Designs, Codes and Cryptography manuscript No.
(will be inserted by the editor)

Probabilistic Algorithm for Finding Roots of Linearized Polynomials

Vitaly Skachek · Ron M. Roth

Abstract A probabilistic algorithm is presented for finding a basis of the root space of a linearized polynomial

$$L(x) = \sum_{i=0}^{t} L_i x^{q^i}$$

over $\mathbb{F}_{q^n}$. The expected time complexity of the presented algorithm is $O(nt)$ operations in $\mathbb{F}_{q^n}$.

Keywords Linearized polynomials · Probabilistic algorithms · Root-finding algorithms · Symbolic GCD

1 Introduction

Let q be a power of a prime and n be a positive integer. A *linearized polynomial over* $\mathbb{F}_{q^n}$ (with respect to $\mathbb{F}_q$) is a polynomial of the form

$$L(x) \ = \ \sum_{i=0}^{t} L_i x^{q^i} \ ,$$

where $L_i \in \mathbb{F}_{q^n}$.

Linearized polynomials were first investigated by Ore (see [12], [13]). References [9, §3.4] and [11, §4.9] contain rather extensive summaries of the properties of linearized polynomials. In particular, it is known that the mapping $\mathbb{F}_{q^n} \rightarrow \mathbb{F}_{q^n}$ defined by $x \mapsto L(x)$ is a linear mapping over $\mathbb{F}_q$. Conversely, every

Vitaly Skachek
Claude Shannon Institute, University College Dublin, Belfield, Dublin 4, Ireland.
E-mail: vitaly.skachek@ucd.ie

Ron M. Roth
Computer Science Department, Technion, Haifa 32000, Israel.
E-mail: ronny@cs.technion.ac.il

linear mapping $\mathbb{F}_{q^n} \to \mathbb{F}_{q^n}$ over $\mathbb{F}_q$ can be realized as a linearized polynomial of degree less than q^n. These properties of linearized polynomials imply that the roots of a given linearized polynomial over $\mathbb{F}_{q^n}$ in any extension field of $\mathbb{F}_q$ form a vector space over $\mathbb{F}_q$. For applications of linearized polynomials to coding theory, see [2], [5], [6], [15], or [16].

One problem that arises in some of those applications is finding a basis over $\mathbb{F}_q$ of the root space in $\mathbb{F}_{q^n}$ of a given linearized polynomial $L(x)$ of degree $q^t < q^n$ over $\mathbb{F}_{q^n}$. To this end, we can first find a representation of the linear mapping $L : \mathbb{F}_{q^n} \to \mathbb{F}_{q^n}$ as an $n \times n$ matrix A over $\mathbb{F}_q$, according to some basis of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$, and then compute a basis of the kernel of A in $\mathbb{F}_q^n$. The fastest algorithms currently known for finding the kernel of an $n \times n$ matrix over $\mathbb{F}_q$ have time complexity which grows at least as $n^{2+\varepsilon}$, where $\varepsilon \approx 0.376$ (see [1] and [4]; this lower bound on the complexity does not take into account the time required to compute the matrix A from $L(x)$).

Alternatively, a basis of the root space of $L(x)$ can be found by an adaptation of Rabin's probabilistic algorithm for root finding of general polynomials over fields of even characteristic [14], taking into account the special structure and properties of linearized polynomials (see also the improved analysis of Ben-Or [3]). It can be shown that such an adapted version of Rabin's algorithm can find *one* nonzero root of a linearized polynomial in expected time complexity of $O(nt^2)$ operations in $\mathbb{F}_{q^n}$.

