Hermite Interpolation With Error Correction: Fields of Zero or Large Characteristic and Large Error Rate

Erich L. Kaltofen
Dept. of Math., NCSU
Raleigh, NC, USA
Dept. of Comp. Sci., Duke University
Durham, NC, USA
kaltofen@ncsu.edu

Clément Pernet
Laboratoire Jean Kuntzmann, Univ.
Grenoble Alpes, CNRS
Grenoble, France
clement.pernet@univ-grenoble-alpes.fr

Zhi-Hong Yang
Coll. of Math. and Statistics
Shenzhen University
Shenzhen, China
zhihongyang2020@outlook.com

ABSTRACT

Multiplicity code decoders are based on Hermite polynomial interpolation with error correction. In order to have a unique Hermite interpolant one assumes that the field of scalars has characteristic 0 or $\geq \ell + 1$, where ℓ is the maximum order of the derivatives in the list of values of the polynomial and its derivatives which are interpolated. For scalar fields of characteristic $\ell + 1$, the minimum number of values for interpolating a polynomial of degree $\leq D$ is $D + 1 + 2E(\ell + 1)$ when $\leq E$ of the values are erroneous. Here we give an error-correcting Hermite interpolation algorithm that requires fewer values, that is, that can tolerate more errors, assuming that the characteristic of the scalar field is either 0 or $\geq D + 1$. Our algorithm requires $(\ell + 1)D + 1 - (\ell + 1)\ell/2 + 2E$ values.

As an example, we consider $\ell = 2$. If the error ratio (number of errors)/(number of evaluations) ≤ 0.16, our new algorithm requires $\lceil (4 + 7/17)\, D - (1 + 8/17) \rceil$ values, while multiplicity decoding requires $25D + 25$ values. If the error ratio is ≤ 0.2, our algorithm requires $5D - 2$ evaluations over fields of characteristic 0 or $\geq D + 1$, while multiplicity decoding for an error ratio 0.2 over fields of characteristic 3 is not possible for $D \geq 3$.

Our algorithm is based on Reed-Solomon interpolation without multiplicities, which becomes possible for Hermite interpolation because of the high redundancy necessary for error-correction.

CCS CONCEPTS

• **Mathematics of computing** → **Interpolation**; *Computations in finite fields*; • **Theory of computation** → **Error-correcting codes**; • **Computing methodologies** → *Algebraic algorithms.*

KEYWORDS

algebraic error correction codes; multiplicity error correction codes; Reed-Solomon error correction codes; list decoding;

ACM Reference Format:
Erich L. Kaltofen, Clément Pernet, and Zhi-Hong Yang. 2021. Hermite Interpolation With Error Correction: Fields of Zero or Large Characteristic and Large Error Rate. In *Proceedings of the 2021 International Symposium on Symbolic and Algebraic Computation (ISSAC '21), July 18–23, 2021, Virtual Event, Russian Federation.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3452143.3465525

ISSAC '21, July 18–23, 2021, Virtual Event, Russian Federation

ACM ISBN 978-1-4503-8382-0/21/07…$15.00
https://doi.org/10.1145/3452143.3465525

1 INTRODUCTION

The number of errors which can be corrected when interpolating a function from values can be dependent on arithmetic properties of the scalar field. In [2] we have demonstrated that when performing sparse interpolation in standard basis with powers of the variable, for real values one can correct a much higher error rate than for complex number values. Here we show that for Hermite polynomial interpolation with error correction, that is, multiplicity code decoding [5, 7], for scalar fields of 0 or large characteristic at higher error rates fewer values are required to compute a unique interpolant. Our new polynomial-time interpolation algorithm is specific to polynomials and is based on iterated Reed-Solomon decoding, unlike the earlier algorithms, which are based on an error-locator polynomial with multiple roots (see also Remark 4.1 below).

For $\ell_1 \geq \cdots \geq \ell_n \geq 0$ we interpolate $f \in \mathsf{K}[x]$ of degree bounded as $\deg(f) \leq D$ from values at n distinct arguments ξ_i with $1 \leq i \leq n$, namely, $\hat{a}_{i,j} \in \mathsf{K}$ where $f^{(j)}(\xi_i) = \hat{a}_{i,j}$ with $1 \leq i \leq n$ and $0 \leq j \leq \ell_i$ at evaluations that are not erroneous. With $f^{(j)}$ we denote the j-th derivative of f in x. We impose that the characteristic of K is either 0 or $\geq \ell_1 + 1$ and that $D \geq \ell_1$, for otherwise $f^{(\ell_1)} = 0$ and all non-erroneous $\hat{a}_{i,\ell_1} = 0$. The set of error locations is

$$I = \{(i, j) \mid f^{(j)}(\xi_i) \neq \hat{a}_{i,j}, 1 \leq i \leq n, 0 \leq j \leq \ell_i\}. \tag{1}$$

As input our algorithms have a degree bound D, $[\xi_i]_{1 \leq i \leq n}$, vectors $\widehat{A}_{i,*} = [\hat{a}_{i,0}, \ldots, \hat{a}_{i,\ell_i}]$ for $1 \leq i \leq n$ and upper bounds for the number of errors: either $E \geq |\{\, i \mid \exists j \colon (i, j) \in I \,\}|$ or $E_{\text{tot}} \geq |I|$. Algorithm 4.1 in [3] can interpolate a unique f, or prove that none exists, when the number of values satisfies

$$L \stackrel{\text{def}}{=} \sum_{i=1}^{n} (\ell_i + 1) \geq N \stackrel{\text{def}}{=} D + 1 + 2 \sum_{i=1}^{E} (\ell_i + 1) = D + 1 + 2E + 2 \sum_{i=1}^{E} \ell_i. \tag{2}$$

We show in [3] that if $L < N$ in (2), one can have multiple interpolants: see Example 1 for $\ell_1 = 1$, $\mathsf{K} = \mathbb{R}$ and $N = D + 4E_{\text{tot}}$, and Example 3 (or Section 2 below) for $\mathsf{K} = \mathbb{Z}_{\ell_1+1}$ and $N = D + 2E_{\text{tot}}(\ell_1 + 1)$. However, in Remark 1 in [3] we observe that for a large bound of errors E_{tot} and for fields K of characteristic 0 or $\geq D+1$ the count (2) is sub-optimal for $\ell_1 = 1$. In fact, if $E_{\text{tot}} \geq D/2$ one can interpolate from $2D + 2E_{\text{tot}}$ values.

Our Algorithm 4.1 below interpolates from $N_{\text{zero}} = (\ell_1+1)D + 1 - \binom{\ell_1+1}{2} + 2E_{\text{tot}}$ values, provided that $n \geq 2E_{\text{tot}}+1$ and that the characteristic of K is either 0 or $\geq D+1$. For an error bound $E_{\text{tot}} > \frac{1}{2}(D - \frac{1}{2}(\ell_1+1))$ our new algorithm requires fewer values than (2) with $E = E_{\text{tot}}$ and $\ell_E = \ell_1$. The reason for the improvement is based on the assumption that D-th derivatives of polynomials of degree D cannot be zero. The general multiplicity code decoder [3, Algorithm 4.1] divides by $\ell_1!$, and its count (2) is optimal for characteristic $\geq \ell_1+1$. Our new algorithm generalizes the Reed-Solomon decoder to the Hermite problem. Our algorithm performs Hermite interpolation by repeated recursive polynomial interpolation at higher and higher derivatives and may divide by $D!$. The reduction to standard polynomial interpolation is possible because one has a high error rate and sufficient redundancy in the values, and is special to the error correcting Hermite problem. In Example 3.1 we perform the calculation for the example in the abstract. In Example 3.2 below we show that the new count is minimal at least for $\ell_1 = 2$.

