Sparse Polynomial Interpolation Codes and their decoding beyond half the minimal distance*

Erich L. Kaltofen
Dept. of Mathematics, NCSU
Raleigh, NC 27695, USA
kaltofen@math.ncsu.edu
www4.ncsu.edu/~kaltofen,
Clément Pernet
U. J. Fourier, LIP-AriC, CNRS, Inria, UCBL, ÉNS de Lyon
46 Allée d’Italie, 69364 Lyon Cedex 7, France
clement.pernet@imag.fr
http://membres-liglab.imag.fr/pernet/
March 18, 2014

Abstract
We present algorithms performing sparse univariate polynomial interpolation with errors in the evaluations of the polynomial. Based on the initial work by Comer, Kaltofen and Pernet [Proc. ISSAC 2012], we define the sparse polynomial interpolation codes and state that their minimal distance is precisely the length divided by twice the sparsity. At ISSAC 2012, we have given a decoding algorithm for as much as half the minimal distance and a list decoding algorithm up to the minimal distance.

Our new polynomial-time list decoding algorithm uses sub-sequences of the received evaluations indexed by a linear progression, allowing the decoding for a larger radius, that is, more errors in the evaluations while returning a list of candidate sparse polynomials. We quantify this improvement for all typically small values of number of terms and number of errors, and provide a worst case asymptotic analysis of this improvement. For instance, for sparsity $T = 5$ with $\leq 10$ errors we can list decode in polynomial-time from 74 values of the polynomial with unknown terms, whereas our earlier algorithm required $2T(E + 1) = 110$ evaluations.

We then propose two variations of these codes in characteristic zero, where appropriate choices of values for the variable yield a much larger minimal distance: the length minus twice the sparsity.

---

*This material is based on work supported in part by the National Science Foundation under Grants CCF-0830347 and CCF-1115772 (Kaltofen), the French ANR under Grant HPAC ANR-11-BS02-013 (Pernet), and the Inria associate team grant QOLAPS (Pernet).
Criteria and Subject Descriptors:
I.1.2 [Symbolic and Algebraic Manipulation]: Algorithms; G.1.1 [Numerical Analysis]: Interpolation-smoothing;
E.4 [Coding and Information Theory]: Error control codes

General Terms: Algorithms, Reliability

Keywords: Blahut’s algorithm, Prony’s algorithm, sparse polynomial interpolation, exact polynomial fitting with errors.

1 Introduction

Evaluation/interpolation schemes is a key tool in many of todays computations. Model fitting for empirical data sets is a well-known one, where additional information on the model helps improving the fit. In particular, models of natural phenomena often happen to be sparse, which has motivated a wide range of research including compressive sensing [3], and sparse interpolation of polynomials [20, 1, 12, 10, 7, 9]. Most algorithms for the latter problem rely on the connection between linear complexity and sparsity, often referred to as Blahut’s theorem (theorem 1 [2, 16]) although it was already used in the 18th century by Prony [20]. The Berlekamp/Massey algorithm [17] makes this connection effective. These exact sparse interpolation techniques have been very successfully applied to numeric computations [8, 13, 5, 14].

In computer algebra, evaluation/interpolation schemes are also widely used as a key computational tool: reducing operations on polynomials to base ring operations, integer and rationals operations to finite fields operations, multivariate polynomials operations to univariate polynomials operations, etc. With the rise of large scale parallel computers, their ability to convert a large sequential computation, into numerous smaller independent tasks is of high importance.

Evaluation/interpolation schemes are also at the core of the famous Reed-Solomon error correcting codes [22, 19]. A block of information, viewed as a dense polynomial over a finite field is encoded by its evaluation in \( n \) points. Decoding is achieved by an interpolation resilient to errors. Blahut’s theorem [2, 16] originates from the decoding of Reed-Solomon codes: the interpolation of the error vector of sparsity \( t \) is a sequence of linear complexity \( t \) whose generator, computed by Berlekamp/Massey algorithm, carries in its roots the information of the error locations. Beyond the field of digital communication and data storage, error correcting codes have found more recent applications in fault tolerant distributed computations [11, 15, 6]. In particular, a parallelization based on evaluation/interpolation techniques can be made fault tolerant if interpolation with error can be performed. This is achieved by Reed-Solomon codes for dense polynomial interpolation and by CRT codes, for residue number systems [15]. The problem of sparse polynomial interpolation with errors rises naturally in this context. We introduced it in [5] and proposed an algorithms to solve it. It is naturally related to the \( k \)-error linear complexity problem [18] from stream cipher theory. A major concern in our previous results is that in order to correct \( E \) errors, the number of evaluations has to be increased by a multiplicative factor linear in \( E \). In comparison, dense interpolation with errors only requires an additive term linear in \( E \). In this paper we further investigates this problem from a coding theory viewpoint. In section 2 we define the sparse polynomial interpolation codes and state that their minimal distance is precisely the length...
divided by twice the sparsity. The algorithms of [5] can be viewed as a unique decoding algorithm for as much as half the minimal distance and a list decoding algorithm up to the minimal distance. In section 3, we propose a new polynomial-time list decoding algorithm that uses sub-sequences of the received evaluations indexed by a linear progression, allowing the decoding for a larger radius. We quantify this improvement on average by experiments, in the worst case for all typically small values of number of terms and number of errors, and provide a worst case asymptotic analysis of this improvement. We then propose in section 5 two variations of these codes in characteristic zero, where appropriate choices of values for the variable yield a much larger minimal distance: the length minus twice the sparsity.

**Linear recurring sequences.** We recall that a sequence \((a_0, a_1, \ldots)\) is linear recurring if there exist \(\lambda_0, \lambda_1, \ldots, \lambda_{t-1}\) such that \(a_{j+t} = \sum_{i=0}^{t-1} \lambda_i a_{i+j} \forall j \geq 0\). The polynomial \(\Lambda(z) = z^t - \sum_{i=0}^{t-1} \lambda_i z^i\) is called a generating polynomial of the sequence and the generating polynomial with least degree is called the minimal generating polynomial of the sequence and its degree is the linear complexity of the sequence.