In this note, we present a fast algorithm for finding a *whole basis* of the root space (in $\mathbb{F}_{q^n}$) of a linearized polynomial over $\mathbb{F}_{q^n}$, in expected time complexity of $O(nt)$ operations in $\mathbb{F}_{q^n}$. Hereafter, by "operations in $\mathbb{F}_{q^n}$" we mean any of the four arithmetic operations—addition, subtraction, multiplication, or division—in $\mathbb{F}_{q^n}$, as well as raising an element to the qth power (referred to here as q-exponentiation). When we represent $\mathbb{F}_{q^n}$ as a ring of polynomials modulo an irreducible polynomial of degree n over $\mathbb{F}_q$, each of the four arithmetic operations in $\mathbb{F}_{q^n}$ can be implemented using $O(n \log^2 n \log\log n)$ arithmetic operations in $\mathbb{F}_q$ (see [1, §8.3] and [8, §§8.3, 9.1, and 11.1]), and q-exponentiation can be implemented by $O(\log q)$ multiplications in $\mathbb{F}_{q^n}$. (Moreover, for a range of values of q and n, the representation of $\mathbb{F}_{q^n}$ according to certain normal bases over $\mathbb{F}_q$ allows us to implement all operations in $\mathbb{F}_{q^n}$—including q-exponentiation—using $O(n \log^2 n \log\log n)$ arithmetic operations in $\mathbb{F}_q$ [7].) Hence, our algorithm has time complexity of $O(n^2 t \log^2 n \log\log n \log q)$ operations in $\mathbb{F}_q$, making it faster than the aforementioned algorithms whenever $t = o(n^\varepsilon / (\log^2 n \log\log n \log q))$.

2 Symbolic GCD

In this section, we summarize several definitions and properties that will be used in the sequel. Most properties can be found in Ore [12, Ch. 1].

Let $L(x)$ and $M(x)$ be linearized polynomials over $\mathbb{F}_{q^n}$. The *symbolic product* of $L(x)$ with $M(x)$ is defined by

$$L(x) \otimes M(x) = L(M(x)) .$$

Symbolic product satisfies associativity and distributivity (with respect to ordinary polynomial addition), but in general it does not satisfy commutativity; i.e. $L(x) \otimes M(x)$ and $M(x) \otimes L(x)$ are typically not equal.

Let $L(x)$ and $M(x)$ be linearized polynomials over $\mathbb{F}_{q^n}$ where $M(x) \neq 0$. Using an algorithm akin to ordinary "long division," one can find unique linearized polynomials $Q(x)$ and $R(x)$ over $\mathbb{F}_{q^n}$ such that

$$L(x) = Q(x) \otimes M(x) + R(x) \quad \text{and} \quad \deg R(x) < \deg M(x) \ . \tag{1}$$

When $R(x) = 0$, we say that $M(x)$ is a *right symbolic divisor* of $L(x)$. The polynomial $M(x)$ is a right symbolic divisor of $L(x)$, if and only if $M(x)$ divides $L(x)$ in the ordinary sense (see [12, p. 561] for the "only if" part; the "if" part is easy to prove).

Let $L(x)$ and $M(x)$ be linearized polynomials over $\mathbb{F}_{q^n}$, not both zero. A *right symbolic greatest common divisor* of $L(x)$ and $M(x)$ is a monic linearized polynomial $G(x)$ over $\mathbb{F}_{q^n}$ of a largest degree such that $G(x)$ is a right symbolic divisor of both $L(x)$ and $M(x)$.

Proposition 1 [12, Theorem 1] *Let $L(x)$ and $M(x)$ be linearized polynomials over $\mathbb{F}_{q^n}$, not both zero. The right symbolic greatest common divisor of $L(x)$ and $M(x)$ is unique and equals the return value of the algorithm in Figure 1.*

The unique right symbolic greatest common divisor of $L(x)$ and $M(x)$ will be denoted by $\mathrm{rgcd}(L(x), M(x))$.

Proposition 2 [12, Theorem 2] *Let $L(x)$ and $M(x)$ be linearized polynomials over $\mathbb{F}_{q^n}$, not both zero. Then*

$$\mathrm{rgcd}(L(x), M(x)) = \gcd(L(x), M(x)) \ ,$$

where $\gcd(L(x), M(x))$ is the monic (ordinary) greatest common divisor of $L(x)$ and $M(x)$.

Similarly to (1), for any two linearized polynomials $L(x)$ and $M(x) \neq 0$ over $\mathbb{F}_{q^n}$ there exist unique linearized polynomials $Q(x)$ and $R(x)$ over $\mathbb{F}_{q^n}$ such that

$$L(x) = M(x) \otimes Q(x) + R(x) \quad \text{and} \quad \deg R(x) < \deg M(x) \ .$$

When $R(x) = 0$, we say that $M(x)$ is a *left symbolic divisor* of $L(x)$. In general, the set of left symbolic divisors of a given linearized polynomial may differ from its set of right symbolic divisors. However, we do have the following result.

