Sparse Polynomial Interpolation Codes and their Decoding Beyond Half the Minimum 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/

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 code-word 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 an arithmetic 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 ≤ 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 code-word length minus twice the sparsity.

Categories 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: sparse polynomial interpolation, Blahut's algorithm, Prony's algorithm, exact polynomial fitting with errors.

*This material is based on work supported in part by the National Science Foundation under Grant CCF-1115772 (Kaltofen), and the Agence Nationale de la Recherche under Grant HPAC ANR-11-BS02-013 and the Inria Associate Teams Grant QOLAPS (Pernet).

ISSAC'14, July 23–25, 2014, Kobe, Japan. ACM 978-1-4503-2501-1/14/07 ...$15.00. http://dx.doi.org/10.1145/2608628.2608660 .

1. INTRODUCTION

Evaluation-interpolation schemes are a key ingredient in many of today's 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 [4], and sparse interpolation of polynomials [23, 1, 15, 13, 9, 11]. Most algorithms for the latter problem rely on the connection between linear complexity and sparsity, often referred to as Blahut's Theorem (Theorem 1 [3, 19]) though already used in the 18th century by Prony [23]. The Berlekamp/Massey algorithm [20] makes this connection effective. These exact sparse interpolation techniques have been very successfully applied to numeric computations [10, 16, 6, 17].

Computer algebra also widely uses evaluation-interpolation schemes 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 [25, 22]. There, 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 [3, 19] 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 [14, 18, 7]. In particular, paral-

lelization based on evaluation-interpolation can be made fault tolerant if interpolation with errors is performed. This is achieved by Reed-Solomon codes for dense polynomial interpolation and by CRT codes, for residue number systems [18]. The problem of sparse polynomial interpolation with errors rises naturally in this context. We give algorithms for the solution of the problem in [6]. Our approach is naturally related to the k-error linear complexity problem [21] 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 investigate this problem from a coding theory viewpoint. In section 2 we define the sparse polynomial interpolation codes. We then focus on the case where the evaluation points are consecutive powers of a primitive root of unity, whose order is divisible by twice the sparsity, in order to to benefit from Blahut/Ben-Or/Tiwari interpolation algorithm. We show that in this stetting the minimal distance is precisely the length divided by twice the sparsity. The algorithms of [6] 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 an arithmetic progression, reaching a larger decoding 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 make connections between the asymptotic decoding capacity and the famous Erdős-Turán problem of additive combinatorics. 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 linearly recurring if there exists $\lambda_0, \lambda_1, \ldots, \lambda_{t-1}$ such that $a_{j+t} = \sum_{i=0}^{t-1} \lambda_i a_{j+i}\ \forall j \geq 0$. The monic polynomial $\Lambda(z) = z^t - \sum_{i=0}^{t-1} \lambda_i z^i$ is called a generating polynomial of the sequence, the generating polynomial with least degree is called the minimal generating polynomial and its degree is the linear complexity of the sequence.

These definitions can be extended to vectors, viewed as 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 $a_{j+t} = \sum_{i=0}^{t-1} \lambda_i a_{i+j}\ \forall 0 \leq j \leq n-t-1$. Note that consequently, any vector is linearly recurring with linear complexity less than n.

Theorem 1 (Blahut [3, 19]) *Let K be a field containing an N-th primitive root of unity. The linear complexity of an N-periodic sequence $A = (a_0, \ldots, a_{N-1}, a_0, \ldots)$ over K is equal to the Hamming weight of the discrete Fourier transform of $(a_0, \ldots, a_{N-1})$.*

The Blahut/Ben-Or/Tiwari Algorithm. We review the Blahut/Ben-Or/Tiwari [2, 1] algorithm in the setting of univariate sparse polynomial interpolation. Let f be a univariate polynomial with t terms, m_j and let c_j the corresponding non-zero coefficients:

$$f(x) = \sum_{j=1}^{t} c_j x^{e_j} = \sum_{j=1}^{t} c_j m_j \neq 0, e_j \in \mathbb{Z}.$$

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

The Blahut/Ben-Or/Tiwari algorithm then proceeds in the four following steps:

1. Find the minimal-degree generating polynomial Λ for $(a_0, a_1, \ldots)$, using the Berlekamp/Massey algorithm.
2. Compute the roots b_j of Λ, using univariate polynomial factorization.
3. Recover the exponents e_j of f, by repeatedly dividing b_j by α.
4. Recover the coefficients c_j of f, by solving the transposed $t \times t$ Vandermonde system

$$\begin{bmatrix} 1 & 1 & \ldots & 1 \\ b_1 & b_2 & \ldots & b_t \\ \vdots & \vdots & \ddots & \vdots \\ b_1^{t-1} & b_2^{t-1} & \ldots & b_t^{t-1} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_t \end{bmatrix} = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{t-1} \end{bmatrix}.$$

By Blahut's theorem, the sequence $(a_i)_{i \geq 0}$ has linear complexity t, hence only $2t$ coefficients suffice for the Berlekamp/Massey algorithm to recover the minimal polynomial Λ. In the presence of errors in some of the evaluations, this fails.

