^2 + 2x + 7, \end{aligned}$$

$$\text{Algorithm 2 } (3, [1, 2, 7, -9, 2, 7]) \Rightarrow P = x + x^2 + x^3,$$

$$\text{with } V_1 = v_0 + v_1x + v_2x^2 + v_3x^3 = -9/670(x + x^2 + x^3),$$

$$\text{and } R \text{ such that } \widehat{S}V_1 = R \bmod x^6, \deg(R) = 2$$

which implies that

$$\begin{aligned} \text{coeff}(\widehat{S}V_1, x, 3) &= a_2v_0 + a_3v_1 + a_4v_2 + a_5v_3 = -9v_1 + 2v_2 + 7v_3 = 0, \\ \text{coeff}(\widehat{S}V_1, x, 4) &= a_1v_0 + a_2v_1 + a_3v_2 + a_4v_3 = 7v_1 - 9v_2 + 2v_3 = 0, \\ \text{coeff}(\widehat{S}V_1, x, 5) &= a_0v_0 + a_1v_1 + a_2v_2 + a_3v_3 = 2v_1 + 7v_2 - 9v_3 = 0. \end{aligned}$$

Furthermore, when $n \gg \deg(P)$, the algorithm can admit a lazy evaluation. In other words, the algorithm can be initiated with less coefficients than $2n$ and if the outcome does not provide the wanted polynomial, we increase the number of coefficients but remark that it is not necessary to initiate again the algorithm because we can take advantages of the computations done before. We will explain this application of the algorithm in Section 3.

Next, we introduce our modified Berlekamp-Massey Algorithm in pseudocode (Algorithm 2):

Algorithm 2. Modified Berlekamp-Massey Algorithm

Input: $n \in \mathbb{N}$. The first $2n$ coefficients of a linearly recurrent sequence defined over \mathbb{K} , given by the list $[a_0, a_1, \dots, a_{2n-1}]$. The minimal polynomial has degree bound n .

Output : The minimal polynomial P of the sequence.

Start

Local variables : $R, R_0, R_1, V, V_0, V_1, Q$: polynomials in x ; $m = 2n - 1$: integer.

initialization

$m := 2n - 1$; $R_0 := x^{2n}$; $R_1 := \sum_{i=0}^m a_{m-i} x^i$; $V_0 = 0$; $V_1 = 1$;

loop

while $n \leq \deg(R_1)$ **do**

$(Q, R) :=$ quotient and remainder of R_0 divided by R_1 ;

$V := V_0 - Q V_1$;

$V_0 := V_1$; $V_1 := V$; $R_0 := R_1$; $R_1 := R$;

end while

exit

Return $P := V_1/\text{lc}(V_1)$;

End.

Now we prove our result. Let $\underline{a} = (a_n)_{n \in \mathbb{N}}$ be an arbitrary list and $i, r, p \in \mathbb{N}$. Let $H_{i,r,p}^{\underline{a}}$ denote the following Hankel matrix of order $r \times p$,

$$H_{i,r,p}^{\underline{a}} = \begin{bmatrix} a_i & a_{i+1} & a_{i+2} & \dots & a_{i+p-1} \\ a_{i+1} & a_{i+2} & & & a_{i+p} \\ a_{i+2} & & & & \\ \vdots & & & & \vdots \\ a_{i+r-1} & a_{i+r} & \dots & \dots & a_{i+r+p-2} \end{bmatrix}$$

and let $P^{\underline{a}}(x)$ be the minimal polynomial of \underline{a} .

The next proposition shows the well known relation between the rank of Hankel matrix and the sequence.

Proposition 2.1. *Let \underline{a} be a linearly recurrent sequence. If \underline{a} has a generating polynomial of degree $\leq n$, then the degree d of its minimal polynomial $P^{\underline{a}}$ is equal to the rank of the Hankel matrix*

$$H_{0,n,n}^{\underline{a}} = \begin{bmatrix} a_0 & a_1 & a_2 & \dots & a_{n-2} & a_{n-1} \\ a_1 & a_2 & & \ddots & a_{n-1} & a_n \\ a_2 & & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ a_{n-2} & a_{n-1} & \dots & \dots & a_{2n-2} & a_{2n-1} \\ a_{n-1} & a_n & \dots & \dots & a_{2n-1} & a_{2n-2} \end{bmatrix}.$$