Input: linearized polynomials $L(x) \neq 0$ and $M(x)$ over $\mathbb{F}_{q^n}$;
$R_{-1}(x) \leftarrow M(x)$; $R_0(x) \leftarrow L(x)$;
for ($i \leftarrow 1$; $R_{i-1}(x) \neq 0$; i++)
  $R_i(x) \leftarrow R_{i-2} - Q_i(x) \otimes R_{i-1}(x)$, where $\deg R_i(x) < \deg R_{i-1}(x)$;
normalize $R_{i-2}(x)$ to be monic;
Output: $R_{i-2}(x)$.

Fig. 1 Algorithm for computing $\mathrm{rgcd}(L(x), M(x))$.

Lemma 3 *A linearized polynomial $M(x)$ over $\mathbb{F}_{q^n}$ is a right symbolic divisor (or an ordinary divisor) of the polynomial $x^{q^n} - x$, if and only if $M(x)$ is a left symbolic divisor of that polynomial.*

Proof Starting with the "only if" part, suppose that

$$x^{q^n} - x = P(x) \otimes M(x)$$

for some linearized polynomial $P(x)$ over $\mathbb{F}_{q^n}$. Next write

$$x^{q^n} - x = M(x) \otimes Q(x) + R(x) \tag{2}$$

for two linearized polynomials $Q(x)$ and $R(x)$ such that $\deg R(x) < \deg M(x)$. Computing the symbolic product of $P(x)$ with both sides of (2) we obtain

$$\begin{aligned} P(x) \otimes (x^{q^n} - x) &= P(x) \otimes M(x) \otimes Q(x) + P(x) \otimes R(x) \\ &= (x^{q^n} - x) \otimes Q(x) + P(x) \otimes R(x) \; . \end{aligned}$$

Now, the two polynomials $P(x) \otimes (x^{q^n} - x)$ and $(x^{q^n} - x) \otimes Q(x)$ vanish at each element of $\mathbb{F}_{q^n}$; therefore, so must $P(x) \otimes R(x)$. On the other hand, since $\deg R(x) < \deg M(x)$, we have

$$\deg(P(x) \otimes R(x)) < \deg(P(x) \otimes M(x)) = \deg(x^{q^n} - x) = q^n \; .$$

Hence, $P(x) \otimes R(x) = 0$ and, so, $R(x) = 0$.

The proof of the "if" part is similar. □

3 Finding roots of linearized polynomials

Figure 2 presents a probabilistic algorithm for finding a basis over $\mathbb{F}_q$ of the roots in $\mathbb{F}_{q^n}$ of a given linearized polynomial $L(x)$ over $\mathbb{F}_{q^n}$. Assuming that $L(x) \neq 0$, we let t denote $\log_q \deg L(x)$.

By Proposition 2 we have that the computed linearized polynomial $G(x)$ in Figure 2 equals $\gcd(L(x), x^{q^n} - x)$. So, the roots of $G(x)$ in $\mathbb{F}_{q^n}$ are the

---

<u>**Input:**</u> linearized polynomial $L(x)$ over $\mathbb{F}_{q^n}$;
$G(x) \leftarrow \text{rgcd}(L(x), x^{q^n} - x)$; /* use the algorithm in Figure 1 */
denote $r = \log_q \deg G(x)$;
compute a linearized polynomial $H(x)$ such that $x^{q^n} - x = G(x) \otimes H(x)$;
for ($j \leftarrow 0$; $j < r$; j++) {
  do {
    select at random an element $z_j \in \mathbb{F}_{q^n}$;
  }
  while $H(z_j)$ is in the linear span of $\{H(z_\ell)\}_{\ell=0}^{j-1}$ over $\mathbb{F}_q$;
}
<u>**Output:**</u> basis elements $H(z_0), H(z_1), \ldots, H(z_{r-1})$.

Fig. 2 Algorithm for finding a basis of the root space of $L(x)$ in $\mathbb{F}_{q^n}$.

roots of $L(x)$ in that field, and the dimension of the linear space of these roots is $r = \log_q \deg G(x)$. Lemma 3 implies that $G(x)$ is also a left symbolic divisor of $x^{q^n} - x$ and, thus, the polynomial $H(x)$ in Figure 2 is well defined.