2 SMALL CHARACTERISTIC WORST CASE

We slightly modify Example 3 in [3, Section 4]. If the field of scalars K has finite characteristic $\geq \ell_1+1$, our count (2) is optimal for higher derivatives. Let $n = 2E+\nu$, for $\nu \geq 1$, and let $\ell_1 = \cdots = \ell_{2E+\nu} = p-1$ for a prime number p which is the characteristic of the field of scalars K, whose cardinality is $|\mathsf{K}| \geq 2E+\nu+1$, so that there exist $n+1$ distinct elements ξ_i in K. Let $f(x) = (x-\xi_1)^p \cdots (x-\xi_\nu)^p$ and let $D = \deg(f) = \nu p$. Then $f(\xi_1) = \cdots = f(\xi_\nu) = 0$ and $f^{(j)}(\xi_i) = 0$ for all $1 \leq i \leq n$ and $1 \leq j \leq \ell_i$. Therefore f and the zero polynomial interpolate all $(2E+\nu)p - 2E$ zero values, and E errors cannot be unambiguously corrected from $N = (2E+\nu)p = D + 2E(\ell_1+1)$ values. If one adds an $(N+1)$-st value $f(\xi_{n+1})$ then $N+1 = D+1+2E(\ell_1+1)$ and Algorithm 4.1 in [3] and the algorithm described in Remark 4.1 interpolate a unique polynomial with $\leq E$ erroneous values.

3 ZERO AND LARGE CHARACTERISTIC

Here we generalize Remark 1 in [3] to arbitrarily high derivatives ℓ_1. We assume that the characteristic of K is either 0 or $\geq D+1$. Let

$$E_{\text{tot}} \geq |\{\, \hat{a}_{i,j} \mid f^{(j)}(\xi_i) \neq \hat{a}_{i,j}, 1 \leq i \leq n, 0 \leq j \leq \ell_i \,\}| \qquad (3)$$

be a bound on the total number of errors. For an unambiguous constant coefficient one must assume that $n \geq 2E_{\text{tot}}+1$.

Because $f^{(D+j)} = 0$ for $j \geq 1$ we assume that $\ell_1 \leq D$. We can prove that

$$\begin{aligned} L = (\ell_1+1) + \cdots + (\ell_n+1) &\geq \\ N_{\text{zero}} &\stackrel{\text{def}}{=} (\ell_1+1)D + 1 - \tbinom{\ell_1+1}{2} + 2E_{\text{tot}} \\ &= D + (D-1) + \cdots + (D-\ell_1) + 1 + 2E_{\text{tot}} \end{aligned} \qquad (4)$$

is a decodable number of evaluations when $2E_{\text{tot}} > D - (\ell_1+1)/2$, in which case

$$N_{\text{zero}} < N_{\text{tot}} \stackrel{\text{def}}{=} D + 1 + 2E_{\text{tot}} + 2E_{\text{tot}}\ell_1 \quad (\text{for } \ell_1 \geq 1). \qquad (5)$$

For $D = 0$ we have $\ell_1 = 0$ and $N_{\text{tot}} = N_{\text{zero}} = n = 2E_{\text{tot}}+1$. Note that for $2E_{\text{tot}} \leq D - (\ell_1+1)/2$ we have $N_{\text{tot}} \leq N_{\text{zero}}$, and f is decodable from N_{tot} evaluations by Algorithm 4.1 in [3]. However, if $N_{\text{zero}} < N_{\text{tot}}$, then a new algorithm is required, because Algorithm 4.1 does not account for the restriction that the characteristic of K is 0 or $\geq D+1$, and, by Section 2 above, the count N_{tot} is required for characteristic $\geq \ell_1+1$.

The assumption $n \geq 2E_{\text{tot}}+1$ is explicit here. For the count (2) in [3] it is implied (see (13) below). In fact, if $n = 2E_{\text{tot}}$ is arbitrary large, one cannot recover an interpolant f from $2(\ell_1+1)E_{\text{tot}}$ evaluations: f' can interpolate $\hat{a}_{i,j}$ without errors for all $1 \leq i \leq 2E_{\text{tot}}$ and $1 \leq j \leq \ell_1$, but f and $f+1$ can both have E_{tot} errors at the $\hat{a}_{i,0}$. Similarly, if $n = E_{\text{tot}}$ is arbitrarily large, the list of interpolants of $(\ell_1+1)E_{\text{tot}}$ values can be all $f+c$ for all $c \in \mathsf{K}$. However, we show in Lemma 3.1 below that for $n \leq 2E_{\text{tot}}$ the count (4) is sufficient to obtain a unique $f^{(\ell_1)}$ for all interpolants f with $\leq E_{\text{tot}}$ errors.

Remark 3.1. Our algorithms use several counts for the number of values and the number of errors. The upper bound E on the number of errors in (2) counts the number of ξ_i where $\hat{a}_{i,j}$ is erroneous for at least one j. Therefore, a "burst" of errors at an index i counts as one error. In the worst case, however, E can be the total number of errors, whose upper bound we denote by E_{tot}. The count N_{tot} in (5) is the minimum number of values the multiplicity decoders require when $E = E_{\text{tot}}$ is the number of errors, one for each index i, and $\ell_1 = \cdots = \ell_E$. The number of values that are input is L (2), which can be more than the minimum count if the inequalities in our estimates for the recursive calls are not sharp. The count N_{zero} in (4) is our new minimum count for characteristic "zero" and large characteristic. □

We first show that the zero polynomial is the only interpolant of degree $\leq D$ with evaluations that yield 0 at any of $N_{\text{zero}} - 2E_{\text{tot}}$ of the evaluations. Therefore an interpolant f of degree $\leq D$ is unique, if it exists with $\leq E_{\text{tot}}$ errors in the $\hat{a}_{i,j}$. For otherwise, if there were polynomial interpolants, denoted by $f^{[1]}$ and $f^{[2]}$, both with $\leq E_{\text{tot}}$ errors, then $f^{[1]} - f^{[2]}$ would be zero at at least $N_{\text{zero}} - 2E_{\text{tot}}$ locations.