The coefficients of $P^{\underline{a}}(x) = x^d - \sum_{i=0}^{d-1} g_i x^i \in \mathbb{K}[x]$ are provided by the unique solution of the linear system

$$H_{0,d,d}^{\underline{a}} G = H_{d,d,1}^{\underline{a}},$$

that is,

$$\begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_{d-1} \\ a_1 & a_2 & & \ddots & a_d \\ a_2 & & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & & \vdots \\ a_{d-1} & a_d & \cdots & \cdots & a_{2d-2} \end{bmatrix} \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ \vdots \\ g_{d-1} \end{bmatrix} = \begin{bmatrix} a_d \\ a_{d+1} \\ a_{d+2} \\ \vdots \\ a_{2d-1} \end{bmatrix}. \quad (1)$$

As an immediate corollary of Proposition 2.1, we have the following result.

Corollary 2.2. *Using the notation of Proposition 2.1, a vector $Y = (p_0, \dots, p_n)$ is solution of*

$$H_{0,n,n+1}^{\underline{a}} Y = 0,$$

that is,

$$\begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_{n-1} & a_n \\ a_1 & a_2 & & \ddots & a_n & a_{n+1} \\ a_2 & & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & & \vdots & \vdots \\ a_{n-1} & a_n & \cdots & \cdots & a_{2n-2} & a_{2n-1} \end{bmatrix} \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ \vdots \\ p_{n-1} \\ p_n \end{bmatrix} = 0 \quad (2)$$

if and only if the polynomial $P(x) = \sum_{i=0}^n p_i x^i \in \mathbb{K}[x]$ is multiple of $P^{\underline{a}}(x)$.

Proof. By Proposition 2.1 the dimension of $\text{Ker}(H_{0,n,n+1}^{\underline{a}})$ is $n - d$. For $0 \leq j \leq n - 1$, let C_j denote the j th column of $H_{0,n,n+1}^{\underline{a}}$, that is $C_j = H_{j,n,1}^{\underline{a}} = [a_j, a_{j+1}, \dots, a_{n+j-1}]^t$. Since $P^{\underline{a}}(x)$ is a generating polynomial of \underline{a} , for $d \leq j \leq n - 1$, we obtain that

$$C_j - \sum_{i=j-d}^{j-1} g_{i-j+d} C_i = 0.$$

Thus the linear independent columns $[-g_0, \dots, -g_{d-1}, 1, 0, \dots, 0]^t, \dots, [0, \dots, 0, -g_0, \dots, -g_{d-1}, 1]^t$ define a basis of $\text{Ker}(H_{0,n,n+1}^{\underline{a}})$. Therefore, $Y = (p_0, \dots, p_n)$ verifies $H_{0,n,n+1}^{\underline{a}} Y = 0$ if and only if the polynomial $P(x) = \sum_{i=0}^n p_i x^i$ is a multiple of $P^{\underline{a}}(x)$. \square

If we consider $m = 2n - 1$ and $\widehat{S} = \sum_{i=0}^m a_{m-i} x^i$, by applying Equation (2) we obtain:

$$\exists R, U \in \mathbb{K}[x] \text{ such that } \deg(R) < n, \deg(P) \leq n \text{ and } P(x)S(x) + U(x)x^{2n} = R(x). \quad (3)$$

Hence, it turns out that finding the minimal polynomial of \underline{a} is equivalent to solving (3) for the minimum degree of P . Moreover, it's well known that

- the extended Euclidean Algorithm, with x^{2n} and \widehat{S} , provides an equality as (3) when the first remainder of degree smaller than $< n$ is reached. Let denote such remainder by R_k ,
- if we consider other polynomials $P'(x)$, $U'(x)$ and $R'(x)$ such that $P'(x)\widehat{S}(x) + U'(x)x^{2n} = R'(x)$ and $\deg(R') < \deg(R_{k-1})$, then $\deg(P') \geq \deg(P)$ and $\deg(U') \geq \deg(U)$.

That proves that our modification of Berlekamp-Massey Algorithm is right.

3 Lazy Evaluation

Our modified Berlekamp-Massey Algorithm admits a lazy evaluation, which may be very useful in solving the following problem.

