MATHEMATICS OF COMPUTATION
Volume 79, Number 272, October 2010, Pages 2317–2332
S 0025-5718(10)02363-X
Article electronically published on April 20, 2010

# MORE ON SOLVING SYSTEMS OF POWER EQUATIONS

YINGQUAN WU

Abstract. It is known that a system of power equations can be reduced to a single-variable polynomial equation by exploiting the so-called *Newton's identities*. In this work, we investigate four new types of power equation systems. In the first two types we allow the powers to be a mix of positive and negative terms, whereas in the literature the system of power equations involves only positive powers. The first type involves only positive signs of powers, whereas the second type expands to involve both positive and negative signs. We present algebraic methods to solve the system and furthermore fully characterize the number of nontrivial solutions. The other two types are defined over finite fields and otherwise are the same as the conventional system of power equations. The methodology for solving the third type can be viewed as a generalization of the Berlekamp algorithm. The solution space of the last system is fully characterized despite the fact that the number of equations is two less than the number of unknowns.

## 1. Introduction

Systems of algebraic equations appear in many application areas of computational algebra, including communications, robotics, chemistry, and mechanics. In these algebraic equations, one commonly aims at carrying out eliminations in order to obtain a triangular system of equations which can be easily solved. The most widely known such methods include resultants, Groebner bases, and characteristic sets. On the other hand, a system of power polynomial equations, i.e., a system of the form

$$\sum_{i=1}^{n} x_i^j = s_j, \quad j = 1, 2, \ldots, n, \tag{1.1}$$

can be solved by reducing it to a single-variable polynomial equation using *Newton's identities*, which are based on simple relationships between elementary symmetric polynomials and power polynomials (cf. [3]).

In [7], Wu and Hadjicostis presented an algebraic solution of a generalized system of (1.1) with mixed positive and negative signs

$$\sum_{i=1}^{k} x_i^j - \sum_{i=1}^{n-k} y_i^j = s_j, \quad j = 1, 2, \ldots, n\,. \tag{1.2}$$

Received by the editor September 30, 2008 and, in revised form, July 24, 2009.
2010 *Mathematics Subject Classification.* Primary 12Y05; Secondary 65H10.
*Key words and phrases.* Power polynomial, composite power polynomial, Newton's identities, system of polynomial equations, generalized Berlekamp algorithm.

The core of an algebraic solution is a linear decomposition of elementary symmetric polynomials corresponding to positive and negative signs, respectively. This mathematical framework has been successfully applied to various situations. In [8], it is employed to decode BCH codes under the Lee metric (the Lee distance between $a$, $b \in \mathrm{GF}(p)$ is defined as the smaller value between $|a-b|$ and $p-|a-b|$, where $p$ is a prime.). In [9], it is employed for the fault identification in discrete event systems that are described by Petri nets (a Petri net is a directed bipartite graph, in which the nodes represent transitions, i.e., discrete events that may occur, places, i.e., conditions, and directed arcs that describe which places are pre- and/or postconditions for which transitions). More specifically, redundancy is incorporated into a given Petri net in a way that enables fault detection and identification to be performed efficiently, in a centralized or distributed manner, using the well-established algebraic coding technique in Hamming and Lee metrics. In [6], it is employed to characterize algebraically the inverse polynomial images of $[-1,\ 1]$, which consists of two Jordan arcs (a Jordan arc denotes a curve that does not cross itself and has no points missing, a curve that can be put into one-to-one correspondence with the closed interval from 0 to 1), by an explicit polynomial equation for the four endpoints of the arcs.

In this paper, we present four new types of systems of power polynomial equations. The first system is an extension of system (1.1) with powers having a mix of positive and negative signs, i.e.,

$$\sum_{i=1}^{n} x_i^j = s_j, \quad j = -k, \ldots, -1, 1, \ldots, n-k. \tag{1.3}$$

We first identify the intrinsic relation of elementary symmetric polynomials with respect to a set of variables and to their inverses. We then present an algebraic solution for the system. It is worth noting that the extended system may have an infinite number of solutions, or no solution at all in an arbitrary space, whereas the original system (1.1) always has a unique solution (in an appropriate space). Likewise, the second system is an extension of system (1.2) with the power spectrum crossing the zero boundary, i.e.,

$$\sum_{i=1}^{r} x_i^j - \sum_{i=1}^{n-r} y_i^j = s_j, \quad j = -k, \ldots, -1, 1, \ldots, n-k. \tag{1.4}$$

The proposed algebraic solution follows the new developments of the first system (1.3) and the system (1.2).

As an immediate application of the proposed second system, it effectively doubles the code-rate spectrum of BCH codes under the Lee metric. Specifically, letting $\alpha$ be a primitive element of $\mathrm{GF}(p^m)$, where $p$ is a prime, in literature, efficient algebraic decoding algorithms are derived to correct up to $t$ Lee errors for the BCH codes defined with spectral nulls at $\alpha^0 = 1$, $\alpha$, $\alpha^2$, $\ldots$, $\alpha^t$, $t < p$ [5, 8], whereas the proposed algebraic solution essentially provides an efficient decoding/correcting of up to $t_1 + t_2$ Lee errors for the BCH codes defined with spectral nulls at $\alpha^{-t_1}$, $\ldots$, $\alpha^{-1}$, $\alpha^0$, $\alpha^1$, $\ldots$, $\alpha^{t_2}$, $t_1, t_2 \leq p$. Here is the connection from the decoding of the newly defined BCH codes under the Lee metric to the proposed equation system. Let the received word polynomial $w(x) = c(x) + e(x)$, where $c(x) \stackrel{\triangle}{=} c_0 + c_1 x + c_2 x^2 + \cdots + c_{N-1} x^{N-1}$ denotes the prototype codeword polynomial and $e(x)$ denotes the corresponding Lee error polynomial with coefficients being either 1 or $-1$. The

decoding objective is to identify $e(x)$ so that $c(x)$ is successfully retrieved through $c(x) = w(x) - e(x)$. By definition, $c(\alpha^i) = 0$, $i = -t_1, -t_1 + 1, \ldots, t_2$. Evaluate $w(x)$ over the spectral nulls $\alpha^i$ and obtain $s_i = w(\alpha^i)$, for $i = -t_1, -t_1 + 1, \ldots, t_2$. The decoding problem is then reduced to the equation system

$$e(\alpha^i) = s_i, \quad i = -t_1, -t_1 + 1, \ldots, t_2. \tag{1.5}$$

Note that $e(\alpha^0) = s_0$ specifies the different number of positive signs ("1" coefficients) and negative signs ("$-1$" coefficients), which is further used to determine the valid choices of the numbers of positive and negative signs ($r$ and $n - r$, respectively, in this context). For each choice, the resulting equation system of (1.5) is a special form of the (general) system (1.4). Therefore, the proposed algebraic solution can be exploited to identify all candidate Lee error polynomials $e(x)$.

The third system limits the power equation system (1.1) to a finite field $\mathrm{GF}(p^m)$ (where $p$ is a prime). Under this constraint, the equations involving powers that are multiples of $p$ become redundant. Thus the effective system has the inconsecutive power spectrum

$$\sum_{i=1}^{n} x_i^j = s_j, \quad j = 1, \ldots, p-1, p+1, \ldots, 2p-1, 2p+1, \ldots, k, \tag{1.6}$$

where $k = \frac{np}{p-1}$. When the field has characteristic $p = 2$, the system is the key equation for decoding binary BCH codes, which was efficiently solved by the Berlekamp algorithm with quadratic complexity $O(n^2)$ [1]. Note that a straightforward trial-and-error effort takes quadruple complexity $O(n^4)$ to compute the minimum-length characteristic polynomial. We divide the system into $p - 1$ subsystem of linear-feedback shift registers and devise a generalized Berlekamp algorithm to compute the minimum-length characteristic polynomial with only quadratic complexity $O(n^2)$. The proposed generalized Berlekamp algorithm is akin to the fundamental iterative algorithm which generalizes the Berlekamp-Massey algorithm to solve multi-sequence linear-feedback shift registers [4]. As an immediate application, the proposed algebraic solution can be used to complement the hard-decision decoding of ternary BCH codes. Specifically, observing that a "$-1$" Hamming error can be viewed as two "1" Lee errors, likewise that a "1" Hamming error can be viewed as two "$-1$" Lee errors in $\mathrm{GF}(3^m)$, when $N_1 + N_{-1} > \frac{k}{2}$ and when $N_1 + 2N_{-1} < \frac{2k}{3}$ or $2N_1 + N_{-1} < \frac{2k}{3}$, where $N_1$ and $N_{-1}$ denote the number of "1" and "$-1$" errors respectively, the proposed solution may potentially provide the correction which otherwise fails by hard-decision decoding (which corrects up to $N_1 + N_{-1} \leq \frac{k}{2}$ Hamming errors).