Lemma 3.1. *Let $\ell_1 \geq \ell_2 \geq \cdots \geq \ell_n \geq 0$ and let (i_λ, j_λ) for $\lambda = 1, \ldots, \widehat{E}$ with $1 \leq i_\lambda \leq n$ and $0 \leq j_\lambda \leq \ell_{i_\lambda}$ be $\widehat{E}$ distinct arbitrary locations, and let $I = \{(i_\lambda, j_\lambda)\}_{1 \leq \lambda \leq \widehat{E}}$. Assume that*

$$(\ell_1+1) + \cdots + (\ell_n+1) - \widehat{E} \geq (\ell_1+1)D + 1 - \tbinom{\ell_1+1}{2}, \qquad (6)$$

where $\binom{1}{2} = 0$. Let $g \in \mathsf{K}[x]$ with $\deg(g) \leq D$ such that $g^{(j)}(\xi_i) = 0$ for $1 \leq i \leq n$ and $0 \leq j \leq \ell_i$ and $(i,j) \notin I$.

1. *Then $g^{(\ell_1)}(x) = 0$.*
2. *If $n \geq \widehat{E}+1$, then $g(x) = 0$.*

Proof. We prove the Lemma by induction on ℓ_1. For $\ell_1 = 0$ we have $g(\xi_i) = 0$ at $\geq D+1$ distinct $i \neq i_\lambda$, so $g = 0$. Now let $\ell_1 \geq 1$ and let $\widehat{E}_0 \leq n$ be the number of locations in I with $j_\lambda = 0$. We shall distinguish 2 cases. If $n - \widehat{E}_0 \geq D+1$ then g is zero at $\geq D+1$ distinct arguments ξ_i and therefore is again equal 0 and the Lemma is proven. The second case is that $\widehat{E}_0 \geq n-D$. For that, we consider the zero values of $g', \ldots, g^{(\ell)}$. We have

$$\begin{aligned} \ell_1 + \cdots + \ell_n - (\widehat{E} - \widehat{E}_0) &\geq (\ell_1+1)D + 1 - \tbinom{\ell_1+1}{2} - n + (n-D) \\ &= \ell_1(D-1) + 1 - \tbinom{\ell_1}{2} \end{aligned}$$

such values. Therefore the induction hypothesis applies to those values and we conclude from Fact 1 for $\ell_1 - 1$ and g' that $g^{(\ell_1)} = 0$,

which is Fact 1.

We finally prove Fact 2, assuming that $n \geq \widehat{E}+1$. By assumption on the characteristic of K and $g^{(\ell_1)} = 0$, we must have $\deg(g) \leq \ell_1 - 1$. We shall count the zero values of $g, g', \ldots, g^{(\ell_1-1)}$. Let

$$m_\mu \stackrel{\text{def}}{=} |\{i \mid 1 \leq i \leq n, \ell_i \geq \mu\}| \tag{7}$$

be the number of μ-th derivative values. We have $n = m_0 \geq m_1 \geq \cdots \geq m_{\ell_1}$ and

$$L = m_0 + \cdots + m_{\ell_1} = (\ell_1 + 1) + \cdots + (\ell_n + 1) \quad \text{(see (2))}. \tag{8}$$

We have assumed that $m_0 = n \geq \widehat{E} + 1$, so $m_1 + \cdots + m_{\ell_1} \leq L - \widehat{E} - 1$. Because m_{ℓ_1} is a minimum, $m_{\ell_1} \leq \big(L - \widehat{E} - 1\big)/\ell_1$. Let $\widetilde{E} = |\{(i_\lambda, j_\lambda) \mid 1 \leq \lambda \leq \widehat{E}, j_\lambda \leq \ell_1 - 1\}|$ count the locations in I for derivatives of order $\leq \ell_1 - 1$. Note that $\widetilde{E} \leq \widehat{E}$. We bound $m_0 + \cdots + m_{\ell_1-1} - \widetilde{E} \geq (\ell_1^{[\text{new}]} + 1)D^{[\text{new}]} + 1 - \binom{\ell_1^{[\text{new}]}+1}{2}$, where $\deg(g) \leq D^{[\text{new}]} \stackrel{\text{def}}{=} \ell_1 - 1$ and $\ell_1^{[\text{new}]} = \ell_1 - 1$, as follows:

$$\begin{aligned}
&m_0 + \cdots + m_{\ell_1-1} - \widetilde{E} \\
&\quad \geq (L - m_{\ell_1}) - \widehat{E} \\
&\quad \geq \big(1 - \frac{1}{\ell_1}\big)(L - \widehat{E}) + \frac{1}{\ell_1} \quad (\text{by } m_{\ell_1} \leq \frac{1}{\ell_1}\big(L - \widehat{E} - 1\big)) \\
&\quad \geq \big(1 - \frac{1}{\ell_1}\big)\Big((\ell_1 + 1)D + 1 - \tbinom{\ell_1+1}{2}\Big) + \frac{1}{\ell_1} \quad (\text{by (6)}) \\
&\quad \geq \big(1 - \frac{1}{\ell_1}\big)\Big((\ell_1 + 1)\,\ell_1 - \tbinom{\ell_1+1}{2}\Big) + 1 \quad (\text{because } D \geq \ell_1) \\
&\quad = \frac{\ell_1^2 - 1}{2} + 1 \\
&\quad \geq \frac{\ell_1^2 - \ell_1}{2} + 1 \quad (\text{because } \ell_1 \geq 1) \\
&\quad = \ell_1(\ell_1 - 1) - \tbinom{\ell_1}{2} + 1 \\
&\quad = (\ell_1^{[\text{new}]} + 1)D^{[\text{new}]} + 1 - \tbinom{\ell_1^{[\text{new}]}+1}{2}.
\end{aligned} \tag{9}$$

Therefore, the induction hypothesis applies to the values of $g, \ldots, g^{(\ell_1-1)}$ and establishes Fact 2. □

In Algorithm 4.1, we will need a slightly more general estimate than (9).

Lemma 3.2. *Let ℓ_i, m_j, $D \geq \ell_1$ be as above with $\ell_1 \geq 1$, and let $\widetilde{E} \geq 0$. Assume that $n \geq \widetilde{E} + 1$ and*

$$(\ell_1 + 1) + \cdots + (\ell_n + 1) - \widetilde{E} \geq (\ell_1 + 1)D + 1 - \tbinom{\ell_1+1}{2}. \tag{10}$$

Then $m_0 + \cdots + m_{j-1} - \widetilde{E} \geq j(j-1) - \binom{j}{2} + 1$ for all $1 \leq j \leq \ell_1 + 1$.