Let $f(x) \in \mathbb{K}[x]$ be a squarefree polynomial of degree n . Let B be the universal decomposition algebra of $f(x)$, let A be a quotient algebra of B and $a \in A$. Thus, A is a zero-dimensional algebra given by

$$A \simeq \mathbb{K}[X_1, \dots, X_n]/\langle f_1, \dots, f_n \rangle,$$

where f_1, \dots, f_n define a Gröbner basis. Our aim is to compute the minimal polynomial of a , or at least, one of its factors. However, the dimension of A , denoted by m , over \mathbb{K} as vector space is normally too big to manipulate matrices of order m . Therefore, we apply the idea of Wiedemann's Algorithm, by computing the coefficients of a linearly recurrent sequence, $a_t = \phi(x^t)$, where ϕ is a linear form over A . Moreover, since the computation of x^t is usually very expensive and the minimal polynomial is likely to have degree smaller than the dimension, we are interested in computing the smallest possible number of coefficients in order to get the wanted polynomial.

Hence, we first choose $l < m$. We start Algorithm 2 with l and $[\phi(x^0), \dots, \phi(x^{2l-1})]$ as input, obtaining a polynomial as a result. Now, we test if such a polynomial is the minimal one. If this is not the case, we choose again another l' , $l < l' \leq m$, and we repeat the process with $2l'$ coefficients. However, in this next step, it is possible to take advantages of all the quotients computed before (with the exception of the last one), such that Euclidean Algorithm starts at $R_0 = U_0 x^{2l'} + V_0 \sum_{i=0}^{2l'-1} (\phi(x^{2l'-1-i}) x^i)$ and $R_1 = U_1 x^{2l'} + V_1 \sum_{i=0}^{2l'-1} (\phi(x^{2l'-1-i}) x^i)$, where U_0, V_0, U_1 and V_1 are Bezout coefficients computed in the previous step. Manifestly, repeating this argument again and again, we obtain the minimal polynomial.

The following pseudocode tries to facilitate the understanding of our lazy version of Berlekamp-Massey Algorithm.

Obviously, the choice of l is not unique. Here we have started at $l = m/4$, adding two coefficients in every further step. In practice, the particular characteristics of the given problem could help to choose a proper l and the method of increasing it through the algorithm. Of course, the simplification of the Euclidean Algorithm in Dornstetter (1987) must be considered to optimize the procedure.

Algorithm 3. The lazy Berlekamp-Massey Algorithm (in some particular context)

Input: $m \in \mathbb{N}$, $C \in \mathbb{K}^n$, G : Gröbner basis, $a \in A$. The minimal polynomial has degree bound m .

Output : The minimal polynomial P of a

Start

Local variables : l, i : integers, $R, R_{-1}, R_0, R_1, V, V_{-1}, V_0, V_1, U, U_{-1}, U_0, U_1, S_0, S_1, Q$: polynomials in x , L, W :lists, val : element of A ;

```

# initialization
 $l = \lfloor m/4 \rfloor$ ;
 $L := [1, a]$ ;  $W := [1, \text{Value}(a, C)]$ ;
 $S_0 := x^{2l}$  ;  $S_1 = W[1]x^{2l-1} + W[2]x^{2l-2}$ ;
# loop
for  $i$  from 3 to  $2l$  do
     $L[i] := \text{normalf}(L[i-1]a, G)$ ;
     $V[i] := \text{Value}(L[i], C)$ ;
     $S_1 = S_1 + V[i]x^{2l-i}$ ;
end for
 $R_0 := S_0$ ;  $R_1 := S_1$ ;  $V_0 = 0$  ;  $V_1 = 1$  ;  $U_0 = 1$  ;  $V_1 = 0$ ;
# loop
while  $l \leq \deg(R_1)$  do
    ( $Q, R$ ) := quotient and remainder of  $R_0$  divided by  $R_1$  ;
     $V := V_0 - QV_1$ ;  $U := U_0 - QU_1$  ;  $U_{-1} := U_0$ ;  $V_{-1} := V_0$ ;
     $V_0 := V_1$  ;  $V_1 := V$  ;  $U_0 := U_1$  ;  $U_1 := U$ ;  $R_0 := R_1$  ;  $R_1 := R$  ;
end while
 $val := \text{Subs}(x = a, V_1)$ ;
# loop
while  $val \neq 0$  do
     $l := l + 1$ ;
    # loop
    for  $i$  from  $2l - 1$  to  $2l$  do
         $L[i] := \text{normalf}(L[i-1]a, G)$ ;
         $W[i] := \text{Value}(L[i], C)$ ;
    end for
     $S_0 = x^2S_0$ ;  $S_1 = x^2S_1 + W[2l-1]x + W[2l]$ ;
     $R_0 := U_{-1}S_0 + V_{-1}S_1$ ;  $R_1 := U_0S_0 + V_0S_1$  ;
     $U_1 := U_0$ ;  $V_1 := V_0$ ;  $U_0 := U_{-1}$ ;  $V_0 := V_{-1}$ ;
    # loop
    while  $l \leq \deg(R_1)$  do
        ( $Q, R$ ) := quotient and remainder of  $R_0$  divided by  $R_1$  ;
         $V := V_0 - QV_1$ ;  $U := U_0 - QU_1$ ;  $U_{-1} := U_0$ ;  $V_{-1} := V_0$ ;
         $V_0 := V_1$  ;  $V_1 := V$  ;  $U_0 := U_1$  ;  $U_1 := U$ ;  $R_0 := R_1$  ;  $R_1 := R$  ;
    end while
     $val := \text{Subs}(x = a, V_1)$ 
end while
# exit
Return  $P := V_1/\text{leadcoeff}(P)$ .
End.