2. SPARSE INTERPOLATION CODES

Definition 1 *Let K be a field, $0 < n \leq m$, two integers and let $x_0, \ldots, x_{n-1}$ be n distinct elements of K. A sparse polynomial evaluation code of parameters (n, T) over K is defined as the set*

$$\mathcal{C}(n,T) = \{(f(x_0), f(x_1), \ldots, f(x_{n-1})) : f \in \mathsf{K}[z] \text{ is } t\text{-sparse with } t \leq T \text{ and } \deg f < m\}$$

In order to benefit from Blahut/Ben-Or/Tiwari algorithm for error free interpolation, we will consider, until section 5, the special case where the evaluation points are consecutive powers of a primitive m-th root of unity $\alpha \in \mathsf{K}$: $x_i = \alpha^i$. In this context, we can state the minimum distance of such codes provided that $2T$ divides m.

Theorem 3 *If $\alpha \in \mathsf{K}$ is a primitive m-th root of unity, $x_i = \alpha^i$, $i \in \{0, \ldots, n-1\}$ and $2T$ divides m, then the corresponding (n, T)-sparse polynomial evaluation code has minimum distance $\delta = \lfloor \frac{n}{2T} \rfloor$.*

The following proof is adapted from [6, §2.1].

Proof. Let $\bar{0}$ denote the zero vector of length $T-1$. Consider two infinite sequences :

$$\begin{aligned} x &= (\bar{0}, 1, \bar{0}, 1, \ldots) \\ y &= (\bar{0}, 1, \bar{0}, -1, \bar{0}, 1, \bar{0}, -1, \ldots) \end{aligned}$$

formed by the repetition of their first $2T$ values and the corresponding vectors $x^{(n)}, y^{(n)} \in \mathsf{K}^n$ and $x^{(m)}, y^{(m)} \in \mathsf{K}^m$ formed by respectively the first n and first m values of these sequences. The sequence x is generated by $z^T - 1$

and y by z^T+1, both are m periodic as $2T$ divides m. Lastly, let $\hat{x}^{(m)} = \mathrm{DFT}_\alpha^{-1}(x^{(m)}) = \frac{1}{m}\mathrm{DFT}_{\alpha^{-1}}(x^{(m)})$. From Blahut's theorem, $\hat{x}^{(m)}$ has Hamming weight T. By identification between K^m and $\mathsf{K}[z]_{<m}$, $\hat{x}^{(m)}$ corresponds to a polynomial f_x of degree less than m and sparsity T. Hence $x^{(n)} = (f_x(\alpha^0), f_x(\alpha^1), \ldots, f_x(\alpha^{n-1}))$ is a code word of an (n,T)-sparse evaluation code. Similarly $y^{(n)}$ is also a code-word. More precisely one verifies that

$$f_x(z) = \frac{1}{T}\sum_{i=0}^{T-1} z^{2i\frac{m}{2T}} = \frac{1}{T}\frac{z^m-1}{z^{\frac{m}{T}}-1},$$

$$f_y(z) = \frac{-1}{T}\sum_{i=0}^{T-1} z^{(2i+1)\frac{m}{2T}} = \frac{-z^{\frac{m}{2T}}}{T}\frac{z^m-1}{z^{\frac{m}{T}}-1}.$$

Since $x^{(n)}$ and $y^{(n)}$ differ by exactly $\lfloor \frac{n}{2T} \rfloor$ values, this is an upper bound on the minimum distance δ.

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

$$\begin{array}{ll} x^{(1)} = (x_1, \ldots, x_{2T}), & y^{(1)} = (y_1, \ldots, y_{2T}) \\ x^{(2)} = (x_{2T+1}, \ldots, x_{4T}), & y^{(2)} = (y_{2T+1}, \ldots, y_{4T}) \\ \vdots & \vdots \\ x^{(\lfloor \frac{n}{2T} \rfloor)}, & y^{(\lfloor \frac{n}{2T} \rfloor)} \end{array}$$

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 $\lfloor \frac{n}{2T} \rfloor$ positions, and consequently $\delta = \lfloor \frac{n}{2T} \rfloor$. □

Unique decoding. There exists an algorithm that does unique decoding of such codes up to half the minimum distance: the Majority Rule Berlekamp/Massey algorithm [6]. It simply consists in running a Berlekamp/Massey algorithm on each of the $\lfloor \frac{n}{2T} \rfloor$ contiguous sub-sequences $x^{(i)} = (x_{2Ti}, \ldots, x_{2T(i+1)-1})$ of the received word x. If $E < \lfloor \frac{n}{2T} \rfloor/2$ errors occurred, then the generator occurring with majority will be the correct one. We refer to [6] 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 occured whenever $n \geq 2T(2E+1)$. This decoding requires $\lfloor n/2T \rfloor$ executions of Berlekamp/Massey algorithm.