These definitions can be extended to vectors, viewed as finite contiguous sub-sequences of an infinite sequence. The minimal generating polynomial of an \(n\)-dimensional vector is the monic polynomial \(\Lambda(z) = z^t - \sum_{i=0}^{t-1} \lambda_i z^i\) of least degree such that \(a_{j+t} = \sum_{i=0}^{t-1} \gamma_i a_{i+j} \forall 0 \leq j \leq n - t - 1\). Note that consequently, any vector is linearly recurring with linear complexity \(\leq n\).

**Theorem 1** (Blahut [2, 16]). The linear complexity of an \(N\)-periodic sequence \(A = (a_0, \ldots, a_{N-1}, a_0, \ldots)\) is equal to the Hamming weight of the discrete Fourier transform of \((a_0, \ldots, a_{N-1})\).

**The Ben-Or/Tiwari Algorithm.** We briefly review Ben-Or’s/Tiwari’s algorithm in the setting of univariate sparse polynomial interpolation. Let \(f\) be a univariate polynomial, \(m_j\) its distinct terms, \(t\) the number of terms, and \(c_j\) the corresponding non-zero coefficients: \(f(x) = \sum_{j=1}^{t} c_j x^{e_j} = \sum_{j=1}^{t} c_j m_j \neq 0, c_j \in \mathbb{Z}\).

**Theorem 2.** [1] Let \(b_j = \omega^{c_j}\), where \(\omega\) is a value from the coefficient domain to be specified later, let \(a_i = f(\omega^i) = \sum_{j=1}^{t} c_j b_j^i\), and let \(\Lambda(z) = \prod_{j=0}^{t}(z - b_j) = z^t + \lambda_{t-1} z^{t-1} + \cdots + \lambda_0\). The sequence \((a_0, a_1, \ldots)\) is linearly generated by the minimal polynomial \(\Lambda(z)\).

The Ben-Or/Tiwari algorithm then proceeds in the four following steps:
1. Find the minimal-degree generating polynomial \(\Lambda\) for \((a_0, a_1, \ldots)\), using the Berlekamp/Massey algorithm.
2. Compute the roots \(b_j\) of \(\Lambda\), using univariate polynomial factorization.
3. Recover the exponents \(e_j\) of \(f\), by repeatedly dividing \(b_j\) by \(\omega\).
4. Recover the coefficients \(c_j\) of \(f\), by solving the transposed \(t \times t\) Vandermonde system

\[
\begin{bmatrix}
1 & 1 & \ldots & 1 \\
1 & 2 & \ldots & t \\
\vdots & \vdots & \ddots & \vdots \\
1 & b_1^{-1} & \ldots & b_t^{-1}
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2 \\
\vdots \\
c_t
\end{bmatrix}
= 
\begin{bmatrix}
a_0 \\
a_1 \\
\vdots \\
a_{t-1}
\end{bmatrix}
\]

By Blahut’s theorem, the sequence \((a_j)_{j \geq 0}\) has linear complexity \(t\), hence only \(2t\) coefficients suffice for the Berlekamp/Massey algorithm to recover the minimal polynomial \(\Lambda\). In the presence of errors in some of the evaluations, this fails.
2 Sparse Polynomial interpolation codes

Definition 1. Let $K$ be a field and $\alpha$ a primitive $m$-th root of unity in $K$. A sparse polynomial evaluation code of parameters $(n, t)$ over $K$ is defined as the set

$$\mathcal{C}(n,T) = \{ (f(\alpha^0), f(\alpha^1), \ldots, f(\alpha^{n-1})) : f \in K[X] \}
\text{is } t\text{-sparse with } t \leq T \text{ and } \deg f < m \}$$

Theorem 3. If $K$ has not characteristic 2, a $(n, T)$-sparse polynomial evaluation code over $K$ has minimal distance $\delta = \frac{n}{2T}$.

The following proof is adapted from [5, §2.1].

Proof. Let $\bar{0}$ denote the zero vector of length $T - 1$. Consider two vectors of $K^n$:

$$x_1 = (\bar{0}, 1, \bar{0}, 1, \ldots)$$
$$x_2 = (\bar{0}, 1, \bar{0}, -1, 0, 1, \bar{0}, -1, \ldots)$$

formed by the repetition of their first $2T$ values. The sequence $x_1$ is generated by $\Lambda_1(z) = -1 + z^T$ and $x_2$ by $\Lambda_2(z) = 1 + z^T$. Applying Blahut’s theorem, we deduce that these vectors originate from evaluations of $T$-sparse polynomials. Hence $x_1$ and $x_2$ are code-words of an $(n, T)$-sparse evaluation code. Since $x_1$ and $x_2$ differ by exactly $\left\lfloor \frac{n}{2T} \right\rfloor$ values, this is an upper bound on the minimal distance $\delta$.

Now consider any pair of distinct code-words $x$ and $y$ and consider their $\left\lfloor \frac{n}{2T} \right\rfloor$ sub-vectors

$$x^{(1)} = (x_1, \ldots, x_{2T}), \quad y^{(1)} = (y_1, \ldots, y_{2T})$$
$$x^{(2)} = (x_{2T+1}, \ldots, x_{4T}), \quad y^{(2)} = (y_{2T+1}, \ldots, y_{4T})$$
$$\vdots$$
$$x^{(\left\lfloor \frac{n}{2T} \right\rfloor)}, \quad y^{(\left\lfloor \frac{n}{2T} \right\rfloor)}$$

If for some $i$, $x^{(i)} = y^{(i)}$ then the vector $z^{(i)} = x^{(i)} - y^{(i)}$ is all zero and is the evaluation of a less than $2T$-sparse polynomial $f - g$. Solving the $2T \times 2T$ corresponding Vandermonde system yields $f = g$ which is a contradiction. Hence $x$ and $y$ differ in at least $\left\lfloor \frac{n}{2T} \right\rfloor$ positions, and consequently $\delta = \left\lfloor \frac{n}{2T} \right\rfloor$.