```

References

- BERLEKAMP, E. R. (1984). *Algebraic coding theory*. Second edition of the 1968 book. Laguna Hills, CA: Aegean Park Press. [E1](#)
- CHENG, U. (1984). On the continued fraction and Berlekamp's algorithm. *IEEE Trans. Inf. Theory*, 30:541–544. [E1](#), [F2](#)
- DORNSTETTER, J. L. (1987). On the equivalence between Berlekamp's and Euclid's algorithms. *IEEE Trans. Inf. Theory*, 33:428–431. [E1](#), [E2](#), [E6](#), [F2](#)
- JONCKHEERE, E. and MA, C. (1989). A simple Hankel interpretation of the Berlekamp-Massey algorithm. *Linear Algebra and its Applications*, 125:65–76. [E1](#), [F2](#)
- MASSEY, J. L. (1969). Shift-register synthesis and BCH decoding. *IEEE Trans. Inf. Theory*, 15:122–127. [E1](#), [F1](#)
- MILLS, W. H. (1975). Continued fractions and linear recurrences. *Mathematics of Computation*, 29(129): 173–180. [E1](#), [F2](#)
- PAN, V. Y. (2000). New techniques for the computation of linear recurrence coefficients. *Finite Fields Appl.*, 6(1):93–118. [E1](#), [F2](#)
- SHOUP, V. (2005). *A computational introduction to number theory and algebra*. Cambridge: Cambridge University Press. [E2](#)
- SUGIYAMA, Y., KASAHARA, M., HIRASAWA, S. et NAMEKAWA, T. (1975). A method for solving key equation for decoding Goppa codes. *Inf. Control*, 27:87–99. [E1](#), [F2](#)
- WELCH, L. R. and SCHOLTZ, R. A. (1979). Continued fractions and Berlekamp's algorithm. *IEEE Trans. Inf. Theory*, 25:19–27. [E1](#), [F2](#)

Une variante de l'algorithme de Berlekamp-Massey

Nadia Ben Atti (*), Gema M. Diaz-Toca (†) Henri Lombardi (‡)

2005, traduction 2022

Résumé

Nous donnons une variante, légèrement plus simple, de l'algorithme de Berlekamp-Massey. L'explication de l'algorithme est également plus facile. En outre cela autorise une approche «dynamique» de cet algorithme utile dans certaines situations

MSC 2000 : 68W30, 15A03

Mots clés : Algorithme de Berlekamp-Massey. Suites récurrentes linéaires.

Note : Cet article est la version française de l'article *The Berlekamp-Massey Algorithm revisited* paru dans AAECC **17** 1 (2006), 75–82. Received : 14 September 2005 / Published online : 9 March 2006

1	Introduction : l'algorithme de Berlekamp-Massey usuel	F1
2	De bonnes raisons pour modifier l'algorithme usuel	F2
3	Une variante paresseuse	F5

1 Introduction : l'algorithme de Berlekamp-Massey usuel

Soit \mathbb{K} un corps arbitraire. Étant donné une suite récurrente linéaire notée $\mathcal{S}(x) = \sum_{i=0}^{\infty} a_i x^i$, $a_i \in \mathbb{K}$, on désire calculer son polynôme minimal, noté $P(x)$. Rappelons que si $P(x) = \sum_{i=0}^d p_i x^i$, alors $P(x)$ est le polynôme de degré le plus petit possible satisfaisant $\sum_{i=0}^d p_i a_{j+i} = 0$, pour tous $j \in \mathbb{N}$.