Proof. We first prove by induction on $\nu \geq 0$ that

$$L_\nu - \widetilde{E} \geq \frac{\ell_1 - \nu}{\ell_1}\Big((\ell_1+1)D - \tbinom{\ell_1+1}{2}\Big) + 1, \quad \text{where } L_\nu \stackrel{\text{def}}{=} m_0 + \cdots + m_{\ell_1-\nu}. \tag{11}$$

The basis $\nu = 0$ is (10). Because $m_0 \geq \widetilde{E} + 1$, we have $m_1 + \cdots + m_{\ell_1-\nu} \leq L_\nu - \widetilde{E} - 1$. Because $m_{\ell_1-\nu}$ is a minimum, $m_{\ell_1-\nu} \leq (L_\nu - \widetilde{E} - 1)/(\ell_1 - \nu) = (L_\nu - \widetilde{E})/(\ell_1 - \nu) - 1/(\ell_1 - \nu)$. Therefore

$$\begin{aligned}
L_{\nu+1} - \widetilde{E} &= (L_\nu - m_{\ell_1-\nu}) - \widetilde{E} = (L_\nu - \widetilde{E}) - m_{\ell_1-\nu} \\
&\geq \big(1 - \frac{1}{\ell_1 - \nu}\big)(L_\nu - \widetilde{E}) + \frac{1}{\ell_1 - \nu} \\
&\geq \big(1 - \frac{1}{\ell_1 - \nu}\big)\Big(\frac{\ell_1 - \nu}{\ell_1}\big((\ell_1 + 1)D - \tbinom{\ell_1+1}{2}\big) + 1\Big) + \frac{1}{\ell_1 - \nu} \\
&\qquad\qquad \text{(by hypothesis (11))} \\
&= \frac{\ell_1 - \nu - 1}{\ell_1}\Big((\ell_1 + 1)D - \tbinom{\ell_1+1}{2}\Big) + 1,
\end{aligned}$$

the latter of which is (11) for $\nu+1$. For $j = \ell_1 - \nu + 1$ we have for (11) with $D \geq \ell_1$ that $(j-1)/\ell_1((\ell_1+1)D - \binom{\ell_1+1}{2}) \geq (j-1)(\ell_1+1)/2 \geq j(j-1)/2 = j(j-1) - \binom{j}{2}$. □

Example 3.1. We briefly show the calculation for the example from the abstract. If for $\ell = \ell_1 = \cdots = \ell_{E_{\text{tot}}} = 2$ we have $N_{\text{tot}} = 25D+25 = D+1+2E_{\text{tot}}(\ell+1)$, then $E_{\text{tot}} = 4D+4$ and $E_{\text{tot}}/N_{\text{tot}} = 4/25$. With L evaluations by (4) one can correct $E \stackrel{\text{def}}{=} \lfloor (L - 3D + 2)/2 \rfloor$ errors, because $L \geq 3D - 2 + 2E$. We have $E \geq (L - 3D + 1)/2$ and

$$\frac{L - 3D + 1}{2L} \geq \rho \stackrel{\text{def}}{=} \frac{4}{25} \iff (1-2\rho)L \geq 3D-1 \iff L \geq \frac{75}{17}D - \frac{25}{17}.$$

Therefore with $L = \lceil 75\,D/17 - 25/17 \rceil = \lceil (4+7/17)\,D - (1+8/17) \rceil$ one can correct E errors yielding an error rate $E/L \geq (L - 3D + 1)/(2L) \geq 4/25$. Note that $n \geq 2E + 1$ is always required.

For $N_{\text{zero}} = 5D - 2$ our new algorithm can correct $(N_{\text{zero}} - 3D + 2)/2 = D$ errors, for an error rate $D/(5D-2) > 1/5$. The example in Section 2 shows that for characteristic $p = 3$ one has an ambiguous interpolant for $N_{\text{tot}} = D + 6E_{\text{tot}}$, which has an error rate of $E_{\text{tot}}/N_{\text{tot}} = 1/(6 + D/E_{\text{tot}}) < 1/5$. □

Example 3.2. We now give an example for $\ell_1 = 2$ where the count (4) is minimal. Let $n = m_0 = D + (D-1) + (D-2) = 3D - 3$, $m_1 = (D-1) + (D-2) = 2D - 3$, $m_2 = D - 2$, and $2E_{\text{tot}} = 2(D-2) + (D-1) = 3D - 5$ (D an odd degree bound). Then $(\ell_0+1)+\cdots+(\ell_{3D-3}+1) = 6D-8 = (\ell_1+1)D - \binom{\ell_1+1}{2} + 2E_{\text{tot}} = N_{\text{zero}} - 1$. Indeed, there can be two interpolants, the zero polynomial and a polynomial f with $f''(\xi_i) = 0$ for $1 \leq i \leq m_2$, $f'(\xi_i) = 0$ for $m_2 + 1 \leq i \leq m_2 + (D-1) = m_1$, and $f(\xi_i) = 0$ for $m_1 + 1 \leq i \leq m_1 + D = m_0$. Both polynomials are zero at $(D-2) + (D-1) + D = 3D - 3$ values, and can have errors at half of the remaining $3D - 5 = 2E_{\text{tot}}$ values. In this example $n = m_0 = 2E_{\text{tot}} + 2$. With an additional value, the bound (4) is satisfied and by Lemma 3.1 there are no multiple interpolants. Furthermore, with $6D - 7$ values one cannot use Algorithm 4.1 in [3]. One can have errors at the first $E = (3D-5)/2 = (D-2) + (D-1)/2$ arguments ξ_i, so one has $D+1+2\sum_{i=1}^{E}(\ell_i+1) = D+1+6(D-2)+4(D-1)/2 = 9D-13 > 6D-7$ for sufficiently large D.

If $2E_{\text{tot}} > 3D - 5$, which we have assumed to be even, one may add new values $\xi_{3D-2}, \ldots, \xi_{2E_{\text{tot}}+2}$, thereby increasing $m_0 = 2E_{\text{tot}}+2$, and set alternatively $\hat{a}_{3D-2,0} = f(\xi_{3D-2})$, $\hat{a}_{3D-1,0} = 0, \ldots, \hat{a}_{2E_{\text{tot}}+1} = f(\xi_{2E_{\text{tot}}+1})$, $\hat{a}_{2E_{\text{tot}}+2,0} = 0$. □

Example 3.3. We give an example for $\ell_1 = 3$, $D = 4$ and $E_{\text{tot}} \geq 3$ where the count (4) is minimal. Let $f(x) = (x^2 - 1)(x^2 - 5)$, then $f'(x) = 4x(x^2 - 3)$, $f''(x) = 12(x^2 - 1)$, and $f^{(3)}(x) = 24x$. We set $\xi_1 = 0$, $\xi_2 = 1$, $\xi_3 = -1$, $\xi_4 = \sqrt{3}$, $\xi_5 = -\sqrt{3}$, $\xi_6 = \sqrt{5}$, $\xi_7 = -\sqrt{5}$.

We first treat the case $E_{\text{tot}} = 3$: let $n = m_0 = 7 = 2E_{\text{tot}}+1$, $m_1 = 5$, $m_2 = 3$ and $m_3 = 1$; therefore we evaluate $L = 7 + 5 + 3 + 1 = 16 =$

Table 1: Values for Example 3.3

$\hat{a}_{i,3}$	0						
$\hat{a}_{i,2}$	$f''(0)$	0	0				
$\hat{a}_{i,1}$	0	$f'(1)$	$f'(-1)$	0	0		
$\hat{a}_{i,0}$	0	0	0	0	0	0	0
ξ_i	$\xi_1 = 0$	$\xi_2 = 1$	$\xi_3 = -1$	$\xi_4 = \sqrt{3}$	$\xi_5 = -\sqrt{3}$	$\xi_6 = \sqrt{5}$	$\xi_7 = -\sqrt{5}$

$(\ell_1+1)D - \binom{\ell_1+1}{2} + 2E_{\text{tot}} = N_{\text{zero}} - 1$ derivates $f^{(j)}(\xi_i)$, $0 \le j \le 3$, $1 \le i \le m_j$. Of those, 6 values are $\ne 0$: $f(\xi_1)$, $f''(\xi_1)$, $f'(\xi_2)$, $f'(\xi_3)$, $f(\xi_4)$, and $f(\xi_5)$. If we select $\hat{a}_{1,2} = f''(\xi_1)$, $\hat{a}_{2,1} = f'(\xi_2)$, $\hat{a}_{3,1} = f'(\xi_3)$ and all other $\hat{a}_{i,j} = 0$ then both the polynomial f and the polynomial 0 have 3 errors, both for 3 distinct ξ_i's. In the following table, the errors for the f polynomial are indicated by red 0's, while the errors for the zero polynomial are at the 3 derivatives of f. The values are shown in Table 1.