Unique decoding. There exists an algorithm that does unique decoding of such codes up to the minimal distance: the Majority Rule Berlekamp/Massey algorithm [5]. It simply consists in running a Berlekamp/Massey algorithm on each of the $\left\lfloor \frac{n}{2T} \right\rfloor$ contiguous sub-sequences $x^{(i)} = (x_{2Ti}, \ldots, x_{2T(i+1) - 1})$ of the received word $x$. If $E < \left\lfloor \frac{n}{2T} \right\rfloor/2$ errors occurred, then the generator occurring with majority will be the correct one. We refer to [5] for further explanations on how to then recover the correct code-word using sequence clean-ups. Equivalently, this algorithm guaranties to find the unique code-word provided that $E$ errors occurred whenever $n \geq 2T(2E + 1)$. This decoding requires $n/2t$ execution of Berlekamp/Massey algorithm and List decoding. Following the same idea, one remarks that if $n \geq 2T(E + 1)$ then necessarily, one sub-sequence $x^{(i)}$ has to be clean of errors and the list of all $\left\lfloor \frac{n}{2T} \right\rfloor$ generators contains the correct one. This makes a trivial list decoding algorithm.
up to the minimal distance (see [5] for further details on how to recover the code-word using sequence clean-ups).

In order to further reduce the bound $n \geq 2T(E+1)$ (or equivalently increase the decoding radius above $\frac{n}{2T}$), we will study in Section 3 an alternative list decoding algorithm.

Beforehand, we want to address a common remark on the choice of the sub-vectors used for the unique and list decoding above.

Remark 1. Instead of partitioning the received word into $n/2T$ disjoint sub-vectors, one would hope to find more error-free sequences by considering all $n-2T$ sub-vectors of the form $(x_i, \ldots, x_{i+2T-1})$. This will very likely allow to decode more errors in many cases (as will be illustrated in Figure 2), but the worst case configuration (see proof of Theorem 3) remains unchanged. Note the the majority rule based unique decoding still works under the same conditions: at most $2TE$ sub-sequences will contain an error, hence at least $n-2T-2TE$ sequences are correct and form a majority as soon as $n-2T-2TE \geq (n-2T)/2$ which is $n \geq 2T(2E+1)$.

In terms of complexity, the number of arithmetic operations required for both unique and list decoding algorithms in [5] is $O(n^2)$ ($n/2T$ runs of Berlekamp/Massey algorithm on sequences of length $2T$, and $O(n/2T)$ calls to the sequence clean-up, each of which costs $O(nT)$). Now the above variant requires to inspect $n-2T$ sub-sequences instead of $n/2T$ and the complexity becomes $O(nT^2 + n^2T)$.

3 Affine sub-sequences

Consider a sequence $(a_0, \ldots, a_{n-1})$ of evaluations of a $t$-sparse polynomial $f(z) = \sum_{j=1}^{t} c_j z^{e_j}$, with $E$ errors. In our previous work, we used to search for sub-sequences of the form $(a_i, \ldots, a_{i+k-1})$ formed by $k$ consecutive elements that did not contain any error. If such a sequence could be found with $k = 2t$, then applying Ben-Or/Tiwari algorithm on it recovers the polynomial $f$ and make the decoding possible. We now propose to consider more sub-sequences of length $k$: all sub-sequences of the form

$$(a_r, a_{r+s}, a_{r+2s}, \ldots, a_{r+(k-1)s})$$

where $r + (k-1)s < n$,

that will be called affine sub-sequences. In the remain of the text, we will denote by $k$ the length of the sub-sequence. We will consider the general case where $k$ can be any positive integer, not necessarily even.

Lemma 1. If $\gcd(s, m) = 1$ and $k \geq 2E$, then such a sub-sequence with no error is sufficient to recover $f$.

Proof. Let $\beta = \alpha^s$ and $g(z) = f(z \alpha^r)$. Note that $\deg g = \deg f$ and $g$ is also $t$-sparse with the same monomial support as $f$. If $\gcd(s, m) = 1$ then $\text{order}(\beta) \geq m$. Then the sub-sequence $(a_r, a_{r+s}, a_{r+2s}, \ldots, a_{r+(k-1)s}) = (f(\alpha^r), f(\alpha^{r+s}), f(\alpha^{r+2s}), \ldots) = (g(\beta^0), g(\beta^1), g(\beta^2), \ldots)$ is formed by evaluations of $g$ in $k$ consecutive powers of an element $\beta$ of order greater than $m \geq \deg g$. One can therefore compute $g = \sum_{j=1}^{t} d_j z^{e_j}$ using Ben-Or/Tiwari algorithm on this sub-sequence. The coefficients of $f$ are directly deduced from that of $g$: $c_j = d_j \alpha^{-re_j}$. □
Example 1. Let \( t = 2 \), and consider a sequence of \( n = 9 \) evaluations \((a_0, a_1, \ldots, a_8)\). Then \( E = 1 \) is the maximal number of errors that the list decoding of [5] can decode as it requires that \( n \geq 2t(E + 1) \). Indeed if two errors occurred e.g. on elements \( a_3 \) and \( a_7 \), there is no contiguous sub-sequence of length \( 2t = 4 \) free of error, thus making the latter decoding fail. Now consider the sub-sequence \((a_0, a_2, a_4, a_6)\). It is free of error and is formed by evaluations of \( f(z) \) in the four consecutive powers of \( \beta = \alpha^2 \). Ben-Or/Tiwari algorithm applied on this sequence will reveal \( f \).