Supposons que ce polynôme minimal a son degré majoré par n . Alors l'algorithme de Berlekamp-Massey utilise seulement les $2n$ premiers coefficients de $\mathcal{S}(x)$ par calculer ce polynôme minimal. Ces $2n$ coefficients définissent le polynôme $S = \sum_{i=0}^{2n-1} a_i x^i$.

Une vaste littérature est parue concernant cet algorithme. L'algorithme original a été créé en 1968 par Berlekamp (voir Berlekamp (1984)) pour décoder les codes de Bose-Chaudhuri-Hocquenghem (BCH). Une année plus tard, l'algorithme a été simplifié dans Massey (1969). La

* Équipe de Mathématiques, CNRS UMR 6623, UFR des Sciences et Techniques, Université de Franche-Comté, 25 030 BESANCON cedex, FRANCE. nadia.benatti@ensi.rnu.tn

† Dpto. de Matematicas Aplicada. Universidad de Murcia, Spain. gemadiaz@um.es

‡ Équipe de Mathématiques, CNRS UMR 6623, UFR des Sciences et Techniques, Université de Franche-Comté, 25 030 BESANCON cedex, FRANCE, henri.lombardi@univ-fcomte.fr, partiellement financé par le réseau européen RAAG CT-2001-00271

ressemblance entre cet algorithme et l'algorithme d'Euclide étendu a été soulignée dans plusieurs articles, par exemple [Cheng \(1984\)](#), [Dornstetter \(1987\)](#), [Mills \(1975\)](#), [Sugiyama et al. \(1975\)](#) and [Welch and Scholtz \(1979\)](#). Des interprétations plus récentes en termes de matrices de Hankel et d'approximants de Padé se trouvent dans [Jonckheere and Ma \(1989\)](#) et [Pan \(2000\)](#).

L'interprétation usuelle de l'algorithme de Berlekamp-Massey est exprimée dans le pseudo code de l'Algorithme 1.

Algorithme 1. Algorithme de Berlekamp-Massey usuel

Entrée : Un entier $n \geq 1$. Une liste non nulle d'éléments du corps \mathbb{K} , $[a_0, a_1, \dots, a_{2n-1}]$: les $2n$ premiers termes d'une suite récurrente linéaire, sous l'hypothèse qu'elle admet un polynôme générateur de degré $\leq n$.

Sortie : Le polynôme générateur minimal P de la suite récurrente linéaire.

Début

Variables locales : $R, R_0, R_1, V, V_0, V_1, Q$: polynômes en X

initialisation

$R_0 := X^{2n} ; R_1 := \sum_{i=0}^{2n-1} a_i X^i ; V_0 = 0 ; V_1 = 1 ;$

boucle

tant que $n \leq \deg(R_1)$ **faire**

$(Q, R) :=$ quotient et reste de la division de R_0 par R_1 ;

$V := V_0 - Q * V_1$;

$V_0 := V_1 ; V_1 := V ; R_0 := R_1 ; R_1 := R$;

fin tant que

sortie

$d := \sup(\deg(V_1), 1 + \deg(R_1)) ; P := X^d V_1(1/X) ;$ Retourner $P = P/\text{lc}(P)$.

Fin.

Bien que très simple cet algorithme a toujours semblé un peu trop difficile à justifier. Dans la littérature, il semble y avoir une petite confusion concernant la définition du polynôme minimal d'une suite récurrente linéaire. Ici, nous introduisons dans la section 2 une légère modification de l'algorithme qui le rend plus naturel et plus facile à justifier. Nous n'avons pas trouvé trace de cette variante dans la littérature parue avant notre première soumission à Mathematics of Computation en 2004. Cette variante est apparue dans le livre de Victor Shoup ([Shoup \(2005\)](#)), sans aucune référence à un article antérieur.

2 De bonnes raisons pour modifier l'algorithme usuel

La variante que nous introduisons est basée sur l'idée suivante. Puisqu'à la fin de l'algorithme, le polynôme V doit être renversé selon une procédure difficile à expliquer (pourquoi doit-on prendre le polynôme réciproque en degré $d = \sup(\deg(V_1), 1 + \deg(R_1))$?), le mieux ne serait-il pas de traiter directement la suite récurrente linéaire «dans le bon sens» (elle a elle-même été renversée au départ lorsqu'on a affecté $R_1 := \sum_{i=0}^{2n-1} a_i X^i$) ?