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 $\lfloor \frac{n}{2T} \rfloor$ generators contains the correct one. This makes a trivial list decoding algorithm up to the minimum distance (see [6] 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 $n-2T+1$ 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 that the majority rule based unique decoding still works under the same conditions: at most $2TE$ sub-sequences will contain an error, hence a majority of subsequences will be correct as soon as $4TE < n-2T+1$, 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 [6] 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 $O(n^2T)$ (as $T = o(n)$).

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 Blahut/Ben-Or/Tiwari algorithm on it recovers the polynomial f and makes the decoding possible. We now propose to consider all length k sub-sequences in arithmetic progression:

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

that will be called affine index sub-sequences or more conveniently affine sub-sequences. In the remaining of the text, k will denote 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 $k \geq 2t$, 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 $\mathrm{order}(\beta) \geq m$. Then the sub-sequence $(a_r, a_{r+s}, a_{r+2s}, \ldots, a_{r+(k-1)s}) = (f(\alpha^r), f(\alpha^{r+s}), f(\alpha^{r+2s}), \ldots, f(\alpha^{r+(k-1)s})) = (g(\beta^0), g(\beta^1), g(\beta^2), \ldots, g(\beta^{k-1}))$ is formed by evaluations of g in k consecutive powers of an element β of order greater than $m \geq \deg g$. One can thus compute $g = \sum_{j=1}^t d_j z^{e_j}$ using Blahut/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 [6] 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 $\beta = \alpha^2$. Blahut/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 $(a_r, a_{r+s}, \ldots, a_{r+s(k-1)})$ 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 [6] and discard it if it can not generate the sequence with less than E errors, for some bound E on the number of errors.
3. For each remaining generator $\Lambda_{r,s}$, apply Blahut/Ben-Or/Tiwari algorithm to recover the associated sparse polynomial $f_{r,s}$.
4. Return the list of the $f_{r,s}$.

A first approach is to explore all sub-sequences for any value of $s \in \{1 \ldots \lfloor n/k \rfloor\}$ and $r \in \{0 \ldots n-(k-1)s-1\}$. This amounts to $O(n^2/k)$ sub-sequences. A second approach, applying Remark 1 considers all values for $s \in \{1 \ldots n/k\}$ but then for each s only considers the disjoint sub-sequences with $s\frac{n}{ks} = n/k$ choices for r. This amounts to $O(n^2/k^2)$ sub-sequences. For each sub-sequence, corresponding to a pair (s, r), Blahut/Ben-Or/Tiwari algorithm is run in $O(k^2)$ (Berlekamp/Massey algorithm and solving the transpose Vandermonde system [30]). The optional sequence clean-up heuristic adds an $O(nk)$ term. Overall, the complexity of the second approach amounts to the same $O(n^2)$ estimate, as the list decoding of [6]. The additional overhead of $O(n^3/k)$ when the sequence clean-up heuristic is used also remains identical. In the first approach, ignoring Remark 1, these complexity estimates are multiplied by a factor k.

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 over 10 000 samples for each value of the pair (E, k). Figure 1 uses the search restricted to disjoint sequences. whereas Figure 2 shows the improvement brought by considering all sub-sequences 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 unique decoding radius in the worst case.

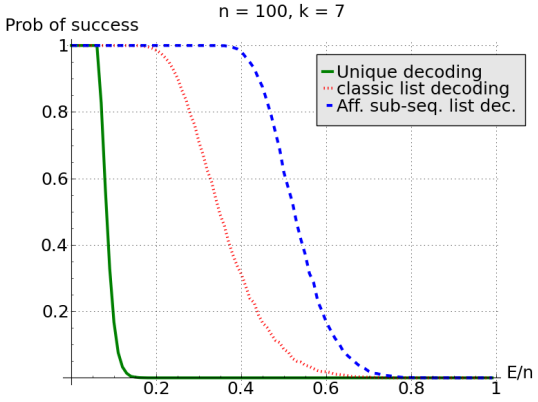

Figure 1: Success rate for unique, standard list decoding and affine sub-sequence list decoding. Only disjoint sub-sequences are considered.

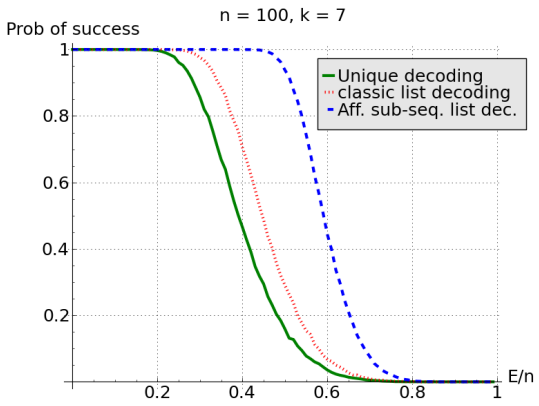

Figure 2: Success rate for unique, standard list decoding and affine sub-sequence list decoding. All sub-sequences are considered.