A list decoding algorithm.

This results in a new list decoding algorithm:

1. For each affine sub-sequence \((a_r, a_{r+s}, a_{r+s(k-1)})\) compute a generator \( \Lambda_{r,s} \) with the Berlekamp/Massey algorithm
2. (Optional heuristic reducing the list size) For each \( \Lambda_{r,s} \), run the sequence clean-up of [5] and discard it if it can not generate the sequence with less than \( E \) error, for some bound \( E \) on the number of errors.
3. For each remaining generator \( \Lambda_{r,s} \), apply Ben-Or/Tiwari algorithm to recover a corresponding sparse polynomial \( f_{r,s} \).
4. Return the list of \( f_{r,s} \).

There are \( n/k \) possible values for \( s \) and \( s_{\frac{n}{k}} = n/k \) for \( r \) when restricting the search to disjoint sub-sequences. Each run of Berlekamp/Massey algorithm costs \( O(k^2) \), so does the transpose Vandermonde system solver [23] in Ben-Or/Tiwari algorithm. The optional sequence clean-up heuristic adds a \( nk \) term. Overall, the complexity of amounts to the same \( O(n^2) \) operations, with an additional overhead of \( O(n^3/k) \) when the sequence clean-up heuristic is used. Applying Remark 1, \( r \) can now take \( n/s \) different values. The time complexity increases by a factor \( k : n^3 + n^2k \).

We implemented the affine sub-sequence search and computed its rate of success in finding a clean sequence for various values of \( E \) and \( k \). The error locations are uniformly distributed. We report in Figures 1 and 2 the average rate of success: for each value of the pair \((E, k)\), we display the rate success over 10000 samples. Figure 1 uses the search restricted to disjoint sequences, whereas Figure 2 shows the improvement brought by considering all sub-sequence as proposed in Remark 1. Again this improvement is important in practice, at the expense of a higher computational complexity for the decoder, but does not improve the worst case decoding radius.

4 Worst case decoding radius

We now focus the worst case analysis: finding estimates on the maximal decoding radius of the affine sub-sequence algorithm. More precisely, we want to determine for fixed \( E \) and \( t \), the smallest possible length \( n \), such that for any error vector of weight up to \( E \), there always exist at least one affine sub-sequence of length \( 2t \) with no error. This is stated in Problem 1 in the more general setting where the length \( k \) of the error free sequence need not be even.
Figure 1: Success rate for unique, standard list decoding and affine sub-sequence list decoding. Only disjoint sub-sequences are considered.

Figure 2: Success rate for unique, standard list decoding and affine sub-sequence list decoding. All sub-sequences are considered.

**Problem 1.** Given $k, E \in \mathbb{Z}_{>0}$, find the smallest $n \in \mathbb{Z}_{>0}$ such that for any sub-set $S \subset \{0, \ldots, n-1\}$ of $E$ distinct integers,

$$\exists r \in \mathbb{Z}_{\geq 0}, s \in \mathbb{Z}_{>0} \text{ with } r + s(k-1) < n \text{ s.t. } \{r, r + s, r + 2s, \ldots r + (k-1)s\} \cap S = \emptyset$$

(1)

In the following, the solution to this problem will be denoted by $n_{k,E}$. In some cases, the affine sub-sequence technique does not help improving the former bound $n \geq k(E + 1)$, not even by saving a single evaluation point.

**Example 2.** For $k = 5$ and $E = 3$, the worst case configuration (errors on $a_4, a_9$ and $a_{14}$) requires $n_{5,3} = 20 = k(E + 1)$ values to find $t$ consecutive clean values.

\[
\begin{array}{cccccccc}
  a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 & a_{10} & \ldots \\
\end{array}
\]
But for $E = 4$, one verifies that $n = 21$ suffices to ensure that a length 5 subsequence will always be found. In particular, in the previous configuration, placing the fourth error on $e_{19}$ leaves the subsequence $(a_0, a_5, a_{10}, a_{15}, a_{20})$ untouched.

We report in Table 1 and Figure 4 the solutions to Problem 1 for all typically small values of $E$ and $k$ computed by exhaustive search. We ran a Sage program for about 7 days on 24 cores of an Intel E5-4620 SMP machine.

---

*The code is available [http://membres-liglab.imag.fr/pernet/Depot/ldsic.sage](http://membres-liglab.imag.fr/pernet/Depot/ldsic.sage).
Table 1: Solutions to Problem 1 for the first values of $k$ and $E$

| $k$ | 2 3 4 5 6 7 8 9 10 11 12 13 14 15 |
|-----|-----------------------------------|
| 2   | 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 |
| 3   | 3 6 7 8 10 12 15 16 17 18 19 22 23 25 27 |
| 4   | 4 7 11 12 14 16 18 20 22 24 26 29 31 32 35 36 |
| 5   | 5 10 15 20 21 22 23 26 30 32 35 40 45 46 47 48 |
| 6   | 6 11 16 21 27 28 30 31 34 38 42 43 47 52 |
| 7   | 7 14 21 28 35 42 43 44 45 47 49 54 58 |
| 8   | 8 15 22 29 36 43 51 52 53 55 57 60 |
| 9   | 9 18 25 32 39 46 53 58 59 62 66 72 |
| 10  | 10 19 29 34 41 48 55 62 65 69 74 |
| 11  | 11 22 33 44 55 66 77 88 99 110 111 112 |
| 12  | 12 23 34 45 56 67 78 89 100 111 123 |
| 13  | 13 26 39 52 65 78 91 104 117 130 143 156 157 |

Figure 3: Solutions to Problem 1 for the first values of $k$ and $E$

In particular Figure 3 shows the improvement of the affine sub-sequence technique over the previous list decoding algorithm (which requires $n \geq k(E+1)$) for some even values of the sub-sequence length $k$.

