Reduced Complexity Interpolation for List Decoding Hermitian Codes

Li Chen, Rolando Carrasco, and Martin Johnston

***Abstract*—List decoding Hermitian codes using the Guruswami-Sudan (GS) algorithm can correct errors beyond half the designed minimum distance. It consists of two processes: interpolation and factorisation. By first defining a Hermitian curve, these processes can be implemented with an iterative polynomial construction algorithm and a recursive coefficient search algorithm respectively. To improve the efficiency of list decoding Hermitian codes, this paper presents two contributions to reduce the interpolation complexity. First, in order to simplify the calculation of a polynomial's zero condition during the iterative interpolation, we propose an algorithm to determine the corresponding coefficients between the pole basis monomials and zero basis functions of a Hermitian curve. Second, we propose a modified complexity reducing interpolation algorithm. This scheme identifies any unnecessary polynomials during iterations and eliminates them to improve the interpolation efficiency. Due to the above complexity reducing modifications, list decoding long Hermitian codes with higher interpolation multiplicity becomes feasible. This paper shows list decoding algorithm can achieve significant coding gain over the conventional unique decoding algorithm.**

***Index Terms*—List decoding, hermitian codes, decoding efficiency.**

I. Introduction

LIST decoding of algebraic-geometric (AG) codes can outperform conventional unique decoding algorithms by correcting errors beyond $\lfloor \frac{d^*-1}{2} \rfloor$, where d^* is the designed minimum distance of the AG code. For a (n, k) AG code with length n and dimension k, $d^* = n - k - g + 1$ where g is the genus of the corresponding algebraic curve. Sudan [1] introduced the first list decoding algorithm for low rate Reed-Solomon (RS) codes, followed by Shokrollahi and Wasserman [2] to list decode low rate AG codes. Guruswami and Sudan [3, 4] later presented a complete version of a list decoding algorithm for all rate RS and AG codes, called the Guruswami-Sudan (GS) algorithm. For a (n, k) AG code, the GS algorithm can correct up to

$$\tau_{GS} = n - \lfloor \sqrt{n(n-d^*)} \rfloor - 1 \qquad (1)$$

errors. The algorithm consists of two processes: interpolation, which builds an interpolated polynomial based on the received information, and factorisation, which finds the transmitted message information based on the interpolated polynomial. They can be implemented with an iterative polynomial construction algorithm [5-8] and a recursive coefficient search algorithm [9-12] respectively.

RS codes are widely used in industry, such as communication and storage systems. Compared with RS codes defined over the same Galois field (GF), AG codes have better error-correction capability because they have longer code word lengths resulting in a large designed minimum distance [13, 14]. Therefore, AG codes could be considered as the successor of RS codes. Among them, Hermitian codes are one of the best performing and well developed AG codes. Johnston and Carrasco [15, 16] evaluated Hermitian code performance by employing the Sakata algorithm [17, 18] with majority voting [19] and inverse discrete Fourier transforms. This is a unique decoding algorithm with the half distance error-correction bound. To decode Hermitian codes beyond this bound, Hoholdt and Nielsen [5, 6] presented a mathematical framework of list decoding Hermitian codes with the GS algorithm. After an extensive study of [5, 6], the authors published the first simulation results of list decoding Hermitian codes in [20], which showed significant coding gains can be achieved over a unique decoding algorithm [17-19]. However, list decoding Hermitian codes with the GS algorithm remains complex and limits the application of the GS algorithm to longer codes. According to the complexity analysis in [8], the GS algorithm's high complexity is mainly caused by the iterative interpolation, in which a set of polynomials is tested for different zero conditions and modified interactively. For Hermitian codes, to define the zero condition of a polynomial, we need to transfer it into a polynomial written with respect to the zero basis functions of a Hermitian curve, which is not very efficient for implementation. However, the zero condition of a polynomial can also be defined without this transfer based on knowledge of the corresponding coefficients between the pole basis monomials and zero basis functions of a Hermitian curve. Inspired by this, we propose an efficient algorithm to determine these coefficients before interpolation so that they can be applied afterwards. In order to improve the list decoding efficiency for RS codes, the authors [8] proposed a complexity reducing scheme to identify any unnecessary polynomials in the set and eliminate them during the iterative interpolation process. This scheme is also valid for list decoding Hermitian codes. This paper proposes a complexity reducing interpolation algorithm applying the scheme in [8]. The factorisation process of AG codes can be implemented by applying the recursive coefficient search algorithm [11]. A more generalised factorisation algorithm which can be applied to both RS and AG codes is later presented by the

Manuscript received June 14, 2007; revised October 31, 2007 and March 7, 2008; accepted April 22, 2008. The associate editor coordinating the review of this paper and approving it for publication was V. K. Bhargava.

The authors are with the School of Electrical, Electronic and Computer Engineering, Newcastle University, Newcastle-upon-Tyne, United Kingdom, NE1 7RU (e-mail: {li.chen, r.carrasco, martin.johnston}@ncl.ac.uk).

Digital Object Identifier 10.1109/T-WC.2008.070615

authors in [12]. By improving the efficiency of interpolation, list decoding longer Hermitian codes with higher multiplicity becomes feasible. This paper evaluates the performance of 4 different Hermitian codes defined in GF(16) and GF(64) over the additive white Gaussian noise (AWGN) and Rayleigh fading channels.

The structure of the paper is organised as follows: Section II presents prerequisite knowledge of Hermitian curves and Hermitian codes; Section III briefly describes the GS decoding of Hermitian codes, in which the zero condition of a polynomial and the decoding parameters are defined; Section IV proposes an efficient algorithm to determine the corresponding coefficients between the pole basis monomials and zero basis functions of a Hermitian curve; Section V presents the reduced complexity interpolation algorithm; simulation results evaluating the performance of 4 different Hermitian codes are presented in Section VI and conclusions are given in Section VII.

II. Prerequisite Knowledge

A Hermitian curve defined over GF(q), where q is the size of Galois field, is given as:

$$H_w(x,y,z) = x^{w+1} + y^w z + y z^w \tag{2}$$

where $w = \sqrt{q}$ and has a genus $g = \frac{w(w-1)}{2}$ [14]. For simplicity, this paper assumes GF(q) is an extension field of GF(2). $H_w(x,y,z)$ is a projective curve with three affine components $H_w(1,y,z)$, $H_w(x,1,z)$ and $H_w(x,y,1)$. By selecting one affine component $H_w(x,y,1)$, there are $n = w^3$ affine points on it, which are of the form $p_i = (x_i, y_i, 1)(0 \le i \le n-1)$, and a point of infinity $p_\infty = (0,1,0)$. Rational functions on the curve H_w must have a pole only at p_∞, and can be generally written as [14]:

$$\frac{x^\lambda y^\delta}{z^{\lambda+\delta}} \ (0 \le \lambda \le w, \delta \ge 0) \tag{3}$$

where $\lambda, \delta \in N$ and N denotes the set of nonnegative integers in this paper. Based on $H_w(x,y,z) = 0$, $\frac{x}{z} = \frac{y^w + y z^{w-1}}{x^w}$, $\frac{y}{z} = \frac{y^{w+1} + y^2 z^{w-1}}{x^{w+1}}$. Therefore, the pole orders [21] of variable $\frac{x}{z}$ and $\frac{y}{z}$ at p_∞ are:

$$v_{p_\infty}\left(\left(\frac{x}{z}\right)^{-1}\right) = w, v_{p_\infty}\left(\left(\frac{y}{z}\right)^{-1}\right) = w+1. \tag{4}$$