The last system is a special power equation system in $\mathrm{GF}(p^m)$ (where $p$ is a prime)

$$\sum_{i=1}^{p} x_i^j = 0, \quad j = 1, 2, \ldots, p-2. \tag{1.7}$$

We identify explicitly all nontrivial solutions and particularly determine precisely the number of distinct solutions. This system sheds lights on the code spectrum of the Lee metric BCH codes defined in [5, 8].

## 2. Preliminaries

In this section we review the necessary background on *power polynomial equations* and *Newton's identities*. For notational simplicity, we use $\mathbf{x}_n$ to denote a set of $n$ variables named by $x$ and with subscription "$_1, _2, \ldots, _n$", namely, $\{x_1, x_2, \ldots, x_n\}$.

*A power polynomial* is defined as

$$\mathcal{S}_k(\mathbf{x}_n) \triangleq \sum_{i=1}^{n} x_i^k \tag{2.1}$$

and a *system of power polynomial equations* is usually represented as in (1.1). It is well known that the system in (1.1) can be solved by employing *Newton's identities* [3]. The identities display a simple relation between the *elementary symmetric polynomials* given by

$$\Lambda_k(\mathbf{x}_n) \triangleq (-1)^k \sum_{1\le i_1<i_2<\cdots<i_k\le n} x_{i_1}x_{i_2}\ldots x_{i_k}, \quad 1 \le k \le n \tag{2.2}$$

(for consistency, we set $\Lambda_0(\mathbf{x}_n) = 1$) and the power polynomials $\{\mathcal{S}_i(\mathbf{x}_n)\}_{i=1}^n$ in (2.1). Newton's identities take the form

$$\begin{cases} \mathcal{S}_k + \Lambda_1\mathcal{S}_{k-1} + \cdots + \Lambda_{k-1}\mathcal{S}_1 + k\Lambda_k = 0, & 1 \le k \le n, \\ \mathcal{S}_k + \Lambda_1\mathcal{S}_{k-1} + \cdots + \Lambda_{n-1}\mathcal{S}_{k-n+1} + \Lambda_n\mathcal{S}_{k-n} = 0, & k > n, \end{cases} \tag{2.3}$$

where $(\mathbf{x}_n)$ is omitted for simplicity, and they *remain true over an arbitrary field* [1]. If $\mathcal{S}_1(\mathbf{x}_n), \mathcal{S}_2(\mathbf{x}_n), \ldots, \mathcal{S}_n(\mathbf{x}_n)$ are known as $s_1, s_2, \ldots, s_n$, respectively, and $\frac{1}{n!}$ is defined in the given field, we can use (2.3) to obtain the *linear* equation array

$$\begin{cases} s_1 + \Lambda_1(\mathbf{x}_n) & = 0, \\ s_2 + \Lambda_1(\mathbf{x}_n)s_1 + 2\Lambda_2(\mathbf{x}_n) & = 0, \\ & \vdots \\ s_n + \Lambda_1(\mathbf{x}_n)s_{n-1} + \cdots + \Lambda_{n-1}(\mathbf{x}_n)s_1 + n\Lambda_n(\mathbf{x}_n) & = 0 \end{cases} \tag{2.4}$$

and can easily solve for $\Lambda_k(\mathbf{x}_n) = \mathcal{F}_k(\mathbf{s}_k)$, $k = 1, 2, \ldots, n$. More specifically,

$$\mathcal{F}_k(\mathbf{s}_k) = \frac{(-1)^k}{k!} \begin{vmatrix} s_1 & 1 & 0 & \ldots & 0 & 0 \\ s_2 & s_1 & 2 & \ldots & 0 & 0 \\ s_3 & s_2 & s_1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ s_{k-1} & s_{k-2} & s_{k-3} & \ldots & s_1 & k-1 \\ s_k & s_{k-1} & s_{k-2} & \ldots & s_2 & s_1 \end{vmatrix}. \tag{2.5}$$

We remark that the solution for $\Lambda_k(\mathbf{x}_n)$ in terms of $\{s_i\}_{i=1}^k$ is independent of the number of original variables $n$. Moreover, the above expression does *not* hold if the inverse $\frac{1}{k!}$ is not defined in the given finite field. Also note that the equation system (2.4) is a *lower triangular Toeplitz linear* system, whose solution requires $O(n \log n)$ arithmetic operations [2].

Once $\{\Lambda_i(\mathbf{x}_n)\}_{i=1}^n$ are available, the solutions $x_1, x_2, \ldots, x_n$ of (1.1) are exactly the roots of the *characteristic polynomial*

$$x^n + \Lambda_1(\mathbf{x}_n)x^{n-1} + \cdots + \Lambda_{n-1}(\mathbf{x}_n)x + \Lambda_n(\mathbf{x}_n) = \prod_{i=1}^{n}(x - x_i) \tag{2.6}$$

and can be solved using well-established methods.

For notational simplicity, we define $\mathcal{F}_0(\mathbf{s}_0) \stackrel{\triangle}{=} 1$. The function $\mathcal{F}(\cdot)$ exhibits the following decomposition property [7].

**Proposition 2.1.** *Let $x_1, x_2, \ldots, x_n$ be $n$ variables and let $s_1, s_2, \ldots, s_k$ be $k$ given (known) parameters. Then,*

$$\mathcal{F}_k(s_1+\mathcal{S}_1(\mathbf{x}_n),\ s_2+\mathcal{S}_2(\mathbf{x}_n),\ \ldots,\ s_k+\mathcal{S}_k(\mathbf{x}_n)) = \sum_{i=0}^{\min\{n,\,k\}} \mathcal{F}_{k-i}(\mathbf{s}_{k-i})\cdot\Lambda_i(\mathbf{x}_n) \tag{2.7}$$

*for $i = 1,\ 2,\ \ldots,\ n$.*

The system of composite power equations (1.2) is first studied in [7] and subsequently improved in [8]. Without loss of generality, we assume $2k \geq n$. We first rearrange the system (1.2)

$$\begin{cases} x_1+\cdots+x_k=s_1+y_1+\cdots+y_{n-k}, \\ x_1^2+\cdots+x_k^2=s_2+y_1^2+\cdots+y_{n-k}^2, \\ \qquad\qquad\vdots \\ x_1^n+\cdots+x_k^n=s_n+y_1^n+\cdots+y_{n-k}^n. \end{cases} \tag{2.8}$$