These data show that the approach based on affine sub-sequences reduces in most cases the smallest value for $n$ required for decoding. The only cases with no improvement, as in Example 2, seem to happen when $k$ is prime and $E < k-1$: then $n = k(E+1)$ is the best value for $n$. We prove fact this in Lemma 2.

**Lemma 2.** If $k$ is prime and $E < k-1$, then the solution to Problem 1 is $n = k(E+1)$.

**Proof.** Splitting $\{0, \ldots, n-1\}$ into $E+1$ contiguous disjoint sets $V_i$ of $k$ elements shows that no sub-set of $E$ elements of $\{0, \ldots, n-1\}$ can intersect all of the $V_i$'s at the same time. Hence $n \leq (E+1)k$. We will show that no smaller value for $n$ is possible.

Suppose that $n < (E+1)k$ and that $\exists r \in \mathbb{Z}_{\geq 0}, s \in \mathbb{Z}_{> 0}$, such that $\{r, r+s, \ldots, r+(k-1)s\} \cap S = \emptyset$. We have $r+s(k-1) < (E+1)k-1$. If $s \geq E+2$, then $r+(E+2)(k-1) <
\((E+1)k - 1\) which implies \(r < (E+1) - k\) and \(r < 0\) which is a contradiction. Consequently \(s \leq E + 1 < k\). As \(k\) is prime and \(s < k\), the order of \(s\) modulo \(k\) is \(k - 1\). Therefore on one hand \(\{r \mod k, r + s \mod k, \ldots, r + (k-1)s \mod k\} = \{0, 1, \ldots, k-1\}\). On the other hand \(S' \mod k = \{k-1 \mod k, 2k-1 \mod k, \ldots, E(k-1) \mod k\}\) = \{\(k-1\)\}. Since \(\{r, r + s, \ldots, r + (k-1)s\} \cap S = \emptyset\), we have

\[
\exists j \leq k - 1 \text{ and } q \in \mathbb{Z}_{>0} \text{ s.t. } \begin{cases} r + js = k - 1 + qk \\ q + 1 > E \end{cases}
\]

Therefore \(r + js = (q+1)k - 1 \geq (E+1)k - 1\), but we also have \(r + js \leq n - 1 \leq (E+1)k - 2\) which is absurd. Consequently \(n \geq k(E + 1)\).

Note that the solution \(n_{k,E}\) is also strictly increase in both \(k\) and \(E\).

**Lemma 3.** For any \(k > 0, E \geq 0\),

(i) \(n_{k+1,E} > n_{k,E}\)

(ii) \(n_{k,E+1} > n_{k,E}\)

This implies that \(n_{k,E}\) is always comprised between the worst-case bounds \((n = k(E + 1))\) for the previous and next primes around \(k\) for small enough values of \(E\).

**Corollary 1.** Let \(p, q\) be two prime numbers such that \(p < k < q\). Then for any \(E < p - 1, p(E + 1) < n_{k,E} < q(E + 1)\).

Lastly Bertrand’s postulate \([4, 21]\) states that there is always a prime \(p\) between \(n\) and \(2n\).

**Corollary 2.** For any \(k > 0\),

(i) If \(E < \frac{k}{2}\), then \(\frac{k}{2}(E + 1) < n_{k,E}\)

(ii) If \(E < k\), then \(n_{k,E} < 2k(E + 1)\)

**Proof.** By Bertrand’s postulate there exist a prime \(p\) such that

\[
\left\lfloor \frac{k}{2} \right\rfloor < p < 2 \left\lceil \frac{k}{2} \right\rceil.
\]

Therefore \(\frac{k}{2}(E + 1) < p(E + 1)n_{p,E} \leq n_{k,E}\) if \(E < \frac{k}{2}\). Similarly, there exist a prime \(q\) such that \(k < q < 2k\), hence \(n_{k,E} < n_{q,E} = q(E + 1) < 2k(E + 1)\) if \(E < k\).

However, for larger value of \(E\), Figure 4 suggests that \(n_{E,k}\) increase at a much lower rate. We will now focus on the worst case asymptotic behavior of \(n_{E,k}\). In order to find a lower bound on the value of \(n\) that solves Problem 1, we consider the dual Problem 2.

**Problem 2.** Given \(k, E \in \mathbb{Z}_{>0}\), find the maximal \(n \in \mathbb{Z}_{>0}\) such that for any subset \(S \subset \{0, \ldots, n - 1\}\) of \(E\) distinct integers \((1)\) does not hold.
In the following, the solution to this problem will be denoted by \( N_{k,E} \). We will construct by induction a subset \( S \) producing a large value for \( n \) that result in a lower bound to the solution of Problem 2 and hence to that of Problem 1. It is a generalization of the worst case error vector of Lemma 2.

For all \( i \in \mathbb{Z}_{\geq 0} \) let \( n_i = k^i(k-1) \) and define the error vector \( v_i \) by the recurrence

\[
\begin{align*}
v_0 &= (0, \ldots, 0) \in K^{n_0} \\
v_i &= (v_{i-1}, 1, \ldots, 1, v_{i-1}, 1, \ldots, 1) \in K^{n_i}
\end{align*}
\]

Let \( E_i \) denote the number of non-zero elements in \( v_i \). It satisfies \( E_0 = 0 \) and \( E_i = (k-1)(E_{i-1} + k^{i-1}) \) which solves into \( E_i = k^{i+1} - k^i - (k-1)^{i+1} \).

**Lemma 4.** Let \( S \) be the support of \( v_i \). If \( k \) is prime, there is no \( r \in \mathbb{Z}_{\geq 0}, s \in \mathbb{Z}_{\geq 0} \) with \( r + s(k-1) < n_i \) such that \( \{r, r + s, r + 2s, \ldots, r + (k-1)s\} \cap S = \emptyset \).