Pole basis L_w of curve H_w contains a set of rational functions with $z = 1$, coefficients 1, and increasing pole orders. As $z = 1$, these rational functions can be reduced to bivariate monomials, defined as [5]:

$$L_w = \{\phi_a \mid v_{p_\infty}(\phi_a^{-1}) < v_{p_\infty}(\phi_{a+1}^{-1}), a \in N\} \tag{5}$$

where pole basis monomial ϕ_a and its pole order can be generally written as:

$$\phi_a = x^\lambda y^\delta, v_{p_\infty}((x^\lambda y^\delta)^{-1}) = w\lambda + (w+1)\delta, \quad (0 \le \lambda \le w, \delta \ge 0). \tag{6}$$

Also, affine points can be distinctively denoted by their x, y coordinates as: $p_i = (x_i, y_i)$. Nonnegative integers can be grouped into nongaps which are the pole orders of monomials in L_w, and gaps otherwise. With respect to every affine point p_i, there exists a zero basis Z_{w,p_i} which contains a set of rational functions $\psi_{p_i,\alpha}$ with increasing zero orders [21] at $p_i(v_{p_i})$, defined as [5]:

$$Z_{w,p_i} = \{\psi_{p_i,\alpha} \mid v_{p_i}(\psi_{p_i,\alpha}) < v_{p_i}(\psi_{p_i,\alpha+1}), \alpha \in N\}. \tag{7}$$

$\psi_{p_i,\alpha}$ has a zero order α at affine point p_i and it can be generally written as [6]:

$$\psi_{p_i,\alpha} = \psi_{p_i,\lambda+(w+1)\delta} = (x-x_i)^\lambda [(y-y_i) - x_i^w (x-x_i)]^\delta, \quad (0 \le \lambda \le w, \delta \ge 0). \tag{8}$$

The relationship between pole basis monomial ϕ_a and zero basis function $\psi_{p_i,\alpha}$ can be written as [6]:

$$\phi_a = \sum_{\alpha \in N} \gamma_{a,p_i,\alpha} \psi_{p_i,\alpha} \tag{9}$$

where $\gamma_{a,p_i,\alpha} \in$ GF(q) are the corresponding coefficients.

The construction of a (n,k) Hermitian code can be described as evaluating the n affine points of H_w over the message polynomial f:

$$\overline{c}(n,k) = (c_0, c_1, \cdots, c_{n-1}) = (f(p_0), f(p_1), \cdots, f(p_{n-1})) \tag{10}$$

where $c_0, c_1, \cdots, c_{n-1} \in$ GF(q) and f could be written as:

$$f(x,y) = f_0\phi_0 + f_1\phi_1 + \cdots + f_{k-1}\phi_{k-1} \tag{11}$$

and $f_0, f_1, \ldots, f_{k-1} \in$ GF(q) are message symbols and $\phi_0, \phi_1, \ldots, \phi_{k-1}$ are the first k monomials in L_w.

To decode a (n,k) Hermitian code with the GS algorithm, the pole order of variable z is defined as $v_{p_\infty}(z^{-1}) = v_{p_\infty}(\phi_{k+1}^{-1})$. Then, the weighted degree of any trivariate monomial $\phi_a z^b$ can be defined as:

$$deg_w(\phi_a z^b) = v_{p_\infty}(\phi_a^{-1}) + b v_{p_\infty}(z^{-1}) \tag{12}$$

and a lexicographic order (ord) can be defined to arrange the monomials $\phi_a z^b$:

$$\phi_{a_1} z^{b_1} < \phi_{a_2} z^{b_2}$$

if $deg_w(\phi_{a_1} z^{b_1}) < deg_w(\phi_{a_2} z^{b_2})$, or $deg_w(\phi_{a_1} z^{b_1}) = deg_w(\phi_{a_2} z^{b_2})$ and $b_1 < b_2$ [5]. $F_q[x,y,z]$ is the ring of polynomials defined over the set of pole basis functions of the Hermitian curve L_w, which can be generally written as: $f(x,y,z) = \sum_{a,b \in N} f_{ab}\phi_a(x,y)z^b$, where $f_{ab} \in$ GF(q) and $\phi_a \in L_w$. Subsequently, $F_q^u[x,y]$ is a subset of $F_q[x,y,z]$ with z degree equal to zero and $v_{p_\infty}(\phi_a^{-1}) \le u$. If we denote $u_z = v_{p_\infty}(z^{-1})$, the message polynomial in (11) is a polynomial in $F_q^{u_z}[x,y]$. The following definition is given for polynomials defined in $F_q[x,y,z]$:

Definition 1: If $\phi_{a'} z^{b'}$ is the maximal monomial in polynomial $f \in F_q[x,y,z]$:

$$\phi_{a'} z^{b'} = max\{\phi_a z^b \mid f_{ab} \ne 0\}$$

then $\phi_{a'} z^{b'}$ is called the leading monomial of f and its coefficient f_{ab} is called the leading coefficient of f, denoted as:

$$LM(f) = \phi_{a'} z^{b'}, \text{ and } LC(f) = f_{a'b'}$$

The weighted degree of f ($deg_w(f)$) and leading order of f ($lod(f)$) are defined as:

$$deg_w(f) = deg_w(\phi_{a'} z^{b'}), \text{ and } lod(f) = ord(\phi_{a'} z^{b'})$$

Based on the above definitions, for any two polynomials f and $h \in F_q[x, y, z]$, $f < h$ if $lod(f) < lod(h)$.

III. GS Decoding of Hermitian Codes

The GS algorithm consists of two processes: interpolation and factorisation. Given a received word $R = (r_0, r_1, \cdots, r_{n-1}) \in$ GF(q), n interpolated units can be formed by combining each received symbol with its respective affine point used in encoding as: $(p_0, r_0), (p_1, r_1), \cdots, (p_{n-1}, r_{n-1})$. Interpolation generates the minimal polynomial $Q \in F_q[x, y, z]$ which has a zero of multiplicity of at least m $(m > 0)$ over the n interpolated units. In general,

$$Q = \sum_{a,b \in N} Q_{ab} \phi_a z^b \tag{13}$$

where $Q_{ab} \in$GF(q). With regard to the interpolated unit (p_i, r_i), if polynomial Q can also be written with respect to the zero basis functions in Z_{w,p_i} as [6]:

$$Q = \sum_{\alpha,\beta \in N} Q_{\alpha\beta}^{(p_i,r_i)} \psi_{p_i,\alpha} (z - r_i)^\beta \tag{14}$$

where $Q_{\alpha\beta}^{(p_i,r_i)} \in$ GF(q). Based on equation (8), clearly, $Q(p_i, r_i) = 0$. Furthermore, if $Q_{\alpha\beta}^{(p_i,r_i)} = 0$ for $\alpha + \beta < m$, polynomial Q has a zero of multiplicity of at least m at unit (p_i, r_i) [5, 6]. As $z^b = (z - r_i + r_i)^b = \sum_{\beta \le b} \binom{b}{\beta} r_i^{b-\beta} (z - r_i)^\beta$ and $\phi_a = \sum_{\alpha \in N} \gamma_{a,p_i,\alpha} \psi_{p_i,\alpha}$, substituting them into (13) gives:

$$\begin{aligned} Q &= \sum_{a,b \in N} Q_{ab} \Big(\sum_{\alpha \in N} \gamma_{a,p_i,\alpha} \psi_{p_i,\alpha} \Big) \Big(\sum_{\beta \le b} \binom{b}{\beta} r_i^{b-\beta} (z - r_i)^\beta \Big) \\ &= \sum_{\alpha,\beta \in N} \Big(\sum_{a,b \ge \beta} Q_{ab} \binom{b}{\beta} \gamma_{a,p_i,\alpha} r_i^{b-\beta} \Big) \psi_{p_i,\alpha} (z - r_i)^\beta . \end{aligned} \tag{15}$$