Applying Proposition 2.1 with respect to $j = k+1,\ k+2,\ \ldots,\ n$, we obtain the following linear system with respect to $\Lambda_i(\mathbf{y}_{n-k})$, $i = 1, 2, \ldots, n-k$:

$$\begin{cases} \Lambda_{k+1}(\mathbf{x}_k) = 0 = \sum_{i=0}^{n-k} \mathcal{F}_{k+1-i}(\mathbf{s}_{k+1-i})\Lambda_i(\mathbf{y}_{n-k}), \\ \Lambda_{k+2}(\mathbf{x}_k) = 0 = \sum_{i=0}^{n-k} \mathcal{F}_{k+2-i}(\mathbf{s}_{k+2-i})\Lambda_i(\mathbf{y}_{n-k}), \\ \qquad\qquad\vdots \\ \Lambda_n(\mathbf{x}_k) \quad = 0 = \sum_{i=0}^{n-k} \mathcal{F}_{n-i}(\mathbf{s}_{n-i})\Lambda_i(\mathbf{y}_{n-k}), \end{cases} \tag{2.9}$$

which can be effectively solved by the Berlekamp-Massey algorithm with complexity $O((n-k)^2)$, regardless of the number of actual nontrivial roots among $\{y_i\}_{i=1}^{n-k}$ [1]. Subsequently, $\Lambda_i(\mathbf{x}_k)$, $i = 1, 2, \ldots, k$, can be obtained through applying Proposition 2.1 with respect to $j = 1, 2, \ldots, k$:

$$\begin{cases} \Lambda_1(\mathbf{x}_k) &= \mathcal{F}_1(\mathbf{s}_1) + \Lambda_1(\mathbf{y}_{n-k}), \\ \Lambda_2(\mathbf{x}_k) &= \mathcal{F}_2(\mathbf{s}_2) + \mathcal{F}_1(s_1)\Lambda_1(\mathbf{y}_{n-k}) + \Lambda_2(\mathbf{y}_{n-k}), \\ &\vdots \\ \Lambda_k(\mathbf{x}_k) &= \sum_{i=0}^{n-k} \mathcal{F}_{n-k-i}(\mathbf{s}_{n-k-i})\Lambda_i(\mathbf{y}_{n-k}). \end{cases} \tag{2.10}$$

Finally, $\{x_i\}_{i=1}^k$ and $\{y_i\}_{i=1}^{n-k}$ can be solved separately utilizing the corresponding characteristic equation as defined in (2.6).

We proceed to introduce the Berlekamp algorithm. In essence, the Berlekamp algorithm effectively determines the minimum-length characteristic polynomial (which is alternatively called the error-locator polynomial, or linear-feedback shift

register (LFSR)) of the following system of Newton's identities in GF($2^m$):

$$
\begin{cases}
0{=}s_1+\Lambda_1(\mathbf{x}_n),\\
0{=}s_3+s_2\Lambda_1(\mathbf{x}_n)+s_1\Lambda_2(\mathbf{x}_n)+\Lambda_3(\mathbf{x}_n),\\
0{=}s_5+s_4\Lambda_1(\mathbf{x}_n)+s_3\Lambda_2(\mathbf{x}_n)+s_2\Lambda_3(\mathbf{x}_n)+s_1\Lambda_4(\mathbf{x}_n)+\Lambda_5(\mathbf{x}_n),\\
\quad\vdots\\
0{=}s_{k-3}+s_{k-4}\Lambda_1(\mathbf{x}_n)+\cdots+s_{k-n-2}\Lambda_{n-1}(\mathbf{x}_n)+s_{k-n-3}\Lambda_n(\mathbf{x}_n),\\
0{=}s_{k-1}+s_{k-2}\Lambda_1(\mathbf{x}_n)+\cdots+s_{k-n}\Lambda_{n-1}(\mathbf{x}_n)+s_{k-n-1}\Lambda_n(\mathbf{x}_n),
\end{cases}
\tag{2.11}
$$

where $k$ is an even number.

The above system is the key equation for decoding binary BCH codes which has a unique solution when the number of variables $n$ is up to half the number of $\{s_i\}_{i=1}^{k}$, i.e., $2n \le k$ [1]. When $2n \le k$, the minimum-length LSFR determined by the Berlekamp algorithm turns out to be the unique (and valid) solution.

The detailed Berlekamp algorithm is described below, followed by its well-established characterizations [1].

**Berlekamp algorithm.**

- Input: $[s_1, s_2, \ldots, s_k]$.
- Initialization: Set $\Lambda^{(0)}(x) = B^{(0)}(x) = 1$, $L_\Lambda^{(0)} = L_B^{(0)} = 0$.
- For $j = 0, 1, 2, \ldots, k$, do:
  - Let $j+1 = 2q + r$, where $0 \le r < 2$. If $r = 0$, then continue for next iteration.
  - Compute $\Delta = \sum_{i=0}^{L_\Lambda^{(j)}} s_{j+1-i}\Lambda_i^{(j)}$.
  - Update $\Lambda^{(j+1)}(x) = \Lambda^{(j)}(x) - \Delta x B^{(j)}(x)$.
  - If $\Delta \neq 0$ and $L_\Lambda^{(j)} \le L_B^{(j)}$, then set

$$B^{(j+1)}(x) \leftarrow \Delta^{-1}x\Lambda^{(j)}(x), \quad L_B^{(j+1)} \leftarrow L_\Lambda^{(j)} + 1, \quad L_\Lambda^{(j+1)} \leftarrow L_B^{(j)} + 1.$$

    Else set

$$B^{(j+1)}(x) \leftarrow x^2 B^{(j)}(x), \quad L_B^{(j+1)} \leftarrow L_B^{(j)} + 2, \quad L_\Lambda^{(j+1)} \leftarrow L_\Lambda^{(j)}.$$

- Output: $\Lambda^{(k)}(x)$, $L_\Lambda^{(k)}$.

**Proposition 2.2.** *At any iteration $j$ of the Berlekamp algorithm:*

(i) *The lengths of the LFSRs $L_\Lambda^{(j)}$ and $L_B^{(j)}$ satisfy $L_\Lambda^{(j)} + L_B^{(j)} = j$.*

(ii) $\deg(\Lambda^{(j)}(x)) \le L_\Lambda^{(j)}$ *and* $\deg(B^{(j)}(x)) \le L_B^{(j)}$.

(iii) *The discrepancy $\Delta$ is always zero when $j$ is even (due to that 2 is the characteristic of the field).*

(iv) $\Lambda^{(j)}(x)$ *is the minimum-length LFSR (with normalized constant term) to satisfy the linear constraints with $s_1, s_2, \ldots, s_j$.*

(v) $xB^{(j)}(x)$ *is the minimum-length LFSR (with zero constant term) to satisfy the linear constraints involved with $s_1, s_2, \ldots, s_j$.*

## 3. Solving systems of power polynomial equations

3.1. **Type-A system of power equations.** In this subsection, we consider the power equation system

$$\left\{\begin{array}{llll} x_1^{-k} & +x_2^{-k} & +\cdots+x_n^{-k} & =s_{-k},\\ & & & \vdots\\ x_1^{-1} & +x_2^{-1} & +\cdots+x_n^{-1} & =s_{-1},\\ x_1^{1} & +x_2^{1} & +\cdots+x_n^{1} & =s_{1},\\ & & & \vdots\\ x_1^{n-k} & +x_2^{n-k} & +\cdots+x_n^{n-k} & =s_{n-k}, \end{array}\right. \tag{3.1}$$

where $s_1, s_2, \ldots, s_n$ are known parameters and $0 < k < n$.

The Newton identities indicate that

$$\Lambda_i(\mathbf{x}_n^{-1}) = \mathcal{F}_i(\mathbf{s}_{-i}), \quad i = 1, 2, \ldots, k. \tag{3.2}$$

where $\mathbf{x}_n^{-1}$ and $\mathbf{s}_{-i}$ denote the sets $\{x_1^{-1}, x_2^{-2}, \ldots, x_n^{-1}\}$ and $\{s_{-1}, s_{-2}, \ldots, s_{-i}\}$, respectively, and

$$\Lambda_i(\mathbf{x}_n) = \mathcal{F}_i(\mathbf{s}_i), \quad i = 1, 2, \ldots, n-k. \tag{3.3}$$

The following lemma identifies the inherent connection between $\Lambda_i(\mathbf{x}_n^{-1})$ and $\Lambda_{n-i}(\mathbf{x}_n)$.