The case $E_{\text{tot}} \ge 4$ is handled as in Example 3.2 by adding $\hat{a}_{8,0} = f(\xi_8)$, $\hat{a}_{9,0} = 0$, $\hat{a}_{10,0} = f(\xi_{10}), \ldots, \hat{a}_{2E_{\text{tot}}-1,0} = 0$, $\hat{a}_{2E_{\text{tot}},0} = f(\xi_{2E_{\text{tot}}})$, $\hat{a}_{2E_{\text{tot}}+1} = 0$ and setting $n = m_0 = 2E_{\text{tot}} + 1$. □

4 HERMITE DECODING OVER ZERO / LARGE CHARACTERISTIC

From the proof of Lemma 3.1 we can obtain a decoding algorithm, which is based on a Reed-Solomon decoder. Those Reed-Solomon algorithms receive as input $\hat{a}_{i,0} \in \mathsf{K}$ and distinct $\xi_i \in \mathsf{K}$ for $i = 1, \ldots, n$ and a degree bound $D < n$, and compute a polynomial f of degree $\le D$ such that $f(\xi_i) \ne \hat{a}_{i,0}$ at no more than $E = \lfloor (n - D - 1)/2 \rfloor$ indices i. The algorithms also indicate if no such f exists. Our Algorithm 4.1 in [3] specializes for $\ell_1 = \cdots = \ell_n = 0$ to a Reed-Solomon decoder.

We now present our new Hermite interpolation algorithm with error correction. Our algorithm iterates on ℓ_1, but a subsequent recursive iteration may not satisfy $n^{[\text{new}]} \ge 2E_{\text{tot}}^{[\text{new}]} + 1$. In that case the algorithm returns the unique $f^{(j^*)}$ for some $j^* \le \ell_1$, with which one can complete the original interpolation for $n \ge 2E_{\text{tot}} + 1$.

4.1 Error-correcting Hermite interpolation (zero / large char)

Input:

- Bounds $D, E_{\text{tot}} \in \mathbb{Z}_{\ge 0}$;
  the scalar field K has characteristic 0 or $\ge D + 1$.
- A set of n distinct argument values $\{\xi_1, \ldots, \xi_n\} \subseteq \mathsf{K}$;
- A list of n row vectors $\widehat{A} = [\widehat{A}_{i,*}]_{1 \le i \le n}$ where
  - $\ell_1 \ge \cdots \ge \ell_n \ge 0$; we shall have $D \ge \ell_1$,
    for otherwise all error-free $\hat{a}_{i,\ell_1}$ are 0.
  - $\widehat{A}_{i,*} = [\hat{a}_{i,0}, \ldots, \hat{a}_{i,\ell_i}] \in \mathsf{K}^{1 \times \ell_i}$;
  - $(\ell_1 + 1) + \cdots + (\ell_n + 1) \ge (\ell_1 + 1)D + 1 - \binom{\ell_1+1}{2} + 2E_{\text{tot}}$.

Output:

We call $f(x) \in \mathsf{K}[x]$ an interpolant if the following are satisfied:

- $\deg(f) \le D$;
- $k = |I| \le E_{\text{tot}}$ for $I = \{(i_\lambda, j_\lambda)\}_{1 \le \lambda \le k}$
  $= \{(i, j) \mid f^{(j)}(\xi_i) \ne \hat{a}_{i,j}, 1 \le i \le n, 1 \le j \le \ell_i\}$;
- $f^{(j)}(\xi_i) = \hat{a}_{i,j}$ for all $1 \le i \le n$ and $0 \le j \le \ell_i$ and $(i, j) \notin I$.

- *Case* $n \ge 2E_{\text{tot}} + 1$:
  - The interpolant $f(x) \in \mathsf{K}[x]$ and the error locations $I = \{(i_\lambda, j_\lambda)\}_{1 \le \lambda \le k}$;
  - Or a message indicating there is no such interpolant.
- *Case* $n \le 2E_{\text{tot}}$:
  - A j^* with $0 \le j^* \le \ell_1$ and $g(x) \in \mathsf{K}[x]$ and the error locations $I = \{(i_\lambda, j_\lambda)\}_{1 \le \lambda \le k}$, which satisfy
    - $\deg(g) \le D - j^*$;
    - $k = |I| \le E_{\text{tot}}$ for
      $I = \{(i_\lambda, j_\lambda)\}_{1 \le \lambda \le k}$
      $= \{(i, j) \mid g^{(j)}(\xi_i) \ne \hat{a}_{i,j}, 1 \le i \le m_{j^*}, j^* \le j \le \ell_i\}$;
    - $g^{(j)}(\xi_i) = \hat{a}_{i,j}$ for all $1 \le i \le m_{j^*}$, $j^* \le j \le \ell_i$ and $(i, j) \notin I$;
    - If at least one interpolant f exists, $g = f^{(j^*)}$, which is then unique for all interpolants; specifically, if $j^* = 0$ then g is a unique interpolant;
  - Or a message indicating there is no interpolant f. Note that a g may be returned even if there is no interpolant.

1. *If* $\ell_1 = 0$, *then* $n \ge D + 1 + 2E_{\text{tot}}$: *perform Reed-Solomon interpolation and return either* f *or "no interpolant exists."*
2. *If* $n \ge 2E_{\text{tot}} + 1$ *and*

$$L = (\ell_1 + 1) + \cdots + (\ell_n + 1) \ge D + 1 + 2(\ell_1 + 1) + \cdots + 2(\ell_{E_{\text{tot}}} + 1), \quad (12)$$

call Algorithm 4.1 *in* [3] *with* $E = E_{\text{tot}}$ *to interpolate* f. Note that if $n \ge E_{\text{tot}}$, $L \ge D + 1 + 2\sum_{i=1}^{E_{\text{tot}}}(\ell_i + 1) \Longrightarrow n \ge 2E_{\text{tot}} + 1$, because if $n \le 2E_{\text{tot}}$ we would have

$$2(\ell_1+1) + \cdots + 2(\ell_{E_{\text{tot}}}+1) \ge (\ell_1+1) + \cdots + (\ell_{E_{\text{tot}}}+1) + (\ell_{E_{\text{tot}}+1}+1) + \cdots + (\ell_n+1), \quad (13)$$