Naturellement l'appréciation selon laquelle la suite récurrente linéaire a été renversée au départ peut sembler subjective. Elle est en fait renforcée par la remarque suivante. Si le polynôme générateur minimal est de degré d nettement plus petit que n les calculs dans l'algorithme ne devraient pas être sensiblement différents lorsqu'on travaille avec les $2d$ premiers termes de la suite ou lorsqu'on travaille avec les $2n$ premiers termes. Or le renversement effectué au début de

l'algorithme change complètement le calcul qui est fait. Tandis qu'en l'absence de renversement, avec notre variante, le calcul sur la suite courte peut facilement être regardé comme le calcul sur la suite longue, tronqué de manière convenable.

Une autre confirmation est le caractère plus simple et plus facile à justifier dans l'affectation finale.

Algorithme 2. Algorithme de Berlekamp-Massey, variante

Entrée : Un entier $n \geq 1$. Une liste non nulle d'éléments du corps \mathbb{K} , $[a_0, a_1, \dots, a_{2n-1}]$: les $2n$ premiers termes d'une suite récurrente linéaire, sous l'hypothèse qu'elle admet un polynôme générateur de degré $\leq n$.

Sortie : Le polynôme générateur minimal P de la suite récurrente linéaire.

Début

Variables locales : $R, R_0, R_1, V, V_0, V_1, Q$: polynômes en X ; $m = 2n - 1$: entier.

initialisation

$m := 2n - 1$; $R_0 := X^{2n}$; $R_1 := \sum_{i=0}^m a_{m-i}X^i$; $V_0 = 0$; $V_1 = 1$;

boucle

tant que $n \leq \deg(R_1)$ **faire**

$(Q, R) :=$ quotient et reste de la division de R_0 par R_1 ;

$V := V_0 - Q * V_1$;

$V_0 := V_1$; $V_1 := V$; $R_0 := R_1$; $R_1 := R$;

fin tant que

sortie

Retourner $P := V_1/\text{lc}(V_1)$.

Fin.

Nous allons maintenant prouver la correction de cet algorithme.

Si $\underline{a} = (a_n)_{n \in \mathbb{N}}$ est une suite arbitraire et si $i, r, p \in \mathbb{N}$ nous noterons $H_{i,r,p}^{\underline{a}}$ la matrice de Hankel suivante, qui possède r lignes et p colonnes :

$$H_{i,r,p}^{\underline{a}} = \begin{bmatrix} a_i & a_{i+1} & a_{i+2} & \cdots & a_{i+p-1} \\ a_{i+1} & a_{i+2} & & & a_{i+p} \\ a_{i+2} & & & & \\ \vdots & & & & \vdots \\ a_{i+r-1} & a_{i+r} & \cdots & \cdots & a_{i+r+p-2} \end{bmatrix}.$$

et nous noterons $P^{\underline{a}}(X)$ le polynôme générateur minimal de \underline{a} .

La proposition suivante est classique (voir par exemple [Jonckheere et Ma \(1989\)](#)).

Proposition 2.1. Si \underline{a} est une suite récurrente linéaire qui admet un polynôme générateur de degré $\leq n$, alors le degré $d \leq n$ de son polynôme générateur minimal $P^{\underline{a}}$ est égal au rang de la matrice de Hankel

$$H_{0,n,n}^{\underline{a}} = \begin{bmatrix} a_0 & a_1 & a_2 & \cdots & a_{n-1} \\ a_1 & a_2 & & \ddots & a_n \\ a_2 & & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & & \vdots \\ a_{n-1} & a_n & \cdots & \cdots & a_{2n-2} \end{bmatrix}.$$

Les coefficients de $P^{\underline{a}}(X) = X^d - \sum_{i=0}^{d-1} g_i X^i \in \mathbb{K}[X]$ sont l'unique solution de l'équation

$$\text{H}_{0,d,d}^{\underline{a}} G = \text{H}_{1,d,d}^{\underline{a}}$$

c'est-à-dire encore l'unique solution du système linéaire

$$\left[\begin{array}{cccccc} a_0 & a_1 & a_2 & \cdots & a_{d-1} \\ a_1 & a_2 & & \ddots & a_d \\ a_2 & & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & & \vdots \\ a_{d-1} & a_d & \cdots & \cdots & a_{2d-2} \end{array} \right] \left[\begin{array}{c} g_0 \\ g_1 \\ g_2 \\ \vdots \\ g_{d-1} \end{array} \right] = \left[\begin{array}{c} a_d \\ a_{d+1} \\ a_{d+2} \\ \vdots \\ a_{2d-1} \end{array} \right]. \quad (1)$$