Therefore, coefficients $Q_{\alpha\beta}^{(p_i,r_i)}$ of (14) can be written as:

$$Q_{\alpha\beta}^{(p_i,r_i)} = \sum_{a,b \ge \beta} Q_{ab} \binom{b}{\beta} \gamma_{a,p_i,\alpha} r_i^{b-\beta}. \tag{16}$$

(16) defines the zero condition constraints of the coefficients Q_{ab} of polynomial Q, so that Q has a zero of multiplicity of at least m over unit (p_i, r_i). If we denote the constraints in (16) to the coefficients of polynomial Q as $D_{\alpha\beta}^{(p_i,r_i)}(Q)$, such that:

$$D_{\alpha\beta}^{(p_i,r_i)}(Q) = Q_{\alpha\beta}^{(p_i,r_i)} = \sum_{a,b \ge \beta} Q_{ab} \binom{b}{\beta} \gamma_{a,p_i,\alpha} r_i^{b-\beta}. \tag{17}$$

Then after interpolating all the affine points, the interpolated polynomial Q shall satisfy:

$$\begin{aligned} Q = \min_{lod(Q)} \{ & Q \in F_q[x, y, z] \mid D_{\alpha\beta}^{(p_i,r_i)}(Q) = 0, for\ i = 0, \\ & 1, \ldots, n-1 \wedge \alpha + \beta < m\ (\alpha, \beta \in N) \}. \end{aligned} \tag{18}$$

As there are $\binom{m+1}{2}$ permutations of (α, β) for $\alpha + \beta < m$, there are in total:

$$C = n \binom{m+1}{2} \tag{19}$$

zero condition constraints that the coefficients Q_{ab} of polynomial Q need to satisfy. C also represents the number of iterations in the interpolation algorithm [5, 6], in which each iteration imposes a zero condition constraint on Q_{ab}. The weighted degree upper bound of interpolated polynomial Q is defined as [5, 6]:

$$max\{deg_w Q\} = l_m v_{p_\infty}(z^{-1}) + t_m \tag{20}$$

where l_m is the maximal number of output candidates from factorisation, defined as:

$$l_m = max\{u \mid \binom{u}{2} v_{p_\infty}(z^{-1}) - (u-1)g \le C\} - 1 \tag{21}$$

and parameter t_m is defined as:

$$\begin{aligned} t_m = max\{u \mid & (l_m + 1)u - \Gamma(u) + \binom{l_m+1}{2} v_{p_\infty}(z^{-1}) \\ & - l_m g \le C\} \end{aligned} \tag{22}$$

where $u \in N$ and $\Gamma(u)$ denotes the number of gaps that are less than or equal to the nonnegative integer u [5].

According to Theorem 3 of [5], if there exists a polynomial $h \in F_q^{u_z}[x, y]$ such that

$$m\Lambda(h, R) > deg_w Q \tag{23}$$

where $\Lambda(h, R) = |\{i \mid h(p_i) = r_i, i = 0, 1, \ldots, n-1\}|$ represents the number of affine points that satisfy $h(p_i) = r_i$, then h is the z root of Q, i.e. $Q(x, y, h) = 0$, or equivalently $(z - h) \mid Q(x, y, z)$. Factorisation finds the z roots of the interpolated polynomial Q, among which the message polynomial of (11) is included [10-12, 20]. Therefore, the GS algorithm's error-correction capability τ_m is:

$$\tau_m = n - \Lambda(h, R) = n - \left\lfloor \frac{deg_w Q}{m} \right\rfloor - 1. \tag{24}$$

As the upper bound of $deg_w Q$ is defined by (20), then:

$$\tau_m \ge n - \left\lfloor \frac{l_m v_{p_\infty}(z^{-1}) + t_m}{m} \right\rfloor - 1. \tag{25}$$

IV. Determining The Corresponding Coefficients

Based on (17), the corresponding coefficients $\gamma_{a,p_i,\alpha}$ are critical for defining the zero condition of a polynomial in $F_q[x, y, z]$. Without the knowledge of them, we must transfer a general polynomial (13) into (14) and find the coefficients $Q_{\alpha\beta}^{(p_i,r_i)}$, which is not efficient during the iterative interpolation. In fact, the corresponding coefficients $\gamma_{a,p_i,\alpha}$ can be determined independently of the received word. Therefore, if they can be determined beforehand and applied during the iterations, the interpolation efficiency can be greatly improved. This section proposes an algorithm to determine them.

The problem we intend to solve can be simply stated as: given an affine point $p_i = (x_i, y_i)$ of the curve H_w and a pole basis monomial ϕ_a, determine the corresponding coefficients $\gamma_{a,p_i,\alpha}$ so that ϕ_a can be written as a sum of the zero basis functions $\psi_{p_i,\alpha}$: $\phi_a = \sum_{\alpha \in N} \gamma_{a,p_i,\alpha} \psi_{p_i,\alpha}$. For any two pole basis monomials ϕ_{a_1} and ϕ_{a_2} in L_w, $\phi_{a_1} \phi_{a_2} = \sum_{a \in N} \phi_a$ and the zero basis function $\psi_{p_i,\alpha}$ in (8) can be written as a sum of pole basis monomials ϕ_a as [6]:

$$\psi_{p_i,\alpha} = \sum_{a \in N} \zeta_a \phi_a \tag{26}$$

where coefficients $\zeta_a \in$ GF(q). Based on (8), we partition $\psi_{p_i,\alpha}$ as:

$$\psi_{p_i,\alpha} = \psi^A_{p_i,\alpha} \cdot \psi^B_{p_i,\alpha} \tag{27}$$

where $\psi^A_{p_i,\alpha} = (x - x_i)^\lambda$ and $\psi^B_{p_i,\alpha} = [(y - y_i) - x_i^w(x - x_i)]^\delta = [y - x_i^w x - (y_i - x_i^{w+1})]^\delta$. It is easy to recognise that $\psi^A_{p_i,\alpha}$ has leading monomial $LM(\psi^A_{p_i,\alpha}) = x^\lambda$ and leading coefficient $LC(\psi^A_{p_i,\alpha}) = 1$. Since $v_{p_\infty}(y^{-1}) > v_{p_\infty}(x^{-1})$, $\psi^B_{p_i,\alpha}$ has leading monomial $LM(\psi^B_{p_i,\alpha}) = y^\delta$ and leading coefficient $LC(\psi^B_{p_i,\alpha}) = 1$. Based on (27), $\psi_{p_i,\alpha}$ has leading monomial $LM(\psi^A_{p_i,\alpha}) \cdot LM(\psi^B_{p_i,\alpha}) = x^\lambda y^\delta$ and leading coefficient $LC(\psi^A_{p_i,\alpha}) \cdot LC(\psi^B_{p_i,\alpha}) = 1$. As $0 \le \lambda \le w$ and $\delta \ge 0$, the set of leading monomials of the zero basis functions in Z_{w,p_i} contains all the monomials defined in pole basis L_w. Summarising the above analysis, we propose *Corollary 1*.

Corollary 1: If ϕ_L is the leading monomial of the zero basis function $\psi_{p_i,\alpha}$ as $LM(\psi_{p_i,\alpha}) = \phi_L$, the leading coefficient of $\psi_{p_i,\alpha}$ equals 1 and (26) can be written as:

$$\psi_{p_i,\alpha} = \sum_{a \in N, a < L} \zeta_a \phi_a + \phi_L. \tag{28}$$

The set of leading monomials of the zero basis functions in Z_{w,p_i} contains all the monomials in L_w:

$$\{LM(\psi_{p_i,\alpha}) = \phi_L, \psi_{p_i,\alpha} \in Z_{w,p_i}\} \subseteq L_w. \tag{29}$$

Following on, by identifying the second largest pole basis monomial ϕ_{L-1} with coefficient $\zeta_{L-1} \in$ GF(q) in $\psi_{p_i,\alpha}$, (28) can also be written as:

$$\psi_{p_i,\alpha} = \sum_{a \in N, a < L-1} \zeta_a \phi_a + \zeta_{L-1}\phi_{L-1} + \phi_L. \tag{30}$$

Now it is sufficient to propose our efficient algorithm to determine the corresponding coefficients $\gamma_{a,p_i,\alpha}$.