**Lemma 3.1.** *If $x_1, x_2, \ldots, x_n$ are nonzero, then*

$$\Lambda_i(\mathbf{x}_n^{-1}) = \frac{\Lambda_{n-i}(\mathbf{x}_n)}{\Lambda_n(\mathbf{x}_n)}, \quad i = 1, 2, \ldots, n. \tag{3.4}$$

The proof follows the equalities below

$$\begin{aligned} x^n + x^{n-1}\Lambda_1(\mathbf{x}_n^{-1}) + \cdots + x\Lambda_{n-1}(\mathbf{x}_n^{-1}) + \Lambda_n(\mathbf{x}_n^{-1}) &= (x - x_1^{-1})(x - x_2^{-1})\ldots(x - x_n^{-1})\\ &= x^n \frac{(x_1 - x^{-1})(x_2 - x^{-1})\ldots(x_n - x^{-1})}{x_1 x_2 \ldots x_n}\\ &= \frac{x^n}{\Lambda_n(\mathbf{x}_n)}(x^{-1} - x_1)(x^{-1} - x_2)\ldots(x^{-1} - x_n)\\ &= \frac{x^n}{\Lambda_n(\mathbf{x}_n)}\left(x^{-n} + x^{-n+1}\Lambda_1(\mathbf{x}_n) + \cdots + x^{-1}\Lambda_{n-1}(\mathbf{x}_n) + \Lambda_n(\mathbf{x}_n)\right)\\ &= x^n + x^{n-1}\frac{\Lambda_{n-1}(\mathbf{x}_n)}{\Lambda_n(\mathbf{x}_n)} + \cdots + x^1\frac{\Lambda_1(\mathbf{x}_n)}{\Lambda_n(\mathbf{x}_n)} + \frac{1}{\Lambda_n(\mathbf{x}_n)}. \end{aligned}$$

The above assertion immediately yields

$$\mathcal{F}_k(\mathbf{s}_{-k}) = \Lambda_k(\mathbf{x}_n^{-1}) = \frac{\Lambda_{n-k}(\mathbf{x}_n)}{\Lambda_n(\mathbf{x}_n)} = \frac{\mathcal{F}_{n-k}(\mathbf{s}_{n-k})}{\Lambda_n(\mathbf{x}_n)}. \tag{3.5}$$

*Case* 1. $\mathcal{F}_k(\mathbf{s}_{-k}) \neq 0$ and $\mathcal{F}_{n-k}(\mathbf{s}_{n-k}) \neq 0$. By (3.5)

$$\Lambda_n(\mathbf{x}_n) = \frac{\mathcal{F}_{n-k}(\mathbf{s}_{n-k})}{\mathcal{F}_k(\mathbf{s}_{-k})} \tag{3.6}$$

and, using (3.2) and (3.4),

$$\Lambda_i(\mathbf{x}_n) = \frac{\mathcal{F}_{n-i}(\mathbf{s}_{-(n-i)})\mathcal{F}_{n-k}(\mathbf{s}_{n-k})}{\mathcal{F}_k(\mathbf{s}_{-k})}, \tag{3.7}$$

for $i = n-k+1, n-k+2, \ldots, n-1$.

*Case* 2. $\mathcal{F}_k(\mathbf{s}_{-k}) = 0$ and $\mathcal{F}_{n-k}(\mathbf{s}_{n-k}) = 0$. In this case $\Lambda_n(\mathbf{x}_n)$ may take an arbitrary nonzero value, say, $a$ ($a \neq 0$), and it follows that

$$\Lambda_i(\mathbf{x}_n) = a\mathcal{F}_{n-i}(\mathbf{s}_{-(n-i)}), \tag{3.8}$$

for $i = n-k+1, n-k+2, \ldots, n-1$.

*Case* 3. $\mathcal{F}_k(\mathbf{s}_{-k}) \neq 0$ and $\mathcal{F}_{n-k}(\mathbf{s}_{n-k}) = 0$, or $\mathcal{F}_k(\mathbf{s}_{-k}) = 0$ and $\mathcal{F}_{n-k}(\mathbf{s}_{n-k}) \neq 0$. Clearly, no solution exists for this case.

**Example 3.2.** Consider the following system of power equations:

$$\begin{cases} x_1^{-1}+x_2^{-1}+x_3^{-1}=0, \\ x_1^1 \ +x_2^1 \ +x_3^1 \ =7, \\ x_1^2 \ +x_2^2 \ +x_3^2 \ =49. \end{cases}$$

It follows that

$$\begin{aligned} &\Lambda_1(\mathbf{x}_3) = \mathcal{F}_1(7) = -7, \qquad \Lambda_2(\mathbf{x}_3) = \mathcal{F}_2(7,49) = 0, \\ &\Lambda_1(\mathbf{x}_3^{-1}) = \tfrac{\Lambda_2(\mathbf{x}_3)}{\Lambda_3(\mathbf{x}_3)} = \mathcal{F}_1(0) = 0. \end{aligned}$$

Note that $\Lambda_2(\mathbf{x}_3)$ and $\Lambda_1(\mathbf{x}_3^{-1})$ are both zero; thus the system has an infinite number of solutions. The preceding analysis indicates that $\mathbf{x}_3$ is the set of roots of $x^3 - 7x^2 + a = 0$, where $a$ is a parameter.

3.2. **Type-B system of power equations.** In this subsection, we consider the composite power equation system

$$\begin{cases} x_1^{-k} \ +\cdots+x_r^{-k} \ -y_1^{-k} \ -\cdots-y_{n-r}^{-k} = s_{-k}, \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\ x_1^{-1} \ +\cdots+x_r^{-1} \ -y_1^{-1} \ -\cdots-y_{n-r}^{-1} = s_{-1}, \\ x_1^{1} \ +\cdots+x_r^{1} \ -y_1^{1} \ -\cdots-y_{n-r}^{1} = s_{1}, \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\ x_1^{n-k}+\cdots+x_r^{n-k}-y_1^{n-k}-\cdots-y_{n-r}^{n-k} = s_{n-k}, \end{cases} \tag{3.9}$$

where $s_{-k}, \ldots, s_{-1}, s_1, \ldots, s_{n-k}$ are known parameters and $0 < k, r < n$. We are interested in nontrivial solutions $\mathbf{x}_r \cup \mathbf{y}_{n-r}$, such that $\mathbf{x}_r \cap \mathbf{y}_{n-r} = \emptyset$.

We first move $y$'s terms to the right side:

$$\begin{cases} x_1^{-k} \ +\cdots+x_r^{-k} \ = s_{-k} \ +y_1^{-k} \ +\cdots+y_{n-r}^{-k}, \\ \qquad\qquad\qquad\qquad \vdots \\ x_1^{-1} \ +\cdots+x_r^{-1} \ = s_{-1} \ +y_1^{-1} \ +\cdots+y_{n-r}^{-1}, \\ x_1^{1} \ +\cdots+x_r^{1} \ = s_{1} \ +y_1^{1} \ +\cdots+y_{n-r}^{1}, \\ \qquad\qquad\qquad\qquad \vdots \\ x_1^{n-k}+\cdots+x_r^{n-k} = s_{n-k}+y_1^{n-k}+\cdots+y_{n-r}^{n-k}. \end{cases} \tag{3.10}$$

It suffices to assume $2r \leq n$; otherwise, (3.9) can be rearranged in such a way that it exhibits precisely the same form as (3.10).

Applying Proposition 2.1, we obtain

$$\Lambda_i(\mathbf{x}_r) = \sum_{j=0}^{\min(i,n-r)} \mathcal{F}_{i-j}(\mathbf{s}_{i-j})\Lambda_j(\mathbf{y}_{n-r}) \tag{3.11}$$

for $i = 1, 2, \ldots, n-k$ and

$$\Lambda_i(\mathbf{x}_r^{-1}) = \sum_{j=0}^{\min(i,n-r)} \mathcal{F}_{i-j}(\mathbf{s}_{-(i-j)})\Lambda_j(\mathbf{y}_{n-r}^{-1}) \tag{3.12}$$

for $i = 1, 2, \ldots, k$.