Corolaire 2.2. On reprend les notations de la proposition 2.1. Pour qu'un vecteur $Y = (p_0, \dots, p_n)$ soit solution de l'équation

$$\text{H}_{0,n,n+1}^{\underline{a}} Y = 0$$

c'est-à-dire

$$\left[\begin{array}{cccccc} a_0 & a_1 & a_2 & \cdots & a_{n-1} & a_n \\ a_1 & a_2 & & \ddots & a_n & a_{n+1} \\ a_2 & & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & & \vdots & \vdots \\ a_{n-1} & a_n & \cdots & \cdots & a_{2n-2} & a_{2n-1} \end{array} \right] \left[\begin{array}{c} p_0 \\ p_1 \\ p_2 \\ \vdots \\ p_{n-1} \\ p_n \end{array} \right] = 0 \quad (2)$$

il faut et il suffit que le polynôme $P(X) = \sum_{i=0}^n p_i X^i \in \mathbb{K}[X]$ soit multiple de $P^{\underline{a}}(X)$.

Par ailleurs, nous faisons la constatation suivante.

Fait 2.3. En posant $m := 2n - 1$ et $S := \sum_{i=0}^m a_{m-i} X^i$, l'équation (2) est équivalente à l'affirmation suivante. Le polynôme P est de degré $\leq n$ et on a :

$$\exists R \in \mathbb{K}[X] \text{ tel que } \deg(R) < n \quad \text{et} \quad P(X)S(X) \equiv R(X) \pmod{X^{2n}}.$$

c'est-à-dire encore

$$\exists R, U \in \mathbb{K}[X] \text{ tels que } \deg(R) < n \quad \text{et} \quad P(X)S(X) + U(X)X^{2n} = R(X) \quad (3)$$

Le problème de trouver le polynôme générateur minimal de \underline{a} est donc ramené au problème de réaliser (3) avec le degré de P minimum. Or il est bien connu que :

- l'algorithme d'Euclide étendu démarrant avec $R_0 = X^{2n}$ et $R_1 = S$ réalise une équation telle que (3) avec $R = R_k$ lorsque le premier reste R_k de degré $< n$ est atteint,
- quand une équation telle que (3) est réalisée par l'algorithme d'Euclide étendu, il n'est pas possible d'obtenir une équation du même type $P'(X)S(X) + U'(X)X^{2n} = R'(X)$ avec $\deg(R') < \deg(R_{k-1})$ et $\deg(P') < \deg(P)$.

Ceci prouve que notre algorithme est correct.

3 Une variante paresseuse

Outre sa simplicité, notre variation sur l'algorithme de Berlekamp-Massey admet une variante paresseuse, qui peut être utile dans certains cas.

Par exemple considérons l'algèbre B de décomposition universelle d'un polynôme $f(X)$ (séparable) sur un corps \mathbb{K} , ou plus généralement dans un quotient A de B . A est une algèbre zéro-dimensionnelle dont on connaît une présentation : $A \simeq \mathbb{K}[X_1, \dots, X_n]/\langle f_1, \dots, f_s \rangle$, où f_1, \dots, f_s est une base de Gröbner. La dimension m de A sur \mathbb{K} est trop grande pour qu'on envisage de manipuler des matrices dans $\mathbb{K}^{m \times m}$. On est intéressé pour calculer le polynôme minimal d'un élément x de A , ou à défaut un facteur de ce polynôme minimal. On utilise pour cela l'algorithme de Wiedemann. On calcule donc les termes successifs d'une suite récurrente linéaire : $a_n = \phi(x^n)$ où ϕ est une forme linéaire sur A . Le calcul de x^n étant assez coûteux (on doit à chaque fois calculer la nouvelle forme normale), et le polynôme minimal étant a priori de degré nettement plus petit que la dimension de A (vu le choix judicieux de x), on a intérêt à tester régulièrement si le polynôme minimal de x n'est pas atteint "avant terme". . . .

Algorithme 3. *L'algorithme de Berlekamp-Massey paresseux (dans un contexte particulier)*

Entrée : $m \in \mathbb{N}$, $C \in \mathbb{K}^n$, G : base de Gröbner, $a \in A$. Le polynôme minimal est de degré $\leq m$.