**Proof.** Let \( r \in \mathbb{Z}_{\geq 0}, s \in \mathbb{Z}_{\geq 0} \) such that \( r + (k-1)s < n_i \) and define \( T = \{r, r + s, \ldots, r + (k-1)s\} \). Let \( \ell \) be the multiplicity of \( k \) in \( s \) (possibly zero) and define \( \alpha \) and \( \beta \) such that \( s = \alpha k^\ell + \beta k^{\ell+1} \) with \( 1 \leq \alpha < k \).

Let \( \tau, \mu, \nu \) be such that \( r = \tau + \mu k^\ell + \nu k^{\ell+1} \) with \( 0 \leq \tau < k^\ell \) and \( 0 \leq \mu < k \). As \( \gcd(\alpha, k) = 1 \) there exists \( 1 \leq j < k \) such that \( j\alpha = k-1-\mu \mod k \). Hence \( j\alpha + \mu = k-1+\lambda k \) for some \( \lambda \in \mathbb{Z} \). As \( j < k \), the set \( T \) contains the element \( x = r + js \) and we write

\[
x = r + js = \tau + (j\alpha + \mu)k^\ell + \nu k^{\ell+1} = \tau + (k-1)k^\ell + (\nu + \lambda)k^{\ell+1}.
\]

We now show that the element of index \( x \) in \( v_i \) is a one. In this last expression, the term \( (\nu + \lambda)k^{\ell+1} \) indicates that \( x \) is located in the \( \nu + \lambda + 1 \)-st block of the form \( (v_\ell, 1, \ldots, 1) \). Then the term \( (k-1)k^\ell = n_\ell \) is precisely the dimension of \( v_\ell \). Lastly, as \( \tau < k^\ell \), we deduce that the element of index \( x \) is a 1 in \( v_i \).

**Lemma 5.** If \( k \) is prime, then \( N_{k,E} = n_i \).

**Proof.** Lemma 4 states that \( N_{k,E} \geq n_i \). There remain to show that \( N_{k,E} \leq n_i \), or equivalently that there is no subset of \( E_i \) elements of \( \{0, \ldots, n_i\} \) that intersects all affine sub-sequences of length \( k \). We will prove it by induction on \( i \). Consider the induction hypothesis:

\[
H_i : \forall j \leq i \quad \begin{cases}
N_{k,E_j} = n_j \\
\forall S \subset \{0, \ldots, n_i - 1\} \text{ verifying (1)} \\
S \cap \{0, k^j, 2k^j, \ldots, (k-2)k^j\} = \emptyset.
\end{cases}
\]

For \( i = 0, E_0 = 0 \), the solution \( N_{k,0} = k-1 \) is obvious.

For a given \( i \), suppose \( H_i \) is true. Suppose \( N_{k,E_{i+1}} > n_{i+1} \). Then there exists a subset \( S \subset \{0, \ldots, n_{i+1}\} \) verifying (1). Then \( \forall p \in \{0, k-2\} \ | S \cap \{pk^i, \ldots, (p+1)k^i - 1\} | = E_i \), otherwise \( H_i \) would not be satisfied. This means that all elements in \( S \) are smaller than \( n_{i+1} \). Moreover, \( S \cap \{0, k^i, \ldots, (k-2)k^i\} = \emptyset \) by induction hypothesis. Finally, the affine sub-sequence of length \( k \) \( (0, k^i, 2k^i, \ldots, (k-1)k^i) \) does not intersect \( S \) which is absurd. \( \square \)
As \( n_{k,i} = N_{k,i} + 1 \), we obtain the Corollary 1.

**Corollary 1.** If \( k \) is prime, then \( n_{k,i} = n_i + 1 \).

We can now estimate the asymptotic behavior of \( n_{k,E} \) with respect to \( k \) and \( E \). More precisely, we will express the maximal number of errors \( E_i \) that can be decoded as a function of \( n_i \) and \( k \).

**Lemma 6.** If \( k \) is prime, then for all \( i \geq 0 \),

\[
n_i \left( 1 - \left( \frac{k-1}{n_i} \right)^{\frac{1}{k^{1/k}}} \right) \leq E_i \leq \frac{n_i}{k} \log_k \frac{n_i}{k-1}.
\]

**Proof.** As the function \( f(x) = x^i \) is convex, we have \( f(k) - f(k-1) \leq f'(k) \). Hence

\[
E_i = (k-1)(k^i - (k-1)^i) \leq (k-1)i k^{i-1} \leq i \frac{n_i}{k}.
\]

Now \( \log_k \frac{n_i}{k-1} = \log_k k^i = i \). Hence \( E_i \leq \frac{n_i}{k} \log_k \frac{n_i}{k-1} \).

Consider now

\[
\frac{E_i}{n_i} = \frac{(k-1)(k^i - (k-1)^i)}{(k-1)k^i} = 1 - \left( 1 - \frac{1}{k} \right)^i
\]

\[
= 1 - e^{i \ln(1 - \frac{1}{k})} \geq 1 - e^{-\frac{1}{k}}.
\]

Again using the fact that \( i = \log_k \frac{n_i}{k-1} \), we get

\[
E_i \geq n_i \left( 1 - \left( \frac{k-1}{n_i} \right)^{\frac{1}{k^{1/k}}} \right).
\]

For any given \( k \) and \( n \), we denote by \( E_{n,k} \) the decoding radius of the affine sub-sequence algorithm, that is, the maximal number of errors for which the algorithm will still return a list containing the correct answer.