Given that the algorithm in Figure 2 halts, it is rather straightforward to see that it returns a basis of the root space of $G(x)$ and, hence, of $L(x)$. The rest of this section is devoted to analyzing the time complexity of the algorithm.

Lemma 4 *The polynomial $G(x)$ in Figure 2 can be computed using less than $3(n+1)(t+1)$ operations in $\mathbb{F}_{q^n}$.*

Proof When $t \le n$ (the typical case), we apply the algorithm in Figure 1 to $R_{-1}(x) = x^{q^n} - x$ and $R_0(x) = L(x)$. Otherwise, we switch the roles of $R_{-1}(x)$ and $R_0(x)$.

Denote by ν the largest value of i in Figure 1 for which $R_i(x) \neq 0$ and, for $i = -1, 0, 1, \ldots, \nu$, let τ_i stand for $(\log_q \deg R_i) + 1$. Using symbolic long division to implement each iteration in the main loop in Figure 1, iteration i requires less than

$$3(\tau_{i-2} - \tau_{i-1} + 1)\tau_{i-1}$$

operations in $\mathbb{F}_{q^n}$. Hence, the overall number of operations in $\mathbb{F}_{q^n}$ that are required to compute $G(x)$ (without applying the last normalization step in Figure 1) is less than three times the value of

$$\begin{aligned}
&\sum_{i=1}^{\nu+1} (\tau_{i-2} - \tau_{i-1} + 1)\tau_{i-1} \\
&\qquad \le (\tau_{-1} - \tau_0 + 1)\tau_0 + \sum_{i=2}^{\nu+1} \Big(\tau_{i-2}(\tau_{i-2} - 1) - \tau_{i-1}(\tau_{i-1} - 1) \Big) \\
&\qquad \le \tau_{-1}\tau_0 \\
&\qquad = (n+1)(t+1) \; .
\end{aligned}$$

The result follows. □

Lemma 5 *The polynomial $H(x)$ in Figure 2 can be computed using less than $3(n-r+1)(r+1)$ operations in $\mathbb{F}_{q^n}$ (where $r = \log_q \deg G(x)$).*

Proof Compute $H(x)$ by symbolic long division. □

Lemma 6 *Given the polynomial $H(x)$, the expected number of operations in $\mathbb{F}_{q^n}$ needed to compute the basis elements $H(z_0), H(z_1), \ldots, H(z_{r-1})$ in Figure 2 is less than $3n(r+2)$.*

Proof In iteration j (which selects z_j), the values

$$H(z_0), H(z_1), \ldots, H(z_{j-1})$$

are linearly independent over $\mathbb{F}_q$. Since $\deg H(x) = q^{n-r}$ and $H(x)$ (being a right symbolic divisor of $x^{q^n} - x$) divides $x^{q^n} - x$ in the ordinary sense, it follows that the size of the kernel of the linear mapping $x \mapsto H(x)$ is q^{n-r}. Therefore, when z_j is randomly selected from $\mathbb{F}_{q^n}$, the probability that $H(z_j)$ is not in the linear span of $\{H(z_\ell)\}_{\ell=0}^{j-1}$ equals

$$\frac{q^n - q^{n-r+j}}{q^n} = 1 - q^{j-r} \ ,$$

and the expected number of random selections until $H(z_j)$ satisfies this property is

$$\frac{1}{1-q^{j-r}} = 1 + \frac{1}{q^{r-j}-1} \le 2 \ .$$

Summing over j, the expected overall number of elements that are randomly selected in Figure 2 is