Algorithm A: Determining the corresponding coefficients $\gamma_{a,p_i,\alpha}$ between a pole basis monomial ϕ_a and zero basis functions $\psi_{p_i,\alpha}$.

Step 1: Initialise all corresponding coefficients $\gamma_{a,p_i,\alpha} = 0$;
Step 2: Find the zero basis function $\psi_{p_i,\alpha}$ with $LM(\psi_{p_i,\alpha})$ =
ϕ_a, and let $\gamma_{a,p_i,\alpha} = 1$;
Step 3: Initialise function $\hat{\psi} = \psi_{p_i,\alpha}$;
Step 4: While $(\hat{\psi} \neq \phi_a)$ {
Step 5: Find the second largest pole basis monomial ψ_{L-1} with coefficient ζ_{L-1} in $\hat{\psi}$;
Step 6: In Z_{w,p_i}, find a zero basis function $\psi_{p_i,\alpha}$ whose leading monomial $LM(\psi_{p_i,\alpha}) = \phi_{L-1}$, and let the corresponding coefficient $\gamma_{a,p_i,\alpha} = \zeta_{L-1}$;
Step 7: Update $\hat{\psi} = \hat{\psi} + \gamma_{a,p_i,\alpha}\psi_{p_i,\alpha}$;
}

Proof: Notice that functions $\psi_{p_i,\alpha}$ with $LM(\psi_{p_i,\alpha}) > \phi_a$ will not contribute to the sum calculation of (9) and their corresponding coefficients $\gamma_{a,p_i,\alpha} = 0$. The zero basis function $\psi_{p_i,\alpha}$ found at step 2 has leading monomial $\phi_L = \phi_a$. Based on (30), it can be written as:

$$\psi_{p_i,\alpha} = \sum_{a' \in N, a' < L-1} \zeta_{a'}\phi_{a'} + \zeta_{L-1}\phi_{L-1} + \phi_a. \tag{31}$$

(31) indicates the corresponding coefficient between ϕ_a and $\psi_{p_i,\alpha}$ is: $\gamma_{a,p_i,\alpha} = 1$. Polynomial $\hat{\psi}$ initialised by step 3 is an accumulated polynomial resulting in ϕ_a. While $\hat{\psi} \neq \phi_a$, in (31), the second largest monomial ϕ_{L-1} with coefficient ζ_{L-1} is identified by step 5. Then, find another zero basis function $\psi_{p_i,\alpha}$ in Z_{w,p_i} where $LM(\psi_{p_i,\alpha}) = \phi_{L-1}$. According to *Corollary 1*, this zero basis function always exists and it can be written as: $\psi_{p_i,\alpha} = \sum_{a' \in N, a' < L-1} \zeta_{a'}\phi_{a'} + \phi_{L-1}$. At step 6, the corresponding coefficient between monomial ϕ_a and the found zero basis function $\psi_{p_i,\alpha}$ can be determined as: $\gamma_{a,p_i,\alpha} = \zeta_{L-1}$. As a result, the accumulated calculation of step 7 can be written as:

$$\begin{aligned}
\hat{\psi} &= \sum_{a' \in N, a' < L-1} \zeta_{a'}\phi_{a'} + \zeta_{L-1}\phi_{L-1} + \phi_a + \gamma_{a,p_i,\alpha}\psi_{p_i,\alpha} \\
&= \sum_{a' \in N, a' < L-1} \zeta_{a'}\phi_{a'} + \zeta_{L-1}\phi_{L-1} + \phi_a + \\
&\quad \sum_{a' \in N, a' < L-1} \zeta_{L-1}\zeta_{a'}\phi_{a'} + \zeta_{L-1}\phi_{L-1} \\
&= \sum_{a' \in N, a' < L-1} \zeta_{a'}\phi_{a'} + \phi_a.
\end{aligned} \tag{32}$$

Therefore in the new accumulated $\hat{\psi}$, $\zeta_{L-1}\phi_{L-1}$ is eliminated while the leading monomial ϕ_a is preserved. If the updated $\hat{\psi} \neq \phi_a$, its second largest monomial ϕ_{L-1} is again eliminated while ϕ_a is always preserved as a leading monomial by the same process. The algorithm terminates after all monomials that are smaller than ϕ_a have been eliminated and results in $\hat{\psi} = \phi_a$. This process is equivalent to the sum calculation of (9). Here a worked example is presented to illustrate *Algorithm A*.

Example: Given $p_i = (\sigma^2, \sigma^2)$ (σ is a primitive element in GF(4) satisfying $\sigma^2 + \sigma + 1 = 0$) is an affine point on curve H_2 and a pole basis (L_2) monomial $\phi_5 = y^2$, determine the corresponding coefficients $\gamma_{5,p_i,\alpha}$ so that ϕ_5 can be written as $\phi_5 = \sum_{\alpha \in N} \gamma_{5,p_i,\alpha}\psi_{p_i,\alpha}$.

Based on (8), the first 8 zero basis functions in Z_{2,p_i} can be listed as:

$\psi_{p_i,0} = (x - \sigma^2)^0 = 1$;
$\psi_{p_i,1} = (x - \sigma^2)^1 = \sigma^2 + x$;
$\psi_{p_i,2} = (x - \sigma^2)^2 = \sigma + x^2$;
$\psi_{p_i,3} = (y - \sigma^2) - \sigma(x - \sigma^2) = \sigma + \sigma x + y$;
$\psi_{p_i,4} = (x - \sigma^2)[(y - \sigma^2) - \sigma(x - \sigma^2)] = 1 + \sigma^2 x + \sigma^2 y + \sigma x^2 + xy$;
$\psi_{p_i,5} = (x - \sigma^2)^2[(y - \sigma^2) - \sigma(x - \sigma^2)] = \sigma^2 + \sigma^2 x + \sigma x^2 + \sigma y^2 + x^2 y$;
$\psi_{p_i,6} = [(y - \sigma^2) - \sigma(x - \sigma^2)]^2 = \sigma^2 + \sigma^2 x^2 + y^2$;
$\psi_{p_i,7} = (x - \sigma^2)[(y - \sigma^2) - \sigma(x - \sigma^2)]^2 = \sigma + \sigma^2 x + \sigma^2 y + \sigma x^2 + xy^2$.

Initialise all $\psi_{5,p_i,\alpha} = 0$. In Z_{2,p_i}, as $LM(\psi_{p_i,6}) = \phi_5$, we let $\gamma_{5,p_i,6} = 1$ and initialise the accumulated polynomial $\hat{\psi} = \psi_{p_i,6} = \sigma^2 + \sigma^2 x^2 + y^2$.

As $\hat{\psi} \neq \phi_5$, its second largest monomial $\phi_{L-1} = x^2$ with coefficient $\zeta_{L-1} = \sigma^2$ is identified. Among the zero basis functions in Z_{2,p_i}, we find $\psi_{p_i,2}$ with $LM(\psi_{p_i,2}) = \phi_{L-1} = x^2$ and let $\gamma_{5,p_i,2} = \zeta_{L-1} = \sigma^2$. Update $\hat{\psi} = \hat{\psi} + \gamma_{5,p_i,2}\psi_{p_i,2} = \sigma + y^2$.