**Lemma 7.** \( \forall n > k > 0 \ E_{n-1,k} \leq E_{n,k} \leq E_{n,k-1} \).

**Corollary 2.** \( E_{n,k} \geq \frac{n}{2k} \left( 1 - \left( \frac{4k^2}{n} \right)^{\frac{1}{k^{1/k}}} \right) \).

**Proof.** Again from Bertrand’s postulate, there exists a prime \( p \) such that \( k < p < 2k \). Let \( i \) be such that \( n_i = (p - 1)p^i < n \leq (p - 1)p^{i+1} = n_{i+1} \). Then

\[
E_{n,k} \geq E_{n,p} \geq E_{n_i,p} \geq n_i \left( 1 - \left( \frac{p-1}{n_i} \right)^{\frac{1}{p^{1/p}}} \right)
\]

\[
\geq \frac{n}{2k} \left( 1 - \left( \frac{4k(k-1)}{n} \right)^{\frac{1}{k^{1/k}}} \right).
\]
This result shows a lower bound on the asymptotic improvement of the affine sub-sequence list decoding (to be compared with the bound $E \geq \frac{n}{k} - 1$ of the list decoding of [5]). This bound is pessimistic, as the actual values for $E$, computed from Table 1 are much higher. Figure 5 draws these values for $k = 3, 5, 6$ and the upper and lower bounds of Lemma 6. The upper bound in Lemma 6 states that the for some cases, the improvement in the decoding radius will never exceed a logarithmic factor in $\frac{n}{k}$. In Figure 5, these points in-between the two bounds can be seen for $k = 3$, $E = 2, 10$ or $k = 5$, $E = 4$.

Figure 5: Decoding radius for the affine sub-sequence algorithm: comparison of upper, lower bounds and actual value

5 Positive characteristic

In this last section we consider the case where the base field has characteristic zero. We show that some choices of evaluation points allow to reach much better minimal distances for sparse polynomial evaluation codes.

Positive real evaluation points.
Theorem 4. Consider $n$ distinct positive real numbers $\xi_0, \ldots, \xi_{n-1} > 0$. The sparse polynomial evaluation code defined by

$$\mathcal{C}(n, T) = \{(f(\xi_0), \ldots, f(\xi_{n-1})) : f \in \mathbb{R}[X] \text{ is } t\text{-sparse}$$

with $t \leq T$$

has minimal distance $\delta = n - 2T + 1$.

Proof. Consider two code words $(f(\xi_0), \ldots, f(\xi_{n-1}))$ and $(g(\xi_0), \ldots, g(\xi_{n-1}))$ for a $t_f$-sparse polynomial $f$ and a $t_g$-sparse polynomial $g$, with $t_f, t_g \leq T$, at Hamming distance $\leq n - 2T$.

Then the polynomial $f - g$ has sparsity $\leq 2T$, and vanishes in least $2T$ distinct positive reals $\xi_i$. By Descartes’s rule of sign $f - g = 0$.

Corollary 3. Suppose we have, for a $t_f \leq T$ sparse real polynomial $f(x)$, values $f(\xi_i)$ for $2T + 2E$ distinct positive real numbers $\xi_i > 0$, where $e \leq E$ of those values can be erroneous $f(\xi_i) + \epsilon_i$. If a $t_g \leq T$ sparse real polynomial $g$ interpolates any $2T + E$ of the $f(\xi_i) + \epsilon_i$, then $g = f$.

So $f$ can be uniquely recovered from $2T + 2E$ values with $e \leq E$ errors.

Sampling primitive elements of co-prime orders in the complex unit circle.

Theorem 5. Let $T, D$, and $n \geq k = 2T \log(D) / \log(2T)$ be given. Consider $n$ $p_i$-th roots of unity $\xi_i \neq 1$, where $2T < p_0 < p_2 < \cdots < p_{n-1}$, $p_i$ prime. The sparse polynomial evaluation code defined by

$$\mathcal{C}(n, T) = \{(f(\xi_0), \ldots, f(\xi_{n-1})) : f \in \mathbb{Q}[X] \text{ is } t\text{-sparse}$$

with $t \leq T$$

has minimal distance $\delta = n - k + 1 = n - 2T \log(D) / \log(2T) + 1$.

Proof. Consider two code words $(f(\xi_0), \ldots, f(\xi_{n-1}))$ and $(g(\xi_0), \ldots, g(\xi_{n-1}))$ for a $t_f$-sparse polynomial $f$ and a $t_g$-sparse polynomial $g$, with $t_f, t_g \leq T$, at Hamming distance $\leq n - k$. Then $(f - g)(\xi_j)$ vanishes for at least $k$ of the $\xi_i$, say for those sub-scripted $j \in J$.

Let $0 \leq e_1 < e_2 < \cdots < e_t$ be the term exponents in $f - g$, with $t \leq 2T$. Suppose $f - g \neq 0$. Consider $M = (e_t - e_1)(e_t - e_2) \cdots (e_t - e_{t-1})$. Since $M \leq 2^{2T}$ and $\prod_{j \in J} p_j > (2T)^k \geq D^{2T}$, not all $p_j$ for $j \in J$ can divide $M$. Let $\ell \in J$ with $M \not\equiv 0 \pmod{p_\ell}$. Then the term $x^{e_\ell} \mod{p_\ell}$ is isolated in $h(x) = (f(x) - g(x) \mod (x^{p_\ell} - 1))$, and therefore the polynomial $h(x)$ is not zero; $h$ has at most $2T$ terms, and $h(\xi) = 0$. This means that $h(x)$ and $\Psi_\ell(x) = 1 + x + \cdots + x^{p_{\ell-1}}$ have a common GCD. Because $\Psi_\ell$ is irreducible over $\mathbb{Q}$, and since $\deg(h) \leq p_\ell - 1$, that GCD is $\Psi_\ell$. So $h$ is a scalar multiple of $\Psi_\ell$ and has $p_\ell > 2T$ non-zero terms, a contradiction.