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.

Problem 1 *Given $k, E \in \mathbb{Z}_{>0}$, find the smallest $n \in \mathbb{Z}_{>0}$ such that for all subsets $S \subset \{0, \ldots, n-1\}$ with E elements, that is, $|S| = E$,*

$$\exists r \in \mathbb{Z}_{\geq 0}, \exists s \in \mathbb{Z}_{>0} \text{ with } r + s(k-1) \leq n-1 : \\ \forall i \text{ with } 0 \leq i \leq k-1 : r + is \notin S. \quad (1)$$

We will denote by $n_{k,E}$ the minimum solution to Problem 1.

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 k consecutive clean values.

a_0	a_1	a_2	a_3	$\mathbf{a_4}$	a_5	a_6	a_7	a_8	$\mathbf{a_9}$	…	$\mathbf{a_{14}}$	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}			

a_0	a_2	$\mathbf{a_4}$	a_6	a_8	a_{10}	a_{12}	$\mathbf{a_{14}}$	a_{16}	a_{18}
a_1	a_3	a_5	a_7	$\mathbf{a_9}$	a_{11}	a_{13}	a_{15}	a_{17}	a_{19}

a_0	a_3	a_6	$\mathbf{a_9}$	a_{12}	a_{15}	a_{18}
a_1	$\mathbf{a_4}$	a_7	a_{10}	a_{13}	a_{16}	a_{19}
a_2	a_5	a_8	a_{11}	$\mathbf{a_{14}}$	a_{17}	

a_0	$\mathbf{a_4}$	a_8	a_{12}	a_{16}	
a_1	a_5	$\mathbf{a_9}$	a_{13}	a_{17}	
a_2	a_6	a_{10}	$\mathbf{a_{14}}$	a_{18}	
a_3	a_7	a_{11}	a_{15}	a_{19}	

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.

a_0	a_5	a_{10}	a_{15}	a_{20}
a_1	a_6	a_{11}	a_{16}	
a_2	a_7	a_{12}	a_{17}	
a_3	a_8	a_{13}	a_{18}	
$\mathbf{a_4}$	$\mathbf{a_9}$	$\mathbf{a_{14}}$	$\mathbf{a_{19}}$	

We report in Table 1 and Figure 3 the values of $n_{k,E}$ 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.

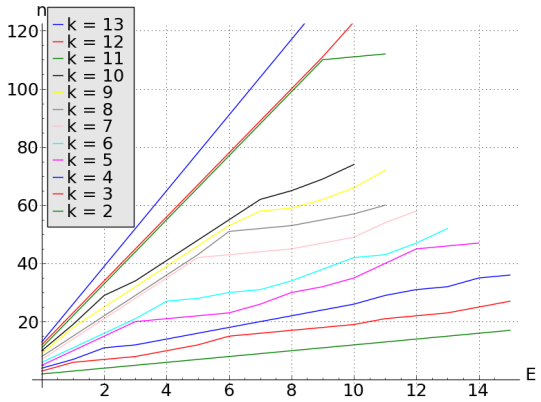

Figure 3: Solutions to Problem 1 for the first values of k and E

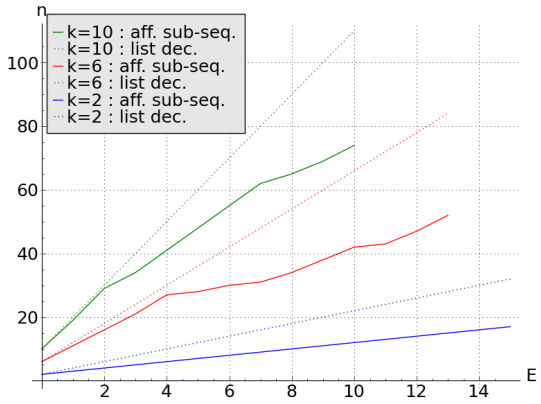

Figure 4: Improvement of the affine subsequence approach over the list decoding of [6]

In particular Figure 4 shows the improvement of the affine sub-sequence technique over the previous list decoding algorithm, requiring $n = k(E+1)$, for some even values of the sub-sequence length k. These data indicate that the optimal value for the length n is improved in almost any case except when k is prime and $E < k-1$, as in Example 2. Lemma 2 states more precisely at which condition the new algorithm does not improve the value $n = k(E+1)$ of the former list decoding.

*The code is available http://membres-liglab.imag.fr/pernet/Depot/ldsic.sage.