In the following, we focus on the case $k \geq r$, whereas the alternative case $k < r$ can be solved similarly. Note that (3.12) can be divided into two different categories:

$$\Lambda_i(\mathbf{x}_r^{-1}) = \sum_{j=0}^{\min(i,n-r)} \mathcal{F}_{i-j}(\mathbf{s}_{-(i-j)})\Lambda_j(\mathbf{y}_{n-r}^{-1}) \tag{3.13}$$

for $i = 1, 2, \ldots, r$ and

$$0 = \sum_{j=0}^{\min(i,n-r)} \mathcal{F}_{i-j}(\mathbf{s}_{-(i-j)})\Lambda_j(\mathbf{y}_{n-r}^{-1}) \tag{3.14}$$

for $i = r+1, r+2, \ldots, k$. Define

$$a \triangleq \frac{\Lambda_{n-r}(\mathbf{y}_{n-r})}{\Lambda_r(\mathbf{x}_r)}. \tag{3.15}$$

Applying Lemma 3.1, we rewrite (3.13) and (3.14) as

$$\Lambda_{r-i}(\mathbf{x}_r) \cdot a = \sum_{j=0}^{\min(i,n-r)} \mathcal{F}_{i-j}(\mathbf{s}_{-(i-j)})\Lambda_{n-r-j}(\mathbf{y}_{n-r}) \tag{3.16}$$

for $i = 1, 2, \ldots, r$, and

$$0 = \sum_{j=0}^{\min(i,n-r)} \mathcal{F}_{i-j}(\mathbf{s}_{-(i-j)})\Lambda_{n-r-j}(\mathbf{y}_{n-r}) \tag{3.17}$$

for $i = r+1, r+2, \ldots, k$.

*Case* 1. $n-k < r$. We combine (3.11), (3.16), and (3.17) to solve for $a$. As can be seen from (3.16), $a$ is a root of a polynomial with degree up to $r$. For each distinct solution of $a$, $\{\Lambda_i(\mathbf{x}_r)\}_{i=1}^{r}$ and $\{\Lambda_i(\mathbf{y}_{n-r})\}_{i=1}^{n-r}$ can be uniquely determined, if the corresponding determinant is nonsingular. Otherwise, $r$ is reduced by 1 and $n$ is reduced by 2 and the corresponding determinant must be checked again. The procedure is repeated until the determinant is nonsingular, and thus $\{\Lambda_i(\mathbf{x}_r)\}_{i=1}^{r}$ and $\{\Lambda_i(\mathbf{y}_{n-r})\}_{i=1}^{n-r}$ can be uniquely determined.

*Case* 2. $n-k \geq r$. We observe that (3.11) with $i = r$ can be expressed as

$$\Lambda_r(\mathbf{x}_r^{-1}) = \frac{1}{a}\Lambda_{n-r}(\mathbf{y}_{n-r}) = \sum_{j=0}^{\min(r,n-r)} \mathcal{F}_{r-j}(\mathbf{s}_{-(r-j)})\Lambda_j(\mathbf{y}_{n-r}^{-1}). \tag{3.18}$$

Thus, combining (3.11), (3.16), and (3.17) indicates that $a$ is a root of a polynomial with degree at most $r+1$.

It is worth noting that when the system composed of (3.11) and (3.12) is linearly dependent (where $a$ is treated as a variable), $a$ has an infinite number of solutions.

**Example 3.3.** Consider the follow system of power equations:

$$\begin{cases} x_1^{-2}+x_2^{-2}+x_3^{-2}+x_4^{-2}-y_1^{-2}=-1, \\ x_1^{-1}+x_2^{-1}+x_3^{-1}+x_4^{-1}-y_1^{-1}=1, \\ x_1^{1}+x_2^{1}+x_3^{1}+x_4^{1}-y_1^{1}=1, \\ x_1^{2}+x_2^{2}+x_3^{2}+x_4^{2}-y_1^{2}=-1, \\ x_1^{3}+x_2^{3}+x_3^{3}+x_4^{3}-y_1^{3}=1. \end{cases}$$

We first reformulate the above system as

$$\begin{cases} y_1^{-2}=1+x_1^{-2}+x_2^{-2}+x_3^{-2}+x_4^{-2}, \\ y_1^{-1}=-1+x_1^{-1}+x_2^{-1}+x_3^{-1}+x_4^{-1}, \\ y_1^{1}=-1+x_1^{1}+x_2^{1}+x_3^{1}+x_4^{1}, \\ y_1^{2}=1+x_1^{2}+x_2^{2}+x_3^{2}+x_4^{2}, \\ y_1^{3}=-1+x_1^{3}+x_2^{3}+x_3^{3}+x_4^{3}. \end{cases}$$

It follows that

$$\Lambda_1(y_1) = 1 + \Lambda_1(\mathbf{x}_4), \tag{3.19}$$

$$0 = \Lambda_1(\mathbf{x}_4) + \Lambda_2(\mathbf{x}_4), \tag{3.20}$$

$$0 = \Lambda_2(\mathbf{x}_4) + \Lambda_3(\mathbf{x}_4), \tag{3.21}$$

$$\Lambda_1(y_1^{-1}) = 1 + \Lambda_1(\mathbf{x}_4^{-1}), \tag{3.22}$$

$$0 = \Lambda_1(\mathbf{x}_4^{-1}) + \Lambda_2(\mathbf{x}_4^{-1}). \tag{3.23}$$

We observe that (3.23) is equivalent to

$$0 = \frac{\Lambda_3(\mathbf{x}_4)}{\Lambda_4(\mathbf{x}_4)} + \frac{\Lambda_2(\mathbf{x}_4)}{\Lambda_4(\mathbf{x}_4)},$$

which is linearly independent to (3.21), and (3.22) is equivalent to

$$\frac{1}{\Lambda_1(y_1)} = 1 + \frac{\Lambda_3(\mathbf{x}_4)}{\Lambda_4(\mathbf{x}_4)}, \tag{3.24}$$

As a result, there are five variables and only four equations: (3.19), (3.20), (3.21), and (3.24). Consequently, $a \triangleq \frac{\Lambda_4(\mathbf{x}_4)}{\Lambda_1(y_1)}$, or alternatively, $\Lambda_1(y_1)$, may take on any nonzero value. If choosing $\Lambda_1(y_1) = 1$, then $\Lambda_i(\mathbf{x}_4) = 0, \ \ i = 1, 2, 3$, whereas $\Lambda_4(\mathbf{x}_4)$ can be arbitrary. Thus, we obtain an infinite family of solutions

$$y_1 = 1, \quad \mathbf{x}_4 = \{f^{1/4}, \ -f^{1/4}, \ i \cdot f^{1/4}, \ -i \cdot f^{1/4}\}, \ \ \forall f.$$

For any other nontrivial choice of $\Lambda_1(y_1)$, there is a unique solution $\Lambda_4(\mathbf{x}_4)$. Since the system that is composed of (3.19), (3.20), (3.21), (3.22), and (3.23) is linearly dependent, the original system of power equations is reduced to

$$\begin{cases} x_1^{-2}+x_2^{-2}+x_3^{-2}=-1, \\ x_1^{-1}+x_2^{-1}+x_3^{-1}=1, \\ x_1^{1}+x_2^{1}+x_3^{1}=1, \\ x_1^{2}+x_2^{2}+x_3^{2}=-1, \\ x_1^{3}+x_2^{3}+x_3^{3}=1, \end{cases}$$

which yields the unique solution $\mathbf{x}_3 = \{1, \ i, \ -i\}$.