As $\hat{\psi} \neq \phi_5$, again its second largest monomial $\phi_{L-1} = 1$ with coefficient $\zeta_{L-1} = \sigma$ are identified. Among the zero basis functions in Z_{2,p_i}, we find $\psi_{p_i,0}$ with $LM(\psi_{p_i,0}) = \phi_{L-1} =$

1, and let $\gamma_{5,p_i,0} = \zeta_{L-1} = \sigma$. Update $\hat{\psi} = \hat{\psi} + \gamma_{5,p_i,0}\psi_{p_i,0} = y^2$.

Now, $\hat{\psi} = \phi_5$, we can stop the algorithm and output $\gamma_{5,p_i,0} = \sigma$, $\gamma_{5,p_i,2} = \sigma^2$ and $\gamma_{5,p_i,6} = 1$. The rest of the corresponding coefficients $\gamma_{5,p_i,\alpha} = 0$ $(\alpha \neq 0, 2, 6)$.

Before interpolation, the monomials ϕ_a that exist in the interpolated polynomial Q are unknown. However, the weighted degree upper bound of polynomial Q is defined by (20), from which the largest pole basis monomial ϕ_{max} that might exist in Q can be predicted by $v_{p_\infty}(\phi_{max}^{-1}) = max\{deg_w Q\}$. Based on an interpolation multiplicity m, with parameter $\alpha < m$, the corresponding coefficients that might be used in interpolation are $\gamma_{0,p_i,\alpha} \sim \gamma_{max,p_i,\alpha}$. Therefore *Algorithm A* can be used to determine all the corresponding coefficients $\gamma_{0,p_i,\alpha} \sim \gamma_{max,p_i,\alpha}$ and only $\gamma_{0,p_i,\alpha} \sim \gamma_{max,p_i,\alpha}$ $(\alpha < m)$ are stored for interpolation in order to minimise the memory requirement. For example, to list decode the (8, 4, 4) Hermitian code with multiplicity $m = 2$, $max\{deg_w Q\} = 13$. Therefore, the largest pole basis monomial that might exist in Q is $\phi_{max} = \phi_{12} = x^2y^3$ and *Algorithm A* can be applied to calculate all the corresponding coefficients $\gamma_{0,p_i,\alpha} \sim \gamma_{12,p_i,\alpha}$ and $\gamma_{0,p_i,\alpha} \sim \gamma_{12,p_i,\alpha}$ $(\alpha < 2)$ are stored.

V. Complexity Reducing Interpolation

Interpolation determines the polynomial Q defined by (18). This can be implemented using an iterative polynomial construction algorithm [5-8]. First, a set of polynomials is initialised. During the iterations, they are tested by different zero condition constraints and modified interactively. As mentioned in Section III, there are in total C iterations as given in (19), after which the minimal polynomial in the set is chosen as the interpolated polynomial Q. According to the iterative process analysis [8], the interpolated polynomial Q has leading order $lod(Q) \leq C$. This indicates that those polynomials with leading order over C will not be the chosen candidates. Also, if there is a polynomial in the set with leading order over C during the iterations, the chosen polynomial Q has not been modified with this polynomial, otherwise $lod(Q) > C$ [8]. Therefore, those polynomials with leading order greater than C can be eliminated from the set during iterations in order to reduce unnecessary computations.

If $f \in F_q[x, y, z]$ has leading monomial $LM(f) = \phi_{a'} z^{b'}$, polynomials in $F_q[x, y, z]$ can be partitioned into the following classes according to the z degree of their leading monomial and the pole order $v_{p_\infty}(\phi_{a'}^{-1})$ of $\phi_{a'}$ as:

$$V_{\lambda+w\delta} = \{f \in F_q[x, y, z] \mid b' = \delta \wedge v_{p_\infty}(\phi_{a'}^{-1}) = uw + \lambda, LM(f) = \phi_{a'} z^{b'}, (\delta, u, \lambda) \in N, \lambda < w\} \quad (33)$$

such that $F_q[x, y, z] = \bigcup_{\lambda,\delta \in N, \lambda < w} V_{\lambda+w\delta}$. According to Section III, the factorisation outputs are the z roots of Q. Therefore, the z degree of Q is less than or equal to the maximal length of the output list l_m of (21) and Q is a polynomial chosen from the following classes:

$$V_j = V_{\lambda+w\delta}, \quad (0 \leq \lambda < w, 0 \leq \delta \leq l_m). \quad (34)$$

At the beginning of the iterative process, a set of polynomials is initialised to represent each of the polynomial classes defined by (34) as:

$$G = \{Q_j = Q_{\lambda+w\delta} = y^\lambda z^\delta, Q_j \in V_j\}. \quad (35)$$

During the iterations, each polynomial Q_j in the set G is the minimal polynomial within its class V_j that satisfies all the tested zero conditions. At the beginning of each iteration, the polynomial set G is modified by:

$$G = \{Q_j \mid lod(Q_j) \leq C\} \quad (36)$$

in order to eliminate those polynomials with leading order over C. Then the remaining polynomials in G are tested by the zero condition defined in (17) as:

$$\Delta_j = D_{\alpha\beta}^{(p_i,r_i)}(Q_j). \quad (37)$$

The determined corresponding coefficients $\gamma_{a,p_i,\alpha}$ are applied for this calculation. Those polynomials with $\Delta_j = 0$ satisfy the zero condition and do not need to be modified. However, those polynomials with $\Delta_j \neq 0$ need to be modified. Among them, find the index of the minimal polynomial as j' and record the minimal polynomial as Q':

$$j' = index(\min_{lod(Q_j)} \{Q_j \mid \Delta_j \neq 0\}). \quad (38)$$

$$Q' = Q_{j'}. \quad (39)$$

For $Q_{j'}$, it is modified as:

$$Q_{j'} = (x - x_i)Q' \quad (40)$$

where x_i is the x coordinate of affine point p_i in the current interpolated unit (p_i, r_i). The modified Q_j satisfies $D_{\alpha\beta}^{(p_i,r_i)}(Q_{j'}) = 0$ because $D_{\alpha\beta}^{(p_i,r_i)}[(x - x_i)Q'] = D_{\alpha\beta}^{(p_i,r_i)}(xQ') - x_i D_{\alpha\beta}^{(p_i,r_i)}(Q') = x_i\Delta_{j'} - x_i\Delta_{j'} = 0$. The rest of the polynomials with $\Delta_j \neq 0$ are modified as:

$$Q_j = \Delta_{j'} Q_j - \Delta_j Q'. \quad (41)$$

The modified Q_j satisfies $D_{\alpha\beta}^{(p_i,r_i)}(Q_j) = 0$ because $D_{\alpha\beta}^{(p_i,r_i)}[\Delta_{j'} Q_j - \Delta_j Q'] = \Delta_{j'} D_{\alpha\beta}^{(p_i,r_i)}(Q_j) - \Delta_j D_{\alpha\beta}^{(p_i,r_i)}(Q') = \Delta_{j'}\Delta_j - \Delta_j\Delta_{j'} = 0$. After C iterations, the minimal polynomial in the set G is chosen as the interpolated polynomial Q:

$$Q = \min_{lod(Q_j)} \{Q_j \mid Q_j \in G\}. \quad (42)$$

From the above description, it can be seen that by applying the complexity reducing scheme in (36), the zero condition calculation in (37) and modifications in (40) and (41) for those polynomials Q_j with $lod(Q_j) > C$ can be avoided, and therefore the interpolation efficiency can be improved. According to [8], this complexity reducing scheme is error dependent and can reduce complexity more significantly in low error weight situations. This is because the modification scheme in (36) takes action in earlier iteration steps for low error weight situations, and therefore computations can be reduced. Fig. 1 shows the interpolation (with different multiplicity m) complexity reduction by applying the scheme in (36) for decoding the (64, 19, 40) Hermitian code. It is shown that complexity can be reduced significantly in low error weight situations, especially when $m = 1$, and complexity can be reduced by up to 48.83%. However, in high error