Corollary 4. Let $T, D, E$ be given and let the integer $k \geq 2T \log(D) / \log(2T)$. Suppose we have, for a $t_f$-sparse polynomial $f \in \mathbb{Q}[x]$, where $t_f \leq T$ and $\deg(f) \leq D$, the values $f(\xi_i)$ for $k + 2E p_i$-th roots of unity $\xi_i \neq 1$, where $2T < p_1 < p_2 < \cdots < p_{N+2E}$, $p_i$ prime. Again $e \leq E$ of those values can be erroneous $f(\xi_i) + \epsilon_i$. If a $t_g$-sparse polynomial $g \in \mathbb{Q}[x]$ with $t_g \leq T$ and $\deg(g) \leq D$ interpolates any $k + E$ of the $f(\xi_i) + \epsilon_i$, then $g = f$. 

14
6 Future Work

The surprisingly large minimal distance found in characteristic zero motivates the search for symbolic/numeric decoding algorithms reaching this decoding radius. A next step is also to find an analogous of theorem 5 over finite fields, using sampling points in extensions defined by an irreducible polynomial with high Hamming weight.

The study of the decoding capacity of interleaved sparse polynomial interpolation codes is also of interest. It should improve the decoding radius and has direct applications to the recovery of vectors of sparse polynomials.

7 Acknowledgments

We would like to thank Bruno Salvy for his helpful remarks and suggestions.

References

[1] Ben-Or, M., and Tiwari, P. A deterministic algorithm for sparse multivariate polynomial interpolation. In Proc. 20th Annual ACM Symp. Theory Comput. (1988), pp. 301–309.

[2] Blahut, R. E. Theory and Practice of Error Control Codes. Addison Wesley, Reading, 1983.

[3] Candes, E., and Tao, T. Near-optimal signal recovery from random projections: Universal encoding strategies? Information Theory, IEEE Transactions on 52, 12 (2006), 5406–5425.

[4] Chebyshev, P. L. Mémoire sur les nombres premiers. J. de Mathématiques Pures et Appliquées 17 (1852), 366–390.

[5] Comer, M. T., Kaltofen, E. L., and Pernet, C. Sparse polynomial interpolation and Berlekamp/Massey algorithms that correct outlier errors in input values. In Proc. ISSAC ’12 (juls 2012), pp. 138–145.

[6] Du, P., Bouteiller, A., Bosilca, G., Herault, T., and Dongarra, J. Algorithm-based fault tolerance for dense matrix factorizations. In PPoPP’12 (New York, NY, USA, 2012), ACM, pp. 225–234.

[7] Garg, S., and Schost, Éric. Interpolation of polynomials given by straight-line programs. Theoretical Comput. Sci. 410, 27-29 (2009), 2659 – 2662.

[8] Giesbrecht, M., Labahn, G., and Lee, W. Symbolic-numeric sparse interpolation of multivariate polynomials. J. Symbolic Comput. 44 (2009), 943–959.

[9] Giesbrecht, M., and Roche, D. S. Interpolation of shifted-lacunary polynomials. Computational Complexity 19, 3 (Sept. 2010), 333–354.
[10] Grigoriev, D. Y., and Karpinski, M. A zero-test and an interpolation algorithm for the shifted sparse polynomials. In Proc. AAECC-10 (1993), vol. 673 of Lect. Notes Comput. Sci., Springer Verlag, pp. 162–169.

[11] Huang, K.-H., and Abraham, J. A. Algorithm-based fault tolerance for matrix operations. IEEE Trans. Comput. 33, 6 (June 1984), 518–528.

[12] Kaltofen, E., Lakshman Y. N., and Wiley, J. M. Modular rational sparse multivariate polynomial interpolation. In Proc. ISSAC’90 (1990), S. Watanabe and M. Nagata, Eds., ACM Press, pp. 135–139. URL: http://www.math.ncsu.edu/~kaltofen/bibliography/90/KLW90.pdf.

[13] Kaltofen, E., and Lee, W. Early termination in sparse interpolation algorithms. J. Symbolic Comput. 36, 3–4 (2003), 365–400. URL: http://www.math.ncsu.edu/~kaltofen/bibliography/03/KL03.pdf.

[14] Kaltofen, E., and Yang, Z. Sparse multivariate function recovery from values with noise and outlier errors. In Proc. ISSAC’13 (2013), pp. 219–226. URL: http://www.math.ncsu.edu/~kaltofen/bibliography/13/KaYa13.pdf.

[15] Khonji, M., Pernet, C., Roch, J.-L., Roche, T., and Stalinsky, T. Output-sensitive decoding for redundant residue systems. In Proc. ISSAC’10 (July 2010), pp. 265–272.

[16] Massey, J., and Schaub, T. Linear complexity in coding theory. In Coding Theory and Applications, G. Cohen and P. Godlewski, Eds., vol. 311 of Lecture Notes in Computer Science. Springer Verlag, 1988, pp. 19–32.

[17] Massey, J. L. Shift-register synthesis and BCH decoding. IEEE Trans. Inf. Theory IT-15 (1969), 122–127.

[18] Meidl, W., and Niederreiter, H. Linear complexity, k-error linear complexity, and the discrete fourier transform. Journal of Complexity 18, 1 (2002), 87 – 103.

[19] Moon, T. K. Error correction coding: mathematical methods and algorithms. Wiley-Interscience, 2005.

[20] Prony, R. Essai expérimental et analytique sur les lois de la Dilatabilité de fluides élastique et sur celles de la Force expansive de la vapeur de l’eau et de la vapeur de l’alkool, à différentes températures. J. de l’École Polytechnique 1 (Floréal et Prairial III (1795)), 24–76.

[21] Ramanujan, S. A proof of Bertrand’s postulate. J. of the Indian Mathematical Society 11 (1919), 181–182.

[22] Reed, I. S., and Solomon, G. S. Polynomial codes over certain finite fields. J. SIAM 8, 2 (June 1960), 300–304.

[23] Zippel, R. Interpolating polynomials from their values. J. Symbolic Comput. 9, 3 (1990), 375–403.