3.3. **Type-C system of power equations.** In this section, we consider the following system defined in $\mathrm{GF}(p^m)$, where $p$ is a prime:

$$\begin{cases} x_1+x_2+x_3+\cdots+x_{n-1}+x_n= s_1, \\ x_1^2+x_2^2+x_3^2+\cdots+x_{n-1}^2+x_n^2= s_2, \\ \qquad\qquad\qquad\qquad\qquad\vdots \\ x_1^k+x_2^k+x_3^k+\cdots+x_{n-1}^k+x_n^k= s_k, \end{cases} \tag{3.25}$$

where $k \triangleq \lfloor \frac{np}{p-1} \rfloor$ and the unknown $\mathbf{x}_n \in \mathrm{GF}(p^m)$ and $s_1, s_2, \ldots, s_k$ are known parameters. Note that due to the nature of finite field operation,

$$s_{pi} = \sum_{j=1}^{n} x_j^{pi} = (\sum_{j=1}^{n} x_j^i)^p = s_i^p. \tag{3.26}$$

Therefore, the equations associated with $s_{pi}$ $(i > 0)$ are redundant. Newton's identities yield

$$\begin{cases} s_1 & & & +1\cdot\Lambda(\mathbf{x}_n) & = 0, \\ & \vdots & & & \vdots \\ s_{p-1} & +s_{p-2}\Lambda_1(\mathbf{x}_n) & +\cdots+s_1\Lambda_{p-2}(\mathbf{x}_n) & +(p-1)\Lambda_{p-1}(\mathbf{x}_n) & = 0, \\ s_{p+1} & +s_p\Lambda_1(\mathbf{x}_n) & +\cdots+s_1\Lambda_p(\mathbf{x}_n) & +1\cdot\Lambda_{p-1}(\mathbf{x}_n) & = 0, \\ & \vdots & & & \vdots \\ s_{2p-1} & +s_{2p-2}\Lambda_1(\mathbf{x}_n) & +\cdots+s_1\Lambda_{2p-2}(\mathbf{x}_n) & +(p-1)\Lambda_{2p-1}(\mathbf{x}_n) & = 0, \\ & \vdots & & & \vdots \\ s_{n+1} & +s_n\Lambda_1(\mathbf{x}_n) & +\cdots+s_2\Lambda_{n-1}(\mathbf{x}_n) & +s_1\Lambda_n(\mathbf{x}_n) & = 0, \\ & \vdots & & & \vdots \\ s_k & +s_{k-1}\Lambda_1(\mathbf{x}_n) & +\cdots+s_{k-n+1}\Lambda_{n-1}(\mathbf{x}_n) & +s_{k-n}\Lambda_n(\mathbf{x}_n) & = 0. \end{cases} \tag{3.27}$$

Note that when $p = 2$, the system (3.27) is the decoding key equation for binary BCH codes and is efficiently solved by the Berlekamp algorithm with complexity $O(n^2)$. Moreover, there is at most a unique solution associated with $n \le \frac{k}{2}$. When $p > 2$, the above system can no longer be viewed as a single LFSR system, but as $p-1$ independent LFSR systems, corresponding to the last term $1\cdot\Lambda(\mathbf{x}_n)$, $2\cdot\Lambda(\mathbf{x}_n)$, $\ldots$, $(p-1)\cdot\Lambda(\mathbf{x}_n)$, respectively. Moreover, there may exist multiple solutions associated with $n \le \frac{k(p-1)}{p}$, as illustrated in the following example.

**Example 3.4.** Let $\alpha$ be a primitive element in $\mathrm{GF}(3^4)$.

(i) Let $\mathbf{s}_{12} = [\alpha^{25}, \alpha^{61}, \alpha^{75}, \alpha^{19}, \alpha^{11}, \alpha^{23}, \alpha^{79}, \alpha^{57}, \alpha^{65}, 0, \alpha^{77}, \alpha^{57}]$, which yields a system of eight linearly independent equations. Both of the following polynomials satisfy the linear constraints imposed by (3.27):

$$\Lambda_1(x) = 1 + \alpha^{65}x^1 + \alpha^{16}x^2 + \alpha^{38}x^3 + \alpha^{37}x^4 + \alpha^{68}x^5 + \alpha^{38}x^6 + \alpha^{39}x^7 + \alpha^{70}x^8,$$
$$\Lambda_2(x) = 1 + \alpha^{65}x^1 + \alpha^{16}x^2 + \alpha^{71}x^4 + \alpha^{22}x^5 + \alpha^{72}x^6.$$

(ii) Let $\mathbf{s}_{12} = [\alpha^{41}, \alpha^{12}, \alpha^{43}, \alpha^{17}, \alpha^{61}, \alpha^{36}, \alpha^{19}, \alpha^{40}, \alpha^{49}, 0, \alpha^{56}, \alpha^{51}]$. Both of the following polynomials satisfy the linear constraints imposed by (3.27):

$$\Lambda_1(x) = 1 + \alpha^{1}x^1 + \alpha^{62}x^2 + \alpha^{68}x^3 + \alpha^{31}x^4 + \alpha^{65}x^5 + \alpha^{77}x^6 + \alpha^{31}x^7 + \alpha^{25}x^8,$$
$$\Lambda_2(x) = 1 + \alpha^{1}x^1 + \alpha^{62}x^2 + \alpha^{66}x^3 + \alpha^{72}x^4 + \alpha^{43}x^6 + \alpha^{44}x^7 + \alpha^{25}x^8.$$

We next present a generalized Berlekamp algorithm to determine the minimum-length LFSR satisfying the above system with the same complexity $O(n^2)$, whereas a straightforward effort will require $O(n^4)$ (by performing trial and error for every possible number of nontrivial variables $x$'s, each with cubic complexity, until successfully finding a solution $\Lambda(x)$). We present the algorithm as follows while justifying it thereafter.

**Generalized Berlekamp algorithm.**

- Input: $[s_1, s_2, \ldots, s_k]$.
- Initialization: Set $\Lambda^{(0)}(x) = 1$, $B_i^{(0)}(x) = i^{-1}x^{i-1}$, $i = 1, 2, \ldots, p-1$. Set $L_\Lambda^{(0)} = 0$, $L_{B_i}^{(0)} = i - 1$, $i = 1, 2, \ldots, p-1$.
- For $j = 0, 1, 2, \ldots, k-1$, do:
  - Let $j + 1 = qp + r$, where $0 \le r < p$. If $r = 0$, then continue with the next iteration.
  - Compute $\Delta = \sum_{i=0}^{L_\Lambda^{(j)}} s_{j+1-i}\Lambda_i^{(j)}$.
  - Update $\Lambda^{(j+1)}(x) = \Lambda^{(j)}(x) - \Delta x B_r^{(q)}(x)$.
  - If $\Delta \neq 0$ and $L_\Lambda^{(j)} \le L_{B_r}^{(q)}$, then set

$$B_r^{(q+1)}(x) \leftarrow \Delta^{-1}x^{p-1}\Lambda^{(j)}(x), \quad L_{B_r}^{(q+1)} \leftarrow L_\Lambda^{(j)} + p - 1, \quad L_\Lambda^{(j+1)} \leftarrow L_{B_r}^{(q)} + 1.$$

Else set

$$B_r^{(q+1)}(x) \leftarrow x^p B_r^{(q)}(x), \quad L_{B_r}^{(q+1)} \leftarrow L_{B_r}^{(q)} + p, \quad L_\Lambda^{(j+1)} \leftarrow L_\Lambda^{(j)}.$$

- Output: $\Lambda^{(k)}(x)$, $L_\Lambda^{(k)}$.

The following theorem asserts the correctness of the above algorithm. Its proof follows trivially the induction as with the binary case given in [1] and thus is omitted.

**Theorem 3.5.** *At any iteration $j$ of the generalized Berlekamp algorithm, let $j = qp + r$, where $0 \le r < p$.*

(i) *If $j = pq$, i.e., $r = 0$, then $L_\Lambda^{(j)} + \sum_{i=1}^{p-1} L_{B_i}^{(q+1)} = qp(p-1) + \frac{(p-2)(p-1)}{2}$.*

(ii) $\deg(\Lambda^{(j)}(x)) \le L_\Lambda^{(j)}$ *and* $\deg(B_r^{(q)}(x)) \le L_{B_r}^{(q)}$, $r = 1, 2, \ldots, p-1$.