Sortie : Le polynôme minimal P de a

Début

Variables locales : l, i : entiers, $R, R_{-1}, R_0, R_1, V, V_{-1}, V_0, V_1, U, U_{-1}, U_0, U_1, S_0, S_1, Q$: polynômes en x , L, W : listes dans A , val : élément de A ;

```

# initialisation
 $l = \lfloor m/4 \rfloor$ ;
 $L := [1, a]$ ;  $W := [1, \text{Value}(a, C)]$ ;
 $S_0 := x^{2l}$ ;  $S_1 = W[1]x^{2l-1} + W[2]x^{2l-2}$ ;
# boucle
pour  $i$  de 3 à  $2l$ 
     $L[i] := \text{normalf}(L[i-1]a, G)$ ;  $V[i] := \text{Value}(L[i], C)$ ;  $S_1 = S_1 + V[i]x^{2l-i}$ ;
fin pour
 $R_0 := S_0$ ;  $R_1 := S_1$ ;  $V_0 = 0$ ;  $V_1 = 1$ ;  $U_0 = 1$ ;  $V_1 = 0$ ;
# boucle
tant que  $l \leq \deg(R_1)$  faire
     $(Q, R) := \text{quotient et reste de } R_0 \text{ divisé par } R_1$ ;
     $V := V_0 - QV_1$ ;  $U := U_0 - QU_1$ ;  $U_{-1} := U_0$ ;  $V_{-1} := V_0$ ;
     $V_0 := V_1$ ;  $V_1 := V$ ;  $U_0 := U_1$ ;  $U_1 := U$ ;  $R_0 := R_1$ ;  $R_1 := R$ ;
fin tant que
 $val := \text{Subs}(x = a, V_1)$ ;
# boucle
tant que  $val \neq 0$  faire
     $l := l + 1$ ;
# boucle
pour  $i$  de  $2l-1$  à  $2l$ 
     $L[i] := \text{normalf}(L[i-1]a, G)$ ;
     $W[i] := \text{Value}(L[i], C)$ ;
fin pour
 $S_0 = x^2S_0$ ;  $S_1 = x^2S_1 + W[2l-1]x + W[2l]$ ;
 $R_0 := U_{-1}S_0 + V_{-1}S_1$ ;  $R_1 := U_0S_0 + V_0S_1$ ;
 $U_1 := U_0$ ;  $V_1 := V_0$ ;  $U_0 := U_{-1}$ ;  $V_0 := V_{-1}$ ;
# boucle
tant que  $l \leq \deg(R_1)$  faire
     $(Q, R) := \text{quotient et reste de } R_0 \text{ divisé par } R_1$ ;
     $V := V_0 - QV_1$ ;  $U := U_0 - QU_1$ ;  $U_{-1} := U_0$ ;  $V_{-1} := V_0$ ;
     $V_0 := V_1$ ;  $V_1 := V$ ;  $U_0 := U_1$ ;  $U_1 := U$ ;  $R_0 := R_1$ ;  $R_1 := R$ ;
fin tant que
 $val := \text{Subs}(x = a, V_1)$ 
fin tant que
# sortie
Retourner  $P := V_1/\text{leadcoeff}(P)$ .
```

Fin.

Références

- BERLEKAMP, E. R. (1984). *Algebraic coding theory* Seconde édition du livre original de 1968. Laguna Hills, CA : Aegean Park Press. **F1**
- CHENG, U. (1984). On the continued fraction and Berlekamp's algorithm. *IEEE Trans. Inf. Theory*, 30:541–544.
- DORNSTETTER, J. L. (1987). On the equivalence between Berlekamp's and Euclid's algorithms. *IEEE Trans. Inf. Theory*, 33:428–431.
- JONCKHEERE, E. et MA, C. (1989). A simple Hankel interpretation of the Berlekamp-Massey algorithm. *Linear Algebra and its Applications*, 125:65–76. **F3**
- MASSEY, J. L. (1969). Shift-register synthesis and bch decoding. *IEEE Trans. Inf. Theory*, 15:122–127.
- MILLS, W. H. (1975). Continued Fractions and Linear Recurrences. *Mathematics of Computation*, 29(129):173–180.
- PAN, V. Y. (2000). New techniques for the computation of linear recurrence coefficients. *Finite Fields Appl.*, 6(1):93–118.
- SHOUP, V. (2005). *A computational introduction to number theory and algebra*. Cambridge : Cambridge University Press. **F2**
- SUGIYAMA, Y., KASAHARA, M., HIRASAWA, S. et NAMEKAWA, T. (1975). A method for solving key equation for decoding Goppa codes. *Inf. Control*, 27:87–99.
- WELCH, L. R. et SCHOLTZ, R. A. (1979). Continued fractions and Berlekamp's algorithm. *IEEE Trans. Inf. Theory*, 25:19–27.