$$\sum_{j=0}^{r-1} \Bigl(1 + \frac{1}{q^{r-j}-1}\Bigr) < r + 2 \ .$$

Now, for each selected element z_j, we compute $H(z_j)$ using at most $3(n-r)+1$ operations in $\mathbb{F}_{q^n}$. Then, we check whether $H(z_j)$ is in the linear span of the set $\{H(z_\ell)\}_{\ell=0}^{j-1}$. To this end, we assume that the j elements of this set have been written as row vectors in $\mathbb{F}_q^n$ thereby forming a $j \times n$ matrix over $\mathbb{F}_q$, and that this matrix has been brought to an upper-echelon form; we then append $H(z_j)$ as a $(j{+}1)$st row to that matrix. Checking whether that row is linearly dependent of the previous rows can be done by Gaussian elimination, which, in turn, requires no more than $2j+1$ operations in $\mathbb{F}_{q^n}$ (specifically, each addition of rows in the matrix amounts to one addition in $\mathbb{F}_{q^n}$, and each multiplication by a scalar from $\mathbb{F}_q$ is over-counted as one multiplication in $\mathbb{F}_{q^n}$). Hence, the overall expected number of operations in $\mathbb{F}_{q^n}$ needed to find $H(z_0), H(z_1), \ldots, H(z_{r-1})$ is at most

$$\begin{aligned}\sum_{j=0}^{r-1} \Bigl(3(n-r) + 1 + (2j+1)\Bigr) \left(1 + \frac{1}{q^{r-j}-1}\right) \\ &< 3(n-r)(r+2) + 4\sum_{j=0}^{r-1}(j+1) \\ &< 3n(r+2) \ .\end{aligned}$$

□

Summing up the results of Lemmas 4, 5, and 6, we conclude that the overall number of operations in $\mathbb{F}_{q^n}$ of the algorithm in Figure 2 is less than $9(n+1)(t+2)$.

Acknowledgements The work of the first author was done in part while he was with the Computer Science Department, Technion—Israel Institute of Technology, Haifa 32000, Israel. This research was supported by the Israel Science Foundation (grant No. 746/04).

References

1. A.V. Aho, J.E. Hopcroft, J.D. Ullman, *The Design and Analysis of Computer Algorithms.* Reading, Massachusetts: Addison-Wesley, 1974.
2. D. Augot, P. Charpin, N. Sendrier, *Studying the locator polynomials of minimum weight codewords of BCH codes, IEEE Trans. Inform. Theory,* 38 (1992), 960–973.
3. M. Ben-Or, *Probabilistic algorithms in finite fields, Proc. 22nd Annual IEEE Symp. Foundations of Computer Science (FOCS'1981),* Nashville, Tennessee (1981), 394–398.
4. D. Coppersmith, S. Winograd, *Matrix multiplication via arithmetic progressions, J. Symb. Comput.,* 9 (1990), 251–280.
5. P. Delsarte, *Bilinear forms over a finite field, with applications to coding theory, J. Comb. Theory A,* 25 (1978), 226–241.
6. E.M. Gabidulin, *Theory of codes with maximum rank distance, Probl. Inform. Transm.,* 21 (1985), 1–12.
7. S. Gao, J. von zur Gathen, D. Panario, V. Shoup, *Algorithms for exponentiation in finite fields, J. Symb. Comput.,* 29 (2000), 879–889.
8. J. von zur Gathen, J. Gerhard, *Modern Computer Algebra.* Cambridge, UK: Cambridge University Press, 1999.
9. R. Lidl, H. Niederreiter, *Finite Fields,* Second Edition. Cambridge, UK: Cambridge University Press, 1997.
10. P. Loidreau, *A Welch–Berlekamp like algorithm for decoding Gabidulin codes, Proc. 4th International Workshop on Coding and Cryptography (WCC'2005),* Bergen, Norway (2005), 36–45.
11. F.J. MacWilliams, N.J.A. Sloane, *The Theory of Error-Correcting Codes.* Amsterdam, The Netherlands: North-Holland, 1977.
12. O. Ore, *On a special class of polynomials, Trans. Amer. Math. Soc.,* 35 (1933), 559–584.
13. O. Ore, *Contributions to the theory of finite fields, Trans. Amer. Math. Soc.,* 36 (1934), 243–274.
14. M.O. Rabin, *Probabilistic algorithms in finite fields, SIAM J. Comput.,* 9 (1980), 273–280.
15. R.M. Roth, *Maximum-rank array codes and their application to crisscross error correction, IEEE Trans. Inform. Theory,* 37 (1991), 328–336.
16. R.M. Roth, *Probabilistic crisscross error correction, IEEE Trans. Inform. Theory,* 43 (1997), 1425–1436.