(iii) *The discrepancy $\Delta$ is always zero when $r = 0$.*

(iv) $\Lambda^{(j)}(x)$ *is the minimum-length LFSR (with normalized constant term) to satisfy the linear constraints involved with $s_1, s_2, \ldots, s_j$.*

(v) $xB_r^{(q)}(x)$, $r = 1, 2, \ldots, p-1$, *are the minimum-length LFSR (with zero constant term) to satisfy the linear constraints involved with $r, s_1, s_2, \ldots, s_j$, respectively.*

We proceed to determine the valid roots of $\Lambda(x)$ (herein "valid" means within the predefined field $\mathrm{GF}(p^m)$ or its subset). A straightforward approach is to utilize derivatives to account for the repeated roots of $\Lambda(x)$. In the worst case, the exhaustive field search is employed $\deg(\Lambda(x))$ times, resulting in a complexity of $O(npq)$

(where $q$ denotes the field size, i.e., $q = p^m$), noting that a root may repeat up to $\frac{p-1}{2}$ times. We next present a novel method using Forney's formula which employs an exhaustive field search only once and exhibits a complexity of $O(nq)$.

Let $x_1^{-1}, x_2^{-1}, \ldots, x_f^{-1}$ be the distinct roots of $\Lambda(x)$ determined by an exhaustive field search. First compute the Hamming error-locator polynomial

$$\Lambda^*(x) \stackrel{\triangle}{=} \prod_{i=1}^{f}(x - x_i^{-1}) = \Lambda_0^* + \Lambda_1^* x + \Lambda_2^* x^2 + \cdots + \Lambda_f^* x^f \ . \tag{3.28}$$

Then compute the derivative of the error-locator polynomial

$$[\Lambda^*]'(x) = \Lambda_1^* + 2\Lambda_2^* x + \cdots + f\Lambda_f^* x^{f-1}. \tag{3.29}$$

Next compute the error-evaluator polynomial

$$\Gamma(x) \stackrel{\triangle}{=} \Lambda^*(x) \cdot \sum_{i=1}^{k} s_i x^{i-1} \pmod{x^k}. \tag{3.30}$$

Finally, the error evaluation is performed utilizing Forney's formula as follows (cf. [1, p. 196]):

$$e_i = \frac{\Gamma(x_i^{-1})}{-x_i^{-2} \cdot [\Lambda^*]'(x_i^{-1})}. \tag{3.31}$$

The (unique) solution is successfully produced if the number of valid roots of the error-locator polynomial is equal to its associated length $L_\Lambda$, i.e., $L_\Lambda = \sum_{i=1}^{f} e_i$.

At last, we go through an example to illustrate the proposed decoding/solving procedure.

**Example 3.6.** Let

$$\mathbf{s}_{12} = [\alpha^{61}, \alpha^{13}, \alpha^{23}, \alpha^{63}, \alpha^{39}, \alpha^{39}, \alpha^{43}, \alpha^{37}, \alpha^{69}, \alpha^{40}, \alpha^{75}, \alpha^{29}].$$

The detailed iteration information of the generalized Berlekamp algorithm is shown as follows:

$$\begin{array}{ll}
L_\Lambda^{(1)} = 1, & \Lambda^{(1)}(x) = 1 + \alpha^{21}x^1, \quad \Delta = \alpha^{61}, \\
L_{B_1}^{(1)} = 2, & B_1^{(1)}(x) = \alpha^{19}x^2, \\
L_\Lambda^{(2)} = 2, & \Lambda^{(2)}(x) = 1 + \alpha^{21}x^1 + \alpha^{34}x^2, \quad \Delta = \alpha^{34}, \\
L_{B_2}^{(1)} = 3, & B_2^{(1)}(x) = \alpha^{46}x^2 + \alpha^{67}x^3, \\
L_\Lambda^{(4)} = 3, & \Lambda^{(4)}(x) = 1 + \alpha^{21}x^1 + \alpha^{34}x^2 + \alpha^{52}x^3, \quad \Delta = \alpha^{73}, \\
L_{B_1}^{(2)} = 4, & B_1^{(2)}(x) = \alpha^{7}x^2 + \alpha^{28}x^3 + \alpha^{41}x^4, \\
L_\Lambda^{(5)} = 4, & \Lambda^{(5)}(x) = 1 + \alpha^{21}x^1 + \alpha^{34}x^2 + \alpha^{75}x^3 + \alpha^{18}x^4, \quad \Delta = \alpha^{71}, \\
L_{B_2}^{(2)} = 5, & B_2^{(2)}(x) = \alpha^{9}x^2 + \alpha^{30}x^3 + \alpha^{43}x^4 + \alpha^{61}x^5,
\end{array}$$

$$
\begin{array}{ll}
L_{\Lambda}^{(7)}=5, & \Lambda^{(7)}(x)=1+\alpha^{21}x^1+\alpha^{34}x^2+\alpha^{62}x^3+\alpha^{53}x^4+\alpha^{60}x^5, \quad \Delta=\alpha^{59},\\
L_{B_1}^{(3)}=6, & B_1^{(3)}(x)=\alpha^{21}x^2+\alpha^{42}x^3+\alpha^{55}x^4+\alpha^{16}x^5+\alpha^{39}x^6,\\
L_{\Lambda}^{(8)}=6, & \Lambda^{(8)}(x)=1+\alpha^{21}x^1+\alpha^{34}x^2+\alpha^{20}x^3+\alpha^{12}x^4+\alpha^{36}x^5+\alpha^{16}x^6, \Delta=\alpha^{75},\\
L_{B_2}^{(3)}=7, & B_2^{(3)}(x)=\alpha^{5}x^2+\alpha^{26}x^3+\alpha^{39}x^4+\alpha^{67}x^5+\alpha^{58}x^6+\alpha^{65}x^7,\ \Delta=\alpha^{42}\\
L_{\Lambda}^{(10)}=7, & \Lambda^{(10)}(x)=1+\alpha^{21}x^1+\alpha^{34}x^2+\alpha^{19}x^3+\alpha^{58}x^4+\alpha^{31}x^5+\alpha^{40}x^6+\alpha^{41}x^7,\\
L_{B_1}^{(4)}=8, & B_1^{(4)}(x)=\alpha^{38}x^2+\alpha^{59}x^3+\alpha^{72}x^4+\alpha^{58}x^5+\alpha^{50}x^6+\alpha^{74}x^7+\alpha^{54}x^8,\ \Delta=0\\
L_{\Lambda}^{(11)}=7, & \Lambda^{(11)}(x)=1+\alpha^{21}x^1+\alpha^{34}x^2+\alpha^{19}x^3+\alpha^{58}x^4+\alpha^{31}x^5+\alpha^{40}x^6+\alpha^{41}x^7,\\
L_{B_2}^{(4)}=10, & B_2^{(4)}(x)=\alpha^{5}x^5+\alpha^{26}x^6+\alpha^{39}x^7+\alpha^{67}x^8+\alpha^{58}x^9+\alpha^{65}x^{10},
\end{array}
$$

An exhaustive search over $\mathrm{GF}(3^4)$ identifies 5 distinct roots:

$$\{x_1^{-1}, x_2^{-1}, x_3^{-1}, x_4^{-1}, x_5^{-1}\} = \{\alpha^{66}, \alpha^{42}, \alpha^{39}, \alpha^{36}, \alpha^{31}\}.$$

We next compute the Hamming error-locator polynomial

$$\begin{aligned}\Lambda^*(x) &= (x-\alpha^{66})(x-\alpha^{42})(x-\alpha^{39})(x-\alpha^{36})(x-\alpha^{31})\\ &= \alpha^{14}+\alpha^{77}x^1+\alpha^{71}x^2+\alpha^{28}x^3+\alpha^{66}x^4+x^5\end{aligned}$$

whose derivative is obtained as

$$[\Lambda^*]'(x) = \alpha^{77}-\alpha^{71}x+\alpha^{66}x^3-x^4.$$

We proceed to compute the error-evaluator polynomial

$$\Gamma(x) = \Lambda^*(x)s(x) \pmod{x^{15}} = \alpha^{75}+\alpha^{14}x^1+\alpha^{7}x^2+\alpha^{70}x^3+\alpha^{40}x^4.$$