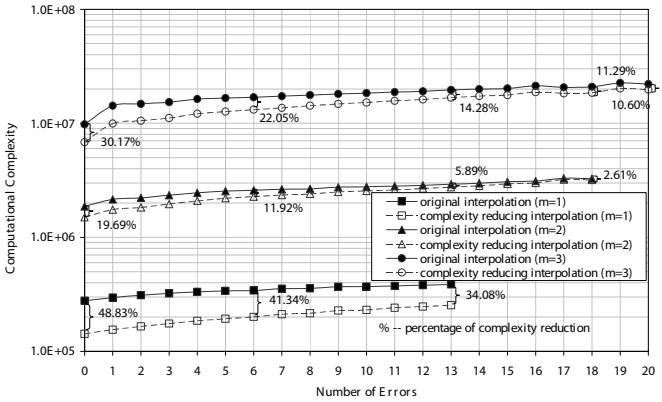


Fig. 1. Complexity analysis for the interpolation of GS decoding Hermitian code (64, 19, 40).

weight situations, complexity reduction is not as significant. Based on Fig. 1, it can also be observed that the complexity reduction also depends on the interpolation multiplicity m. When $m = 1$, complexity reduction is significant; when $m = 2$, complexity reduction is marginal.

Summarising Section IV and V, the reduced complexity interpolation process for GS decoding Hermitian codes can be stated as:

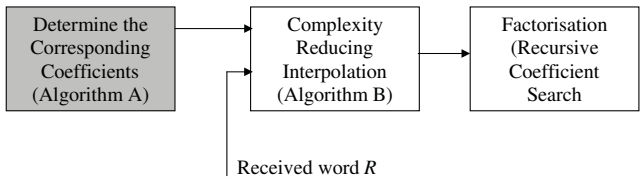
Initial computation: Apply *Algorithm A* to determine all the necessary corresponding coefficients $\gamma_{a,p_i,\alpha}$ and store them to be used by the iterative polynomial construction algorithm (*Algorithm B*);

Algorithm B: Iterative Polynomial Construction.

Initialisation: Initialise the set of polynomials G by (35);
Step 1: For each interpolated unit $(p_i, r_i)(i = 0, 1, \cdots, n-1)$ {
Step 2: For each pair of the zero condition parameters $(\alpha, \beta)(\alpha + \beta < m)$ {
Step 3: Modify polynomial set G by (36);
Step 4: Test the zero condition Δ_j of each polynomial in G by (37);
Step 5: For polynomials Q_j with $\Delta_j \neq 0$ {
Step 6: Denote the index of the minimal polynomial as j' by (38) and record it as Q' by (39);
Step 7: If $j = j'$, Q_j is modified by (40);
Step 8: If $j \neq j'$, Q_j is modified by (41);
}}}
At the end of the iterations, the minimal polynomial Q is chosen from the set G as (42).

VI. Performance Evaluation

Employing the above efficiency improved interpolation, list decoding of longer Hermitian codes with different multiplicity is feasible. The authors have implemented the GS decoding of Hermitian codes [3, 5] using C programming language, in which the factorisation process is implemented by the recursive coefficient search algorithm [11, 12]. The evaluating list decoder structure is presented in Fig. 2. Figs. 3 and 4 present the performance of (64, 19, 40) and (64, 39, 20) Hermitian codes, while Figs. 5 and 6 present the performance of (512, 153, 332) and (512, 289, 196) Hermitian codes. Simulations are run over AWGN and Rayleigh fading channels



Note: ▭ indicates the pre-calculation step of the efficient list decoding algorithm.

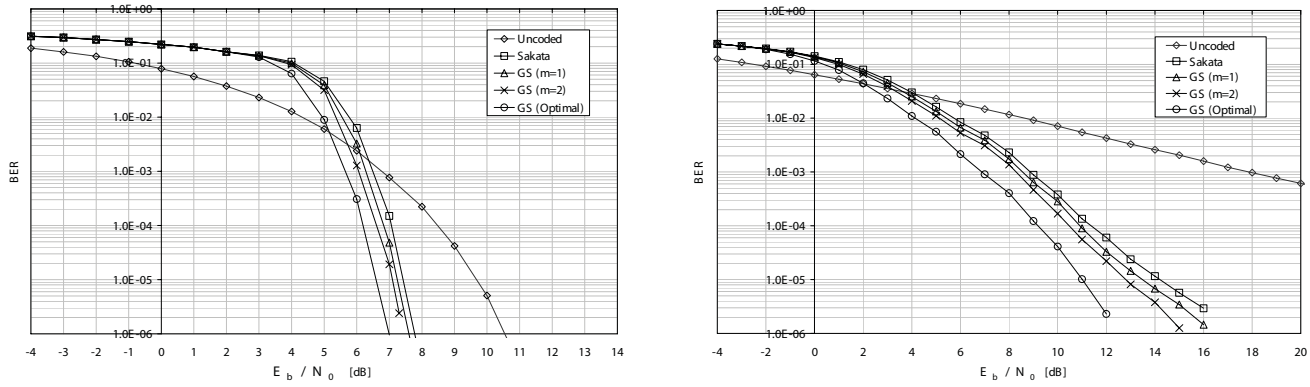
Fig. 2. Efficiency improved list decoder structure for Hermitian codes.

using the QPSK modulation. The Rayleigh fading channel is frequency nonselective with Doppler frequency 126.67 Hz and date rate of 30 kb/s. Over the fading channel, a block interleaver with size $100 \times n$ is used, where n is the length of the code. Their performances are evaluated by measuring their coding gains (dB) over the unique decoding algorithm [15-19] at a bit error rate (BER) of 10^{-5}.

According to the interpolation description in Section V, for interpolation with multiplicity m, there are $w(l_m + 1)$ polynomials being initialised that take part in C iterations. Even though some of them will be eliminated during the iteration by the scheme in (36), a high value of m will still lead to infeasibility for implementation as both the iteration number and polynomial set size grow with m exponentially. Therefore, we have achieved some feasible results from the GS algorithm $(m = 1, 2)$. Table I[1]presents our simulation parameters for these 4 Hermitian codes. Before interpolation, *Algorithm A* is applied to determine the corresponding coefficients $\gamma_{0,p_i,\alpha} \sim \gamma_{max,p_i,\alpha}$ and only $\gamma_{0,p_i,\alpha} \sim \gamma_{max,p_i,\alpha}$ $(\alpha = 0, 1)$ are stored in order to minimise the memory requirement. From Table I, it can be observed that to achieve the optimal result from the GS algorithm remains too complex for implementation. For example, to optimally decode (64, 19, 40) Hermitian code, there are 116 polynomials taking part in 9792 iterations in interpolation. But if assuming the GS algorithm is able to correct τ_{GS} errors and the transmitted code word $\bar{c}$ is known by the decoder, the theoretical optimal performance of the GS algorithm can also be evaluated without employing the interpolation and factorisation processes. This is achieved by measuring the Hamming distance between the received word R and the transmitted code word $\bar{c}$. If it is not greater than τ_{GS}, decoding is claimed to be successful; otherwise, decoding is failure. Figs. 3 to 6 show that the GS algorithm approaches its optimal result with increasing interpolation multiplicity m. Among the performance evaluations, it is worth highlighting Fig. 4 which shows GS decoding the (64, 39, 20) Hermitian code with multiplicity $m = 2$ is close to the theoretical optimal result.