in contradiction to (12).
A second algorithm is described in Remark 4.1 below.
3. $E_0 \leftarrow \lfloor (n - D - 1)/2 \rfloor$. *If* $E_0 \ge 0$ *then attempt a Reed-Solomon interpolation of* f *from* $\hat{a}_{i,0}$ *for* $1 \le i \le n$ *with degree bound* D *and with bound* E_0 *for the number of errors. If the decoding yields a candidate* f, *then check if* $f, f', \ldots, f^{(\ell_1)}$ *interpolate all* $\hat{a}_{i,j}$ *with* $\le E_{\text{tot}}$ *errors.*
If there is success and $n \ge 2E_{\text{tot}} + 1$, *then return* f *and the error locations.* By Lemma 3.1 with $\widehat{E} = 2E_{\text{tot}}$ the interpolant is unique.
If there is success and $n \le 2E_{\text{tot}}$,
then $j^* \leftarrow \ell_1$;
$g \leftarrow f^{(\ell_1)}$;
$I^{[\text{high}]} \leftarrow \{(i, \ell_1) \mid f^{(\ell_1)}(\xi_i) \ne \hat{a}_{i,\ell_1}, 1 \le i \le m_{\ell_1}\}$;
go to Step 6.
By Lemma 3.1 with $\widehat{E} = 2E_{\text{tot}}$, for all interpolants the ℓ_1-st

derivative is unique. In Example 4.2 we show that the interpolants may be ambiguous.

4. Here we know that if there is an interpolant f, then either $n < D+1$ or the number k_0 of errors for f in $\hat{a}_{i,0}$ satisfies $k_0 \geq E_0 + 1 \geq (n-D-1)/2 - 1/2 + 1 = (n-D)/2$. Note that the Reed-Solomon algorithm may compute in Step 3 a candidate for f which fails the error count elsewhere, in which case any interpolant f has more than E_0 errors in the $\hat{a}_{i,0}$. Both cases lead to $n \leq D + 2k_0$. Therefore, if an interpolant f exists, we have $\leq E_{\text{tot}} - k_0$ errors in the remaining values $\hat{a}_{i,j}$, with $1 \leq i \leq m_1$, $1 \leq j \leq \ell_i$. The number of such values for $f', \ldots, f^{(\ell_1)}$ is bounded as

$$\begin{aligned}\ell_1 + \cdots + \ell_n &\geq (\ell_1+1)D + 1 - \tbinom{\ell_1+1}{2} + 2E_{\text{tot}} - n \\ &\geq (\ell_1+1)D + 1 - \tbinom{\ell_1+1}{2} + 2E_{\text{tot}} - D - 2k_0 \\ &= \ell_1(D-1) + 1 - \tbinom{\ell_1}{2} + 2(E_{\text{tot}} - k_0). \qquad (14)\end{aligned}$$

Recursively, call Algorithm 4.1 with degree bound $D^{[\text{new}]} = D-1$, *error bound*

$$E_{\text{tot}}^{[\text{high}]} \leftarrow \left\lfloor \frac{1}{2}\left(\ell_1 + \cdots + \ell_n - \ell_1(D-1) - 1 + \tbinom{\ell_1}{2}\right) \right\rfloor \quad (\text{which is} \geq E_{\text{tot}} - k_0), \qquad (15)$$

$n^{[\text{new}]} \leftarrow m_1$ *and derivative values* $\hat{a}_{i,j-1}^{[\text{new}]} \leftarrow \hat{a}_{i,j}$ *for* $1 \leq i \leq m_1$ *and* $1 \leq j \leq \ell_i$;
The recursive call may for $n^{[\text{new}]} = m_1 \leq 2E_{\text{tot}}^{[\text{high}]}$ return j^*, g and I, for $n^{[\text{new}]} \geq 2E_{\text{tot}}^{[\text{high}]} + 1$ return an interpolant, denoted by $f^{[\text{high}]}$, and I, or in either case "no interpolant."
If "no interpolant" was returned then return that no interpolant f *exists.*
Because we use j^* and g in subsequent steps, we assign $f^{[\text{high}]}$ appropriately.
If $n^{[\text{new}]} \geq 2E_{\text{tot}}^{[\text{high}]} + 1$ *then* $j^* \leftarrow 0; g \leftarrow f^{[\text{high}]}$.

5. Shift the derivatives orders to pre-recursive call values.
$j^* \leftarrow j^* + 1$; $I^{[\text{high}]} \leftarrow \{(i_\lambda, j_\lambda + 1) \mid (i_\lambda, j_\lambda) \in I\}$.

6. $E_{\text{tot}}^{[\text{low}]} \leftarrow E_{\text{tot}} - |I^{[\text{high}]}|$; *if* $n \leq 2E_{\text{tot}}^{[\text{low}]}$ *then return* j^*, g *and* $I^{[\text{high}]}$.

7. Now $n \geq 2E_{\text{tot}}^{[\text{low}]} + 1$, where $E_{\text{tot}}^{[\text{low}]}$ is an upper bound for the allowed number of errors in the derivatives of order $\leq j^* - 1$. One can attempt to complete the interpolation of the then unique f.
From g *compute* $\bar{f}(x) = c_d x^d + \cdots + c_{j^*} x^{j^*}$ *with* $\bar{f}^{(j^*)} = g$. Note that $d \leq D$. One can integrate g j^*-times because the characteristic of K is 0 or $\geq D+1$.

8. We interpolate $(f \bmod x^{j^*}) = f - \bar{f}$ from the derivative values of order $\leq j^* - 1$.
$D^{[\text{new}]} \leftarrow j^* - 1$; $\ell_i^{[\text{new}]} \leftarrow \min\{\ell_i, j^* - 1\}$ *for all* $1 \leq i \leq n$;
$\hat{a}_{i,j}^{[\text{new}]} \leftarrow \hat{a}_{i,j} - \bar{f}^{(j)}(\xi_i)$ *for all* $1 \leq i \leq n$ *and* $0 \leq j \leq \ell_i^{[\text{new}]}$.
Recursively, call Algorithm 4.1 with $D^{[\text{new}]}$, $E_{\text{tot}}^{[\text{low}]}$, $[\xi_i]_{1 \leq i \leq n}$, $\widehat{A}^{[\text{new}]}$.
There are $m_0 + \cdots + m_{j^*-1}$ values in $\widehat{A}^{[\text{new}]}$. By $E_{\text{tot}}^{[\text{low}]} \leq E_{\text{tot}}$ we have

$$\begin{aligned}(\ell_1+1) + \cdots + (\ell_n+1) - 2E_{\text{tot}}^{[\text{low}]} &\geq (\ell_1+1) + \cdots + (\ell_n+1) - 2E_{\text{tot}} \\ &\geq (\ell_1+1)D + 1 - \tbinom{\ell_1+1}{2},\end{aligned}$$

and by Lemma 3.2 with $\widetilde{E} = 2E_{\text{tot}}^{[\text{low}]}$ we have

$$\begin{aligned}m_0 + \cdots + m_{j^*-1} - 2E_{\text{tot}}^{[\text{low}]} &\geq j^*(j^*-1) - \tbinom{j^*}{2} + 1 \\ &= (\ell_1^{[\text{new}]} + 1)D^{[\text{new}]} - \tbinom{\ell_1^{[\text{new}]}+1}{2} + 1, \qquad (16)\end{aligned}$$

so the input specifications are satisfied with $n \geq 2E_{\text{tot}}^{[\text{low}]} + 1$. Therefore, the recursive call returns either an interpolant, denoted by $f^{[\text{mod}]}$, and error index set $I^{[\text{low}]}$, or a message indicating that there is no interpolant.