Lemma 2 *We have for the minimum solution $n_{k,E}$ of Problem 1: $n_{k,E} \leq k(E+1)$, with equality $n_{k,E}=k(E+1)$ if and only if $E+2 \leq g$, where g denotes the smallest prime factor of k.*

In particular, this implies that for even $k = 2t$, the new list decoder performs always better.

Proof. Let $n = k(E+1)$. Splitting $\{0, \ldots, n-1\}$ into $E+1$ contiguous disjoint sets V_i of k elements shows that no subset of E elements of $\{0, \ldots, n-1\}$ can intersect all of the V_i's at the same time. Hence $n_{k,E} \leq (E+1)k$.

We will denote by $P_{r,s}$ the arithmetic progression $\{r, r+s, \ldots, r+(k-1)s\}$.

Suppose $n = k(E+1) = n_{k,E}$. Then there is a subset S of E elements of $\{0, \ldots, n-2\}$ that intersects all $P_{r,s} \subset \{0, \ldots, n-2\}$. We will show that $S = \{k-1, 2k-1, \ldots, Ek-1\}$. Indeed, as the E segments $P_{ik,1}$ for $0 \leq i \leq E-1$ are disjoint, each of them must contain exactly one element of S. Hence, $\{Ek, \ldots, Ek+k-2\} \cap S = \emptyset$, and therefore $Ek-1 \in S$, otherwise $P_{Ek-1,1} \cap S = \emptyset$. By the same argument, we deduce iteratively that $ik-1 \in S$ for all $i \leq E-1$. It follows that $E+1 < g$, otherwise $n \geq kg$ and $P_{0,g} \subset \{0, \ldots, n-2\}$ but $P_{0,g} \cap S = \emptyset$ since $P_{0,g} \bmod g = \{0\}$ and $S \bmod g = \{k-1\}$.

Now suppose $E+1 < g$. We will show that $S = \{k-1, 2k-1, \ldots, Ek-1\}$ intersects all $P_{r,s} \subset \{0, 1, \ldots, k(E+1)-2\}$ from which we shall deduce that $n_{k,E} = k(E+1)$.

First note that $s < g$: otherwise $r + g(k-1) \leq r + s(k-1) \leq k(E+1) - 2 \leq k(g-1) - 2$ would imply $r+k \leq g-2$, which is absurd. Hence s is co-prime with k. Therefore $P_{r,s} \bmod k = \{0, 1, \ldots, k-1\}$, and $\exists j \leq k-1$ and $q \in \mathbb{Z}_{\geq 0}$ such that $r+js = k-1+qk = (q+1)k-1$. As $r+js \leq (E+1)k-2$ we have $q < E$. Hence $r+js \in S$ and finally $P_{r,s} \cap S \neq \emptyset$. □

Note that the solution $n_{k,E}$ is also strictly increasing in both k and E: $\forall k > 0, E \geq 0 \begin{cases} n_{k+1,E} > n_{k,E} \\ n_{k,E+1} > n_{k,E} \end{cases}$. This implies that $n_{k,E}$ always verifies $p(E+1) \leq n_{k,E} \leq k(E+1)$ where p is the prime previous to k for $E < p-1$. Corollary 1 gives a lower bound on $n_{k,E}$.

Corollary 1 *If $E < \frac{k+1}{2}$, then $\frac{k+1}{2}(E+1)+1 < n_{k,E}$ for any $k > 0$.*

Proof. By Bertrand's postulate [5, 24], if $k \geq 6$, there exist a prime p such that

$$\left\lceil \frac{k+1}{2} \right\rceil < p < 2\left\lceil \frac{k+1}{2} \right\rceil - 2 \leq k.$$

Therefore $\frac{k+1}{2}(E+1) < p(E+1) = n_{p,E} < n_{k,E}$ if $E < \frac{k+1}{2}$. One verifies the cases $k \leq 5$ on table 1. □

However, for larger values of E, Figure 3 suggests that $n_{E,k}$ increases at a much lower rate. We will now focus on the asymptotic behavior of $n_{k,E}$. In order to find a lower bound on the value of n that solves Problem 1, we will construct by induction a subset S producing a large value for n. It is a generalization of the worst case error vector of Lemma 2.

E	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$k=3$	3	6	7	8	10	12	15	16	17	18	19	21	22	23	25	27
$k=4$	4	7	11	12	14	16	18	20	22	24	26	29	31	32	35	36
$k=5$	5	10	15	20	21	22	23	26	30	32	35	40	45	46	47	48
$k=6$	6	11	16	21	27	28	30	31	34	38	42	43	47	52	53	
$k=7$	7	14	21	28	35	42	43	44	45	47	49	54	58			
$k=8$	8	15	22	29	36	43	51	52	53	55	57	60	64			
$k=9$	9	18	25	32	39	46	53	58	59	62	66	72	74			
$k=10$	10	19	29	34	41	48	55	62	65	69	74	79				
$k=11$	11	22	33	44	55	66	77	88	99	110	111	112				
$k=12$	12	23	34	45	56	67	78	89	100	111	123	124				
$k=13$	13	26	39	52	65	78	91	104	117	130	143	156				