Applying Forney's formula, we obtain the error magnitude as follows:

$$(\alpha^{14}, 2),\ \ (\alpha^{38}, 1),\ \ (\alpha^{41}, 2),\ \ (\alpha^{44}, 1),\ \ (\alpha^{49}, 1).$$

3.4. **Type-D system of power equations.** In this subsection, we are interested in solving the following special system in $\mathrm{GF}(p^m)$, where $m>1$ and $p$ is a prime greater than 2:

$$\left\{\begin{array}{llllll} x_1 & +x_2 & +x_3 & +\cdots+x_{p-1}+x_p & = 0,\\ x_1^2 & +x_2^2 & +x_3^2 & +\cdots+x_{p-1}^2+x_p^2 & = 0,\\ & & & & \vdots\\ x_1^{p-2}+x_2^{p-2}+x_3^{p-2}+\cdots+x_{p-1}^{p-2}+x_p^{p-2} & & & & = 0. \end{array}\right. \tag{3.32}$$

Note that $p$ unknowns are coupled by only $p-2$ constraints. Clearly, $x_1=x_2=\cdots=x_p$ is a set of trivial solutions of the above system. We next seek nontrivial solutions where at least two $x$'s are distinct.

Newton's identities indicate that

$$\Lambda_i(\mathbf{x}_p)=0, \qquad i=1,2,\ldots,p-2. \tag{3.33}$$

Therefore, $x_1, x_2, \ldots, x_p$ must be all roots of the characteristic equation

$$x^p-ax+b=0, \tag{3.34}$$

where $a, b\in\mathrm{GF}(p^m)$ are arbitrary. Let $x_1$ and $x_2$ be two distinct roots of (3.34). Then we have

$$x_1^p-ax_1 = x_2^p-ax_2.$$

Consequently, we obtain

$$a(x_1-x_2) = x_1^p-x_2^p = (x_1-x_2)^p, \tag{3.35}$$

where the second equality is due to the operation in GF($p^m$) ($p \geq 3$). Since $x_1 \neq x_2$ by assumption, we immediately have

$$a = (x_1 - x_2)^{p-1}. \tag{3.36}$$

Let $a = \alpha^t$, $0 \leq t < p^m - 1$, where $\alpha$ is the primitive element in GF($p^m$). Then (3.36) implies

$$t + i(p^m - 1) = j(p - 1), \tag{3.37}$$

where $i, j$ are some integers. Since $p - 1$ divides $p^m - 1$, we conclude that $p - 1$ divides $t$, say $t = (p - 1)f$. Thus, $x_1 - x_2, x_1 - x_3, \ldots, x_1 - x_p$ are $p - 1$ distinct roots of the equation $x^{p-1} = a$ and take the form

$$x_1 - x_i = \alpha^{f + i\frac{p^m-1}{p-1}}, \qquad i = 2, 3, \ldots, p. \tag{3.38}$$

Conversely, any set of $\{x_i\}_{i=1}^p$ satisfying (3.38) is a valid solution of the system (3.32). Therefore, (3.38) (with arbitrary value $f$ and $x_1$) completely characterizes the nontrivial solutions of Type D system (3.32).

Note that the solutions identified by (3.38) may be repetitious. We proceed to determine the number of distinct nontrivial solutions. We first show that if $x_1 - x_i$, $i = 2, 3, \ldots, p$, are the roots of $x^{p-1} = a$, so is $x_i - x_j$ for any $i \neq j$. Note that (3.35) indicates

$$x_1^p - x_i^p = a(x_1 - x_i),$$
$$x_1^p - x_j^p = a(x_1 - x_j).$$

We immediately have

$$x_i^p - x_j^p = a(x_i - x_j),$$

which further indicates that $x_i - x_j$ is a root of $x^{p-1} = a$. Therefore, each nontrivial solution $\{x_1, x_2, \ldots, x_p\}$ is counted $p$ times, since $x_1$ (as used in (3.38)) can take the values of $x_1, x_2, \ldots, x_p$, respectively, and $x_1, x_2, \ldots, x_p$ are distinct, as indicated by (3.38). Finally, note that $a$ ranges from $\{p - 1, 2(p - 1), 3(p - 1), \ldots, p^m - 1\}$. Thus, there are $\frac{p^m-1}{p-1}p^{m-1}$ distinct nontrivial solutions in Type D system (3.32).

**Example 3.7.** All 12 distinct nontrivial solutions of the above system over GF($3^2$) are listed as follows:

$$\begin{array}{lll} (0, 1, \alpha^4), & (a^1, \alpha^7, \alpha^6), & (a^5, \alpha^2, \alpha^3), \\ (0, \alpha^1, \alpha^5), & (1, \alpha^7, \alpha^2), & (a^4, \alpha^6, \alpha^3), \\ (0, \alpha^6, \alpha^2), & (1, \alpha^1, \alpha^3), & (a^4, \alpha^7, \alpha^5), \\ (0, \alpha^7, \alpha^3), & (1, \alpha^6, \alpha^5), & (a^4, \alpha^1, \alpha^2). \end{array}$$

## 4. Concluding remarks

In this work, we present algebraic solutions for four new types of power equation systems. In the first two types we allow the powers to be a mix of positive and negative terms. The first type involves only positive signs, whereas the second type extends to involve both positive and negative signs. We present algebraic methods to solve the system and furthermore fully characterize the number of nontrivial solutions. In the literature the system of power equations involves purely positive or negative powers. The last two types are defined in finite fields and are otherwise the same as the conventional power equation system. The methodology for solving the third type can be viewed as a generalization of the Berlekamp algorithm which

exhibits quadratic complexity and compares favorably to the straightforward trail-and-error method with quadruple complexity.

It is worth noting that the proposed extensions in the first three types of systems lose the uniqueness property of the original systems. It is imperative to determine the minimal additional constraints such that the resulting systems have up to one valid solution. Any such characterization will significantly boost their applicability.

## References


1. E. R. Berlekamp, *Algebraic Coding Theory,* Rev. Ed., Laguna Hills, CA: Aegean Park Press, 1984.
2. D. Bini and V. Y. Pan, *Polynomial and Matrix Computations, vol. 1: Fundamental Algorithms*, Birkhäuser Boston, Cambridge, MA, 1994. MR1289412 (95k:65003)
3. D. Dobbs and R. Hanks, *A Modern Course on the Theory of Equations*, 2nd edition, Polygonal Publishing House, Washington, NJ, 1992. MR1169295 (93b:12001)
4. G.-L. Feng and K. K. Tzeng, "A generalization of the Berlekamp-Massey algorithm for multisequence shift-register synthesis with applications to decoding cyclic codes," *IEEE Trans. Inform. Theory*, vol. 37, pp. 1274–1287, Sept. 1991. MR1136665 (92m:94012)
5. R. M. Roth and P. H. Siegel, "Lee-metric BCH codes and their application to constrained and partial-response channels," *IEEE Trans. Inform. Theory*, vol. 40, pp. 1083–1095, July 1994. MR1301420 (95h:94047)
6. K. Schiefermayr, "Inverse polynomial images which consists of two Jordan arcs–An algebraic solution," *Journal of Approximation Theory*, vol. 148, pp. 148–157, 2007. MR2362448 (2008k:41008)
7. Y. Wu and C. N. Hadjicostis, "On solving composite power polynomial equations," *Mathematics of Computation*, vol. 74, no. 250, pp. 853–865, 2005. MR2114652 (2005k:65099)
8. ———, "Decoding algorithm and architecture for BCH codes under the Lee metric," *IEEE Trans. Communications*, vol. 56, pp. 2050–2059, Dec. 2008.
9. ———, "Algebraic approaches for fault identification in discrete-event systems," *IEEE Trans. Automatic Control*, vol. 50, pp. 2048–2055, Dec. 2005. MR2186276 (2006f:93075)



Link_A_Media Devices Corporation, 2550 Walsh Avenue, Suite 200, Santa Clara, California 95051

*E-mail address*: yingquan_wu@yahoo.com