9. *If no interpolant* $f^{[\text{mod}]}$ *and error index set* $I^{[\text{low}]}$ *is computed in Step 8, then return that no interpolant* f *exists.*

10. The total number of errors satisfies $|I^{[\text{low}]}| + |I^{[\text{high}]}| \leq E_{\text{tot}}^{[\text{low}]} + |I^{[\text{high}]}| = E_{\text{tot}}$.
If $n \geq 2E_{\text{tot}} + 1$
then return $f \leftarrow \bar{f} + f^{[\text{mod}]}$ *and* $I \leftarrow I^{[\text{low}]} \cup I^{[\text{high}]}$;
Else return $j^* \leftarrow 0, g \leftarrow \bar{f} + f^{[\text{mod}]}$ *and* $I \leftarrow I^{[\text{low}]} \cup I^{[\text{high}]}$.

Remark 4.1. Step 2 performs multiplicity code decoding when the error rate is small. Our algorithm in [3] is, for polynomial interpolation, essentially the Welch-Berlekamp algorithm, which is described in [5, Section 3.1.1]. There is the following alternative, based on Lagrange interpolation and Chinese remaindering with error correction. In [8] a Lagrangian interpolation formula is given for Hermite polynomial interpolation. For a polynomial $f(x) \in \mathsf{K}[x]$ we have

$$f(x) \equiv f(\xi_i) + f'(\xi_i)(x-\xi_i) + \cdots + \frac{f^{(\ell_i)}(\xi_i)}{\ell_i!}(x-\xi_i)^{\ell_i} \pmod{(x-\xi_i)^{(\ell_i+1)}}, 1 \leq i \leq n.$$

Therefore, one can Chinese remainder the polynomial residues $\sum_{j=0}^{\ell_i} \hat{a}_{i,j}/j!(x-\xi_i)^j$ with respect to the polynomial moduli $(x-\xi_i)^{\ell_i+1}$, and correct erroneous residues; see [4] and the literature cited there. Each erroneous residue requires another good residue, so if the residues for $i = 1, 2, \ldots, E_{\text{tot}}$ are erroneous one needs an additional $(\ell_1+1) + \cdots + (\ell_{E_{\text{tot}}}+1)$ good values, in addition to $D+1$ good values, which is the count (2). □

Remark 4.2. When Algorithm 4.1 returns "no interpolant," in both the cases $n \geq 2E_{\text{tot}} + 1$ and $n \leq 2E_{\text{tot}}$ for some inputs the algorithm computes an order $j^* \geq 1$ and a polynomial g that interpolates all j-th order derivatives for $j^* \leq j \leq \ell_1$ with $|I| \leq E_{\text{tot}}$ errors. Optionally, the triple j^*, g, I could be returned as a partial solution. □

Example 4.1. If $n \geq 2E_{\text{tot}}+1$, any valid input with no interpolant will be flagged. We show by example that for $n \leq 2E_{\text{tot}}$, a j^* and g may be returned even if there is no unique interpolant. Let $n = 2E_{\text{tot}} \geq 2D$ and $\ell_1 = 1$, $m_1 = 2D$; then $n + m_1 = (\ell_1+1)D + 1 - \binom{\ell_1+1}{2} + 2E_{\text{tot}}$ and the input specification is satisfied. Suppose for a polynomial f of degree $\leq D$ we have $f'(\xi_i) = \hat{a}_{i,1}$ for all $1 \leq i \leq 2D$, and $\hat{a}_{i,0} = f(\xi_i)$ and $\hat{a}_{E_{\text{tot}}+i,0} = f(\xi_i) + 1$ for all

$1 \le i \le E_{\text{tot}}$. Both f and $f+1$ interpolate all points with E_{tot} errors, but Algorithm 4.1 returns $j^* = 1$ and f' either in Step 3 or in Step 6. □

Example 4.2. We show by example that in Step 3 for $n \le 2E_{\text{tot}}$ there may be ambiguous interpolants. Let $\ell_1 = 1$, $n = 2E_{\text{tot}} - (D+1) \ge 3D+1$ and $m_1 = 3D+1$. Then $(\ell_1+1)+(\ell_2+1) = n + m_1 = 2D + 2E_{\text{tot}} = (\ell_1+1)D + 1 - \binom{\ell_1+1}{2} + 2E_{\text{tot}}$ and the input specifications are satisfied. Now let f be a polynomial of degree $\le D$ and let $\hat{a}_{i,0} = f(\xi_i)$ for $1 \le i \le E_{\text{tot}}$, let $\hat{a}_{i,0} = f(\xi_i)+1$ for $E_{\text{tot}}+1 \le i \le n$ and let $\hat{a}_{i,1} = f'(\xi_i)$ for $1 \le i \le m_1$. Now f has $E_{\text{tot}} - (D+1) = (n-(D+1))/2$ errors at $(E_{\text{tot}}+1, 0), \ldots (n, 0)$ and will be computed in Step 3, but $f+1$ has E_{tot} errors at $(1,0), \ldots, (E_{\text{tot}}, 0)$ and constitutes a second interpolant with $\le E_{\text{tot}}$ errors. Therefore, a successful interpolant found in Step 3 does not constitute the only solution, and Step 6 can only return the unique f'. □

Example 4.3. We show another example which in Step 3 for $n \le 2E_{\text{tot}}$ has two interpolants, now for $\ell_1 = 2$. Let $m_0 = m_1 = m_2 = 3D - 2 = E_{\text{tot}}$ and $\ell_1 = \cdots = \ell_{3D-2} = 2$. Then $(\ell_1+1)+\cdots+(\ell_{3D-2}+1) = (\ell_1+1)D + 1 - \binom{3}{2} + 2E_{\text{tot}}$ and the input conditions are satisfied. Suppose that f interpolates $\hat{a}_{i,j}$ without error for all i and $j = 0$ and $j = 2$. If $\hat{a}_{i,1} = f'(\xi_i)+1$ then $g(x) = f(x) + x$ interpolates $\hat{a}_{i,j}$ for all i and $j = 1, 2$. Both f and g have $\le E_{\text{tot}}$ errors. □

Algorithm 4.1 makes recursive calls with different error bounds, so if for the recursive call $n^{[\text{new}]} \ge 2E_{\text{tot}}^{[\text{new}]} + 1$ that recursive call may produce a unique interpolant. Therefore, if $n \le 2E_{\text{tot}}$ initially, the returned g may have $j^* < \ell_1$. We now prove that the input/output specifications are enforced by Algorithm 4.1.