Table 1: Solution $n_{k,E}$ to Problem 1 for the first values of k and E

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

$$\begin{cases} v_1 = (\underbrace{0,\ldots,0}_{k-1 \text{ times}}) \in \mathsf{K}^{m_1} \\ v_{i+1} = (\underbrace{v_i, \overbrace{1,\ldots,1}^{k^{i-1} \text{ times}}, \ldots, v_i, \overbrace{1,\ldots,1}^{k^{i-1} \text{ times}}}_{k-1 \text{ times}}) \in \mathsf{K}^{m_{i+1}} \end{cases}$$

Lastly, define w_i as the vector v_i without its trailing $k^{i-2}+k^{i-3}+\cdots+1$ ones. w_i has length

$$n_i = k^i - k^{i-1} - \cdots - 1 = k^i - \frac{k^i-1}{k-1} = \frac{(k-2)k^i+1}{k-1}.$$

The Hamming weight of v_i satifies $w_H(v_1)=0$ and $w_H(v_{i+1}) = (k-1)(w_H(v_i)+k^{i-1})$ which solves into $w_H(v_i) = k^i - k^{i-1} - (k-1)^i = m_i - (k-1)^i$. Finally, w_i has weight $E_i = n_i - (k-1)^i$.

Lemma 3 *Let S be the support of v_i. If k is prime, there is no $r \in \mathbb{Z}_{\geq 0}, s \in \mathbb{Z}_{>0}$ with $r+s(k-1) < n_i$ such that $\{r, r+s, r+2s, \ldots r+(k-1)s\} \cap S = \emptyset$.*

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

Let $\overline{r}, \mu, \nu$ be such that $r = \overline{r} + \mu k^\ell + \nu k^{\ell+1}$ with $0 \leq \overline{r} < k^\ell$ 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 $P_{r,s}$ contains the element $x = r + js$ and we write

$$\begin{aligned} x &= r + js = \overline{r} + (j\alpha + \mu)k^\ell + \nu k^{\ell+1} \\ &= \overline{r} + (k-1)k^\ell + (\nu+\lambda)k^{\ell+1}. \end{aligned}$$

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 $(v_\ell, \overbrace{1,\ldots,1}^{k^\ell \text{ times}})$. Then the term $(k-1)k^\ell = m_\ell$ is precisely the dimension of v_ℓ. Lastly, as $\overline{r} < k^\ell$, we deduce that the element of index x is a 1 in v_i. □

Remark 2 As suggested by a referee, we remark that problem 1 is closely related to the famous problem of finding the largest sub-sequence of $\{1,\ldots,n\}$ not containing k terms in arithmetic progression. Let $r(k,n)$ denoted the size of such a largest sub-sequence. If $n \geq r(k,n)+E+1$, a subset of E errors can not suffice to intersect all arithmetic progressions of k terms. Hence $n_{k,E} = \min\{n : n - r(k,n) \geq E+1\}$. Noting that $r(k,n) \leq r(k,n+1) \leq r(k,n)+1$, we deduce that for given k and E, there always exists a n^* such that $n^* - r(k,n^*) = E+1$ and consequently $n_{k,E} = n^* = r(k,n^*)+E+1$. The value $r(k,n)$ has been first studied by Erdős and Turán [8] who conjectured that for all $k \geq 3$, $\lim_{n\to\infty} r(k,n)/n = 0$ which was proven by Szemeredi [28]. In particular the construction of a bad error vector w_i for k prime has connections with a construction of [8, 29]: its support is formed by any element of $\{1,\ldots,n\}$ whose base k expansion contains at least one digit equal to $k-1$. This yields to the estimate

$$r\left(k, \frac{(k-2)k^i+1}{k-1}\right) \geq (k-1)^i. \tag{2}$$

Szekeres conjectured that equality held in (2) (see [8]) which was disproved by Salem and Spencer [27].

The error correction rate of the affine sub-sequence list decoding is therefore directly related to the growth of the ratio $r(k,n)/n$ which is a core problem in additive combinatorics.

$$\frac{E}{n} = 1 - \frac{r(k,n)}{n} - \frac{1}{n}. \tag{3}$$

Szemeredi's theorem states that arithmetic progressions are dense, i.e. an asymptotically large number of errors is necessary to intersect all of them and rule out any list decoding possibility. Now there is unfortunately no known expression of $r(k,n)/n$ as a function of the information rate k/n, to the best of our knowledge and we will now try to estimate bounds on this decoding capacity.