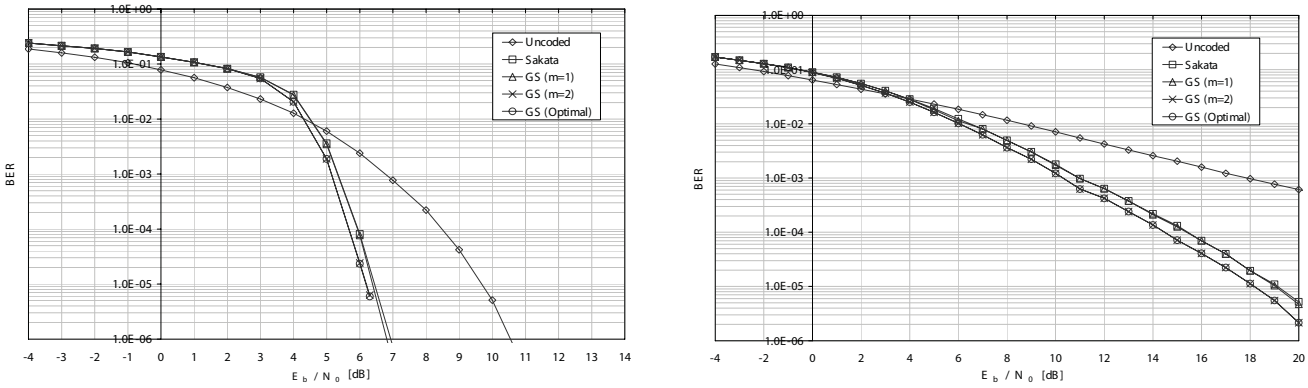
Table II analyses the simulation results shown in Figs. 3 to 6. During our simulations, the average number of errors $\bar{\tau_m}$ that the GS algorithm is able to correct in order to achieve the corresponding performance is measured. Based on Table II, it can be observed that the GS algorithmŠs coding gains grow with interpolation multiplicity m and they are especially significant over the Rayleigh fading channel.

[1]In Table I, $max\{deg_w Q\}$ represents the weighted degree upper bound of the interpolated polynomial; ϕ_{max} represents the maximal pole basis monomial that might exist in the interpolated polynomial Q.



(a) over AWGN channel (b) over Rayleigh fading channel

Fig. 3. List decoding performance of Hermitian code (64, 19, 40).



(a) over AWGN channel (b) over Rayleigh fading channel

Fig. 4. List decoding performance of Hermitian code (64, 39, 20).

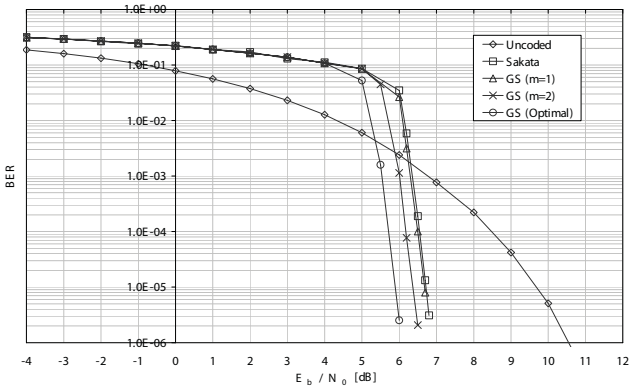
TABLE I
LIST DECODING PARAMETERS

Hermitian codes	Interpolation multiplicity	C	l_m	$w(l_m+1)$	$max\{deg_w Q\}$	ϕ_{max}
(64, 19, 40)	$m=1$	64	2	12	50	$\phi_{44}=y^{10}$
	$m=2$	192	3	16	90	$\phi_{84}=y^{18}$
	optimal($m=17$)	9792	28	116	–	–
(64, 39, 20)	$m=1$	64	1	8	60	$\phi_{54}=y^{12}$
	$m=2$	192	2	12	114	$\phi_{108}=xy^{22}$
	optimal($m=11$)	4224	13	56	–	–
(512, 153, 332)	$m=1$	512	2	24	373	$\phi_{345}=x^5y^{37}$
	$m=2$	1536	3	32	682	$\phi_{654}=x^2y^{74}$
	optimal($m=213$)	11668992	359	2880	–	–
(512, 289, 196)	$m=1$	512	1	16	442	$\phi_{414}=x^8y^{42}$
	$m=2$	1536	2	24	856	$\phi_{828}=x^8y^{88}$
	optimal($m=93$)	2237952	118	952	–	–

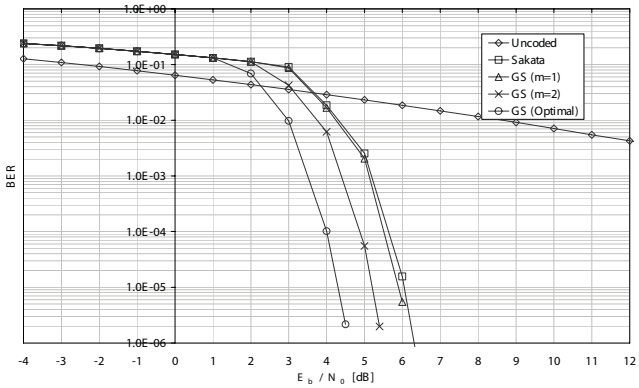
For example, GS decoding (64, 19, 40) Hermitian code with multiplicity $m = 2$ can achieve 1.42 dB coding gain over the Rayleigh fading channel. The GS algorithm can achieve more significant coding gains for low rate codes, but this is at the expense of higher decoding complexity. For example, comparing the (512, 153, 332) and (512, 289, 196) Hermitian codes, more significant coding gains can be achieved for the former. However, according to Table I, GS decoding (512, 153, 332) Hermitian code has higher decoding complexity because there are more polynomials being initialised that take part in the iterative interpolation. The same result can also be found by comparing the (64, 19, 40) and (64, 39, 20) Hermitian codes.

VII. CONCLUSION

This paper presents two contributions to reduce interpolation complexity so as to improve the efficiency of list decoding Hermitian codes. We first propose an efficient algorithm to determine the corresponding coefficients between the pole basis monomials and zero basis functions of a Hermitian curve. The coefficients are stored to be applied during the iterative interpolation in order to simplify the zero condition



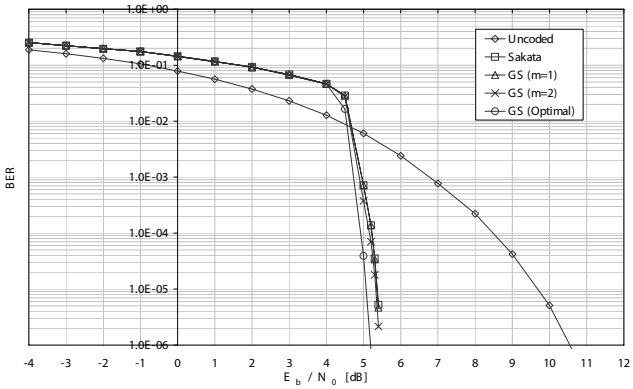
(a) over AWGN channel



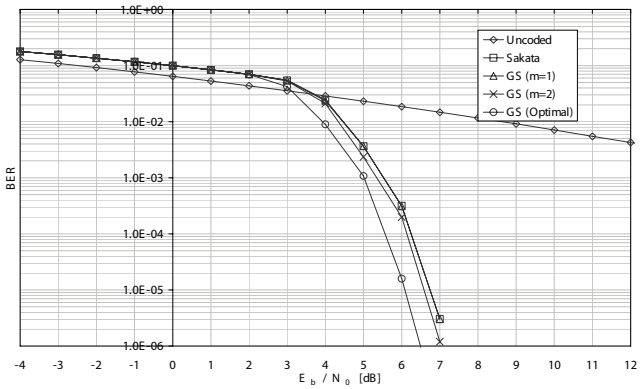
(b) over Rayleigh fading channel

Fig. 5. List decoding performance of Hermitian code (512, 153, 332)

.