Theorem 4.1. *Algorithm 4.1 computes an interpolant f or g if one or more interpolants exist. If $n \ge 2E_{\text{tot}}+1$ it returns the then unique f or diagnoses that no such f exists. If $n \le 2E_{\text{tot}}$ and there exists at least one interpolant f, then $f^{(j^*)} = g$, which is unique for all interpolants.*

Proof. We first consider the case $n \le 2E_{\text{tot}}$. If there is no interpolant f for all values in $\widehat{A}$ with $\le E_{\text{tot}}$ errors, then either the g or "no interpolant" outputs are correct. Now suppose that there is one interpolant f or more. By Lemma 3.1 the ℓ_1-st derivatives of all such interpolants are unique. Therefore, Step 3, if successful, computes a correct $j^* = \ell_1$ and g. Otherwise, all interpolants have $\ge (n-D)/2$ errors in the $\hat{a}_{i,0}$, and by (14) and (15) there are $\le E_{\text{tot}} - k_0 \le E_{\text{tot}}^{[\text{high}]}$ errors in all other values $\hat{a}_{i,j}$ with $j \ge 1$. The number of values for any f' is bounded as

$$m_1 + \cdots + m_{\ell_1} = \ell_1 + \cdots + \ell_n \ge \ell_1(D-1) + 1 + \binom{\ell_1}{2} + 2E_{\text{tot}}^{[\text{high}]},$$

which is the induction hypothesis for the correctness of Algorithm 4.1, and Steps 5 and 6 return correct outputs.

Because the j^*-th derivative of all possible interpolants f' is equal to g after Step 5 by hypothesis, and the number of errors in derivatives of order $\le j^*-1$ of any possible interpolant $f^{[\text{mod}]}$ in Step 8 satisfies the input specifications by (16), Steps 7–10 by hypothesis correctly compute the unique $f^{[\text{mod}]}$ or flag that none exists.

We now consider the case $n \ge 2E_{\text{tot}}+1$. If there exists an interpolant f, which is then unique, then $g = f^{(j^*)}$ at Step 7 by the hypothesis for Step 4. In Step 8 the input specifications for $f \bmod x^{j^*}$ are satisfied and the Step computes by induction hypothesis correctly the missing residue of f. If no f exists, Step 9 will flag such input, because only interpolants with $\le E_{\text{tot}}$ errors are returned in Step 10. □

Theorem 4.2. *Algorithm 4.1 performs $O(\ell_1(D+\ell_1^2)L(\log L)^2 \times \text{loglog}(L))$ arithmetic operations in* K, *where* $L = (\ell_1+1)+\cdots+(\ell_n+1)$ (2).

Proof. New interpolant polynomials are computed only in Steps 1–3 of each recursive invocation. The most costly is Step 3 with $O(D \times L(\log L)^2 \text{loglog}(L))$ arithmetic steps. The cost for the Reed-Solomon algorithm and Step 2 is $O(L(\log L)^2 \text{loglog}(L))$, but in Step 3 there are evaluations of polynomials of degree $\le D$ at m_0, $m_1, \ldots, m_{\ell_1}$ points. Note that $D < L$ which implies $D^2 = O(DL)$ for the cost of computing all derivatives of f in Step 3. The recursive descent in Step 4 may take ℓ_1 recursive calls before a candidate g is produced, at a total cost of $O(\ell_1 DL(\log L)^2 \text{loglog}(L))$.

Subsequently, $\widehat{A}$ is updated, again for each new interpolant $\bar{f}$ at cost no more than $O(DL(\log L)^2 \text{loglog}(L))$. Then the algorithm continues at the new global derivative order $\ell_1^{[\text{new}]} = j^* - 1 \le \ell_1 - 1$ in Step 8, computing interpolants and updating $\widehat{A}$. Because the recursive call at Step 8 can be reached at most ℓ_1 times, as the global order j^* decreases each time, and $m_0 + \cdots + m_{j^*-1} < L$ and $D^{[\text{new}]} < \ell_1$, the overall complexity is bounded by ℓ_1 times the bound $O(\ell_1^2 L(\log L)^2 \text{loglog}(L))$ for each subsequent descent. □

Remark 4.3. List-decoding of multiplicity codes [1, 6] interpolates in polynomial-time a list of valid polynomials, that is, code words, from a message word of polynomial values, which contain errors, from fewer than N_{tot} (5) evaluations. If the list-decoding error-rate is no more than the error rate for our new count N_{zero} (4) and the number of distinct arguments n is sufficiently large, then for fields of characteristic 0 or $\ge D+1$ the returned list contains no more than one element. Our new algorithm could also perform Reed-Solomon list-decoding and list-decode from a number of evaluations $< N_{\text{zero}}$. □

ACKNOWLEDGMENTS

This research was supported by the National Science Foundation under Grant CCF-1717100 (Kaltofen).

REFERENCES

[1] Venkatesan Guruswami and Carol Wang. 2011. Optimal rate list decoding via derivative codes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, Leslie Ann Goldberg, Klaus Jansen, R. Ravi, and José D. P. Rolim (Eds.). Springer, 593–604.

[2] Erich L. Kaltofen and Clément Pernet. 2014. Sparse Polynomial Interpolation Codes and Their Decoding Beyond Half the Minimal Distance. In *ISSAC 2014 Proc. 39th Internat. Symp. Symbolic Algebraic Comput.*, Katsusuke Nabeshima (Ed.). Association for Computing Machinery, New York, N. Y., 272–279. URL: http://users.cs.duke.edu/~elk27/bibliography/14/KaPe14.pdf.

[3] Erich L. Kaltofen, Clément Pernet, and Zhi-Hong Yang. 2020. Hermite Rational Function Interpolation with Error Correction. In *Computer Algebra in Scientific Computing, CASC 2020 (Lect. Notes Comput. Sci., Vol. 12291)*, F. Boulier, M. England, T. Sadykov, and E. Vorozhtsov (Eds.). Springer, 335–357. URL: http://users.cs.duke.edu/~elk27/bibliography/20/KPY20.pdf.

[4] Majid Khonji, Clément Pernet, Jean-Louis Roch, Thomas Roche, and Thomas Stalinski. 2010. Output-sensitive decoding for redundant residue systems. In *Proc. 2010 Internat. Symp. Symbolic Algebraic Comput. ISSAC 2010*, Stephen M. Watt (Ed.). Association for Computing Machinery, New York, N. Y., 265–272. URL: https://dl.acm.org/doi/10.1145/1837934.1837985.

[5] Swastik Kopparty. 2014. Some remarks on multiplicity codes. In *Discrete Geometry and Algebraic Combinatorics: AMS Spec. Session (Contemporary Mathematics, Vol. 625)*, Alexander Barg and Oleg R. Musin (Eds.). 155–176. URL: https://sites.math.rutgers.edu/~sk1233/multcode-survey.pdf.

[6] Swastik Kopparty. 2015. List-decoding multiplicity codes. *Theory of Computing* 11, 1 (2015), 149–182. URL: https://sites.math.rutgers.edu/~sk1233/part2.pdf.

[7] M. Yu. Rosenbloom and Michael A. Tsfasman. 1997. Codes for the m-metric. *Problemy Peredachi Informatsii* 33, 1 (1997), 55–63.

[8] A. Spitzbart. 1960. A Generalization of Hermite's Interpolation Formula. *The American Mathematical Monthly* 67, 1 (1960), 42–46. DOI: 10.1080/00029890.1960.11989446.