The error vectors w_i approach a worst error distribution (but the result of Salem and Spencer proves that it is not the worst case one). Consequently we can derive from equation (2) an upper bound on the maximal correction radius E: for $n = \frac{(k-2)k^i+1}{k-1}$ we have

$$E = n - r(k,n) - 1 \leq k^i - (k-1)^i - \frac{k^i - 1}{k-1} - 1$$
$$\leq (i-1)k^{i-1} - \frac{k^{i-1}-1}{k-1} - 1,$$

as the function $f(x) = x^i$ is convex. Hence

$$E \leq (i-1)k^{i-1} - \frac{k^{i-1}}{k-1} - \frac{k-2}{k-1}$$

As $k^{i-1} = \frac{(k-1)n-1}{k(k-2)}$ we have $i-1 \leq \log_k \frac{n}{k-2}$ and $\frac{k^{i-1}}{k-1} = \frac{n}{k(k-2)} - \frac{1}{k(k-2)(k-1)}$, therefore

$$E \leq \frac{n}{k-2}\left(\log_k \frac{n}{k-2} - \frac{1}{k}\right) - \frac{k-2}{k} \quad (4)$$
$$\leq \frac{n}{k-2}\log_k \frac{n}{k-2}$$

This shows that, in the worst case, the improvement of the affine sub-sequence technique to the correction radius, compared to the previous list decoding ($\frac{n}{k} - 1$) is essentially no bigger than a logarithmic factor.

The task of bounding E or equivalently E/n from below is much harder. In [26], Roth proved $r(3,n) \leq \frac{c}{\log\log n}$, leading to $\frac{E}{n} \geq 1 - \frac{c}{\log\log n}$ but for an arbitrary k the best known bound is given by Gowers [12]:

$$\frac{E}{n} \geq 1 - \frac{1}{(\log\log n)^{1/2^{2k+9}}}$$

Figure 5 compares the upper bound on the correction capacity E of equation (4) with the actual values of Table 1 for $k = 5, 7$.

5. CHARACTERISTIC ZERO

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 minimum 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*

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

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

Proof. Consider the 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 ξ_i. By Descartes's rule of sign $f - g = 0$. □

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

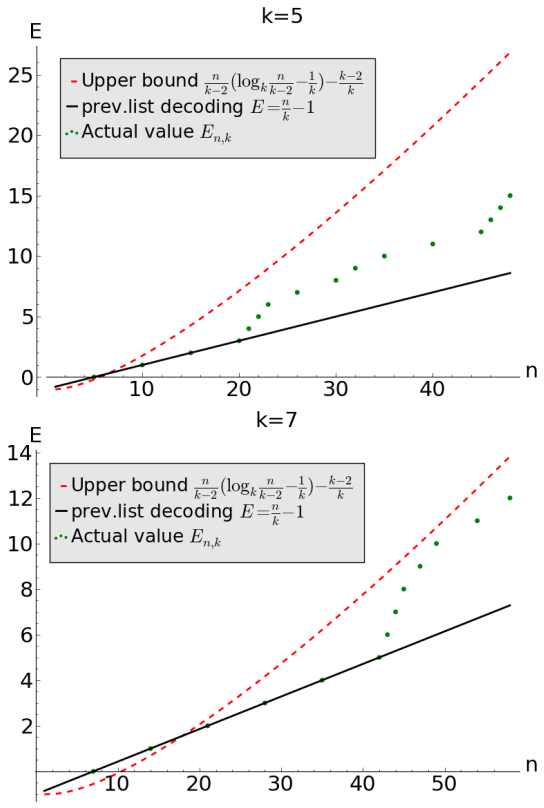

Figure 5: Decoding radius for the affine sub-sequence algorithm: comparison of upper, lower bounds and actual value

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

Remark 3 We do not have an efficient (in polynomial time) decoder up to half this minimum distance. However, notice that when choosing the evaluation points $\xi_i = \alpha^i$ for some $\alpha \in \mathbb{R}_{>0} \backslash \{1\}$, the list decoder presented in Section 3 can be used. Interestingly, it turns out to be a unique decoder as long as an affine sub-sequence free of error exists. Indeed, the list of candidates can be sieved by removing the polynomials which evaluations differ by more than $\delta/2$ positions with the received word. Finally the minimum distance of Theorem 4 ensures that only one code-word lies within less than $\delta/2$ modifications of the received word, hence the decoding is unique.