(a) over AWGN channel



(b) over Rayleigh fading channel

Fig. 6. List decoding performance of Hermitian code (512, 289, 196).

TABLE II
ANALYSIS OF EVALUATION RESULTS

Hermitian codes	Interpolation multiplicity	τ_m	Coding gains (dB)	
			AWGN	Rayleigh fading
(64, 19, 40)	$m = 1$	20	0.17	0.71
	$m = 2$	21	0.33	1.42
	optimal $(m = 17)$	$\tau_{GS} = 24$	0.91	3.30
(64, 39, 20)	$m = 1$	9	0.10	0.01
	$m = 2$	10	0.30	0.94
	optimal $(m = 11)$	$\tau_{GS} = 10$	0.30	0.94
(512, 153, 332)	$m = 1$	167	0.05	0.16
	$m = 2$	184	0.40	0.88
	optimal $(m = 213)$	$\tau_{GS} = 208$	0.88	1.84
(512, 289, 196)	$m = 1$	97	0.01	0.01
	$m = 2$	99	0.08	0.16
	optimal $(m = 93)$	$\tau_{GS} = 109$	0.32	0.72

calculation of a polynomial. We then propose a complexity reducing interpolation algorithm by applying a scheme developed by the authors which eliminates any unnecessary polynomials during iterations. It is shown that this scheme can improve the interpolation efficiency significantly. Given that complexity can be reduced by the proposed methods, we have evaluated list decoding performance of longer Hermitian codes with higher interpolation multiplicity. Our results show that the GS algorithm can achieve significant coding gains over the unique decoding algorithm. The GS algorithmŠs coding gains increase with interpolation multiplicity and it is more significant for low rate codes. However, this performance is at the expense of higher decoding complexity.

ACKNOWLEDGMENT

The authors would like to thank Dr. R. Refslund Nielsen for his helpful discussion on the list decoding of Hermitian codes.

REFERENCES

[1] M. Sudan, "Decoding of Reed-Solomon codes beyond the error-correction bound," *J. Complexity*, vol. 13, pp. 180-193, 1997.

[2] M. Shokrollahi and H. Wasserman, "List decoding of algebraic-geometric codes," *IEEE Trans. Inform. Theory*, vol. 45, pp. 432-437, 1999.

[3] V. Guruswami and M. Sudan, "Improved decoding of Reed-Solomon and algebraic-geometric codes," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1757-1767, 1999.

[4] V. Guruswami, *List Decoding of Error-Correcting Codes.* Berlin/Heidelberg: Springer-Verlag, 2004.

[5] T. Hoholdt and R. R. Nielsen, "Decoding Hermitian codes with Sudan's algorithm," in *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (Lecture Notes in Computer Science)*, vol. 1719, H. I. N. Fossorier, S. Lin, and A. Pole, ed. Berlin: Springer-Verlag, 1999, pp. 260-270.

[6] R. R. Nielsen, *List Decoding of Linear Block Codes*. Lyngby, Denmark: Tech. Univ. Denmark, 2001.

[7] R. J. McEliece, "The Guruswami-Sudan decoding algorithm for Reed-Solomon codes," California Institute. Tech, Pasadena, California, IPN Progress Rep 42-153, 2003.

[8] L. Chen, R. A. Carrasco, and E. G. Chester, "Performance of Reed-Solomon codes using the Guruswami-Sudan algorithm with improved interpolation efficiency," *IET Commun*, vol. 1, pp. 241-250, 2007.

[9] R. Roth and G. Ruckenstein, "Efficient decoding of Reed-Solomon codes beyond half the minimum distance," *IEEE Trans. Inform. Theory*, vol. 46, pp. 246-257, 2000.

[10] X. W. Wu and P. Siegel, "Efficient root-finding algorithm with application to list decoding of algebraic-geometric codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 2579-2587, 2001.

[11] X. W. Wu, "An algorithm for finding the roots of the polynomials over order domains," in *Proc. ISIT 2002*, Lausanne, Switzerland, 2002.

[12] L. Chen, R. A. Carrasco, M. Johnston, and E. G. Chester, "Efficient factorisation algorithm for list decoding algebraic-geometric and Reed-Solomon codes," in *Proc. ICC 2007*, Glasgow, UK, 2007.

[13] V. D. Goppa, "Codes on algebraic curves," *Soviet Math*, vol. dok. 24, pp. 75-91, 1981.

[14] I. Blake, C. Heegard, T. Høholdt, and V. Wei, "Algebraic-geometric codes," *IEEE Trans. Inform. Theory*, vol. 44, pp. 2596-2618, 1998.

[15] M. Johnston, R. A. Carrasco, and B. L. Burrows, "Design of new algebraic-geometric codes over fading channels," *IEE Electron. Lett.*, 2004.

[16] M. Johnston and R. A. Carrasco, "Construction and performance of algebraic-geometric codes over AWGN and fading channels," *IEE Proc. Commun.*, vol. 152, pp. 713-722, 2005.

[17] S. Sakata, J. Justesen, Y. Madelung, H. E. Jensen, and T. Høholdt, "Fast decoding of algebraic-geometric codes up to the designed minimum distance," *IEEE Trans. Inform. Theory*, vol. 41, pp. 1672-1677, 1995.

[18] S. Sakata, "Finding a minimal set of linear recurring relations capable of generating a given finite two-dimensional array," *J. Symbol. Comput.*, vol. 5, pp. 321-337, 1998.

[19] G. L. Feng and T. R. N. Rao, "Decoding algebraic-geometric codes up to the designed minimum distance," *IEEE Trans. Inform. Theory*, vol. 39, pp. 37-46, 1993.

[20] L. Chen, R. A. Carrasco, and M. Johnston, "List decoding performance of algebraic geometric codes," *IET Electron. Lett.*, vol. 42, 2006.

[21] O. Pretzel, *Codes and Algebraic Curves*. Oxford: Clarendon Press, 1998.

Li Chen received his BSc degree in applied physics from Jinan University, P. R. China, 2003, the MSc degree in communication and signal processing from Newcastle University, UK, 2004, and was awarded the PhD degree in mobile communications from Newcastle University in 2008. He is currently a research associate with the School of electrical, electronic and computer engineering, Newcastle University. His research interests include algebraic geometric coding schemes, list decoding systems, other non-binary coding schemes and their applications in the cooperative wireless network.

Rolando Carrasco obtained his Bachelors BEng(Hons) degree in Electrical Engineering from Santiago University, Chile in 1969; his PhD degree in Signal Processing from the University of Newcastle-upon-Tyne in 1980. He was awarded the IEE Heaviside Premium in 1982 for his work in multiprocessor systems. Between 1982 and 1984 he was employed by Alfred Peters Limited, Sheffield (now Meditech) and carried out research and development in signal processing associated with cochlear stimulation and response. He had been with Staffordshire University since 1984 and joined Newcastle University in 2004 as Professor of Mobile Communications. His principle research interests are digital signal processing algorithm for data communication systems, mobile and network communication systems, speech recognition and processing. Professor Carrasco has over two hundred scientific publications, 5 chapters in telecommunications reference texts and a patent to his name.

Martin Johnston received the BSc(Hons) degree in physics with electronics from the University of Birmingham in 1999, the MSc degree in Electronic Engineering from Staffordshire University in 2001 and was awarded the PhD degree in the design and construction of algebraic-geometric codes from Newcastle University in 2006. His research interests include the design of non-binary error-correcting codes for wireless and data storage applications and low complexity decoding algorithms. He is currently a Research Associate at Newcastle University investigating the design of new error-correction coding schemes for high density magnetic storage channels.