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

Theorem 5 *Let T, D, and $n \geq k = 2T\frac{\log(D)}{\log(2T)}$ 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*

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

has minimum distance

$$\delta = n - k + 1 = n - 2T\frac{\log(D)}{\log(2T)} + 1.$$

PROOF. Consider the 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 $(f-g)(\zeta_j)$ vanishes for at least k of the ζ_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 $M \leq D^{2T}$ 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 $x^{e_t \bmod p_\ell}$ is isolated in $h(x) = (f(x) - g(x) \bmod (x^{p_\ell}-1))$, and therefore the polynomial $h(x)$ is not zero; h has at most $2T$ terms, and $h(\zeta_\ell) = 0$. This means that $h(x)$ and $\Psi_\ell(x) = 1 + x + \cdots + x^{p_\ell - 1}$ have a common GCD. Because Ψ_ℓ is irreducible over $\mathbb{Q}$, and since $\deg(h) \leq p_\ell - 1$, that GCD is Ψ_ℓ. So h is a scalar multiple of Ψ_ℓ and has $p_\ell > 2T$ non-zero terms, a contradiction. □

Corollary 3 *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 $f(\zeta_i)$ for $k+2E$ p_i-th roots of unity $\zeta_i \neq 1$, where $2T < p_1 < p_2 < \cdots < p_{N+2E}$, p_i prime. Again $e \leq E$ of those values can be erroneous $f(\zeta_i)+\epsilon_i$. 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 $f(\zeta_i)+\epsilon_i$, then $g = f$.*

6. CONCLUSION

Our codes, arising from a natural construction, are surprisingly rich and difficult to analyze. On one hand, it is natural to choose evaluation points as consecutive powers of a primitive root of unity, in order to benefit from the efficient interpolation algorithm of Blahut/Ben-Or/Tiwari, but it is precisely this setting that implies existence of bad worst case error vectors and hence reduces their minimum distance. Much better minimum distances should be attained in the general case, as suggested by Theorem 5, but then no efficient decoding algorithm is available. Those are apparently difficult problems left to be solved.

7. ACKNOWLEDGMENTS

We are thankful to Daniel Augot, Bruno Salvy and the referees for their helpful remarks and suggestions.

Note added on July 18, 2015: Corrected the spelling of G. Szekeres. Added entries $n_{6,14} = 53, n_{8,12} = 64$, $n_{9,12} = 74, n_{10,11} = 79$ to Table 1.

8. 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. A universal reed-solomon decoder. *IBM Journal of Research and Development 28*, 2 (March 1984), 150–158.

[3] BLAHUT, R. E. *Theory and Practice of Error Control Codes*. Addison Wesley, Reading, 1983.

[4] CANDES, E., AND TAO, T. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Trans. Inf. Theory 52*, 12 (2006), 5406–5425.

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

[6] 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* (july 2012), pp. 138–145.

[7] 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.

[8] ERDÖS, P., AND TURÁN, P. On Some Sequences of Integers. *J. London Math. Soc. S1-11*, 4 (1936), 261–264.

[9] GARG, S., AND SCHOST, ÉRIC. Interpolation of polynomials given by straight-line programs. *Theoretical Comput. Sci. 410*, 27-29 (2009), 2659 – 2662.

[10] GIESBRECHT, M., LABAHN, G., AND LEE, W. Symbolic-numeric sparse interpolation of multivariate polynomials. *J. Symbolic Comput. 44* (2009), 943–959.

[11] GIESBRECHT, M., AND ROCHE, D. S. Interpolation of shifted-lacunary polynomials. *Computational Complexity 19*, 3 (Sept. 2010), 333–354.

[12] GOWERS, W. T. A new proof of Szemerédi's theorem. *Geom. Funct. Anal. 11*, 3 (2001), 465–588.

[13] 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.

[14] HUANG, K.-H., AND ABRAHAM, J. A. Algorithm-based fault tolerance for matrix operations. *IEEE Trans. Comput. 33*, 6 (June 1984), 518–528.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] MASSEY, J., AND SCHAUB, T. Linear complexity in coding theory. In *Coding Theory and App.*, G. Cohen and P. Godlewski, Eds., vol. 311 of *LNCS*. Springer Verlag, 1988, pp. 19–32.

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

[21] MEIDL, W., AND NIEDERREITER, H. Linear complexity, k-error linear complexity, and the discrete fourier transform. *J. Complexity 18*, 1 (2002), 87 – 103.

[22] MOON, T. K. *Error correction coding: mathematical methods and algorithms*. Wiley-Interscience, 2005.

[23] 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.

[24] RAMANUJAN, S. A proof of Bertrand's postulate. *J. of the Indian Mathematical Society 11* (1919), 181–182.

[25] REED, I. S., AND SOLOMON, G. S. Polynomial codes over certain finite fields. *J. SIAM 8*, 2 (June 1960), 300–304.

[26] ROTH, K. F. On certain sets of integers. *J. London Math. Soc. 28* (1953), 104–109.

[27] SALEM, R., AND SPENCER, D. C. On sets which do not contain a given number of terms in arithmetical progression. *Nieuw Arch. Wiskunde (2) 23* (1950), 133–143.

[28] SZEMERÉDI, E. On sets of integers containing no k elements in arithmetic progression. In *Proc. Int. Congress of Mathematicians (Vancouver, BC, 1974), Vol. 2* (1975), Canad. Math. Congress, Montreal, QC, pp. 503–505.

[29] Wagstaff, Jr., S. S. On k-free sequences of integers. *Math. Comp. 26* (1972), 767–771.

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