INTERNATIONAL COMPUTER SCIENCE INSTITUTE

1947 Center St. • Suite 600 • Berkeley, California 94704-1198 • (510) 643-9153 • FAX (510) 643-7684

Decoding Algebraic-Geometric Codes Beyond the Error-Correction Bound

M. Amin Shokrollahi and Hal Wasserman

TR-98-019

June 1998

Abstract

We generalize Sudan's results for Reed-Solomon codes [41] to the class of algebraic-geometric codes, designing polynomial-time algorithms which decode beyond the error-correction bound $(d-1)/2$, where d is the minimum distance of the code.

We introduce $[n,k,e,b]_q$-codes, which are linear $[n,k]_q$-codes such that any Hamming sphere of radius e contains at most b codewords. Using the sequence of Garcia-Stichtenoth function fields [10], we construct sequences of constant-rate $[n,k,e,b]_q$-codes for which $e/n \to \epsilon > 1/2$ as n grows large, while b and q remain fixed. Equivalently, we specify arbitrarily large constant-rate codes over a fixed field $\mathbb{F}_q$ such that a codeword is efficiently, non-uniquely reconstructible after more than half of its letters have been arbitrarily corrupted. Additionally, we discover a very simple algorithm for conventional decoding of AG-codes.

Furthermore, we construct codes such that a codeword is *uniquely* and efficiently reconstructible after more than half of its letters have been corrupted by noise which is random in a specified sense. We summarize our results in terms of bounds on asymptotic parameters, giving a new characterization of decoding beyond the traditional error-correction bound.

1 Introduction

A linear error-correcting $[n,k]_q$-code C is a k-dimensional subspace of $\mathbb{F}_q^n$. The elements of C are called *codewords.* If the minimum Hamming distance between any two distinct codewords is d, then C is called an $[n,k,d]_q$-code. n, k, and d are called the *block length*, *dimension*, and *minimum distance* of C, respectively.

Following a construction of Goppa [13], one can use algebraic curves over finite fields to design linear error-correcting codes called *algebraic-geometric codes* or *AG-codes*. These codes can be viewed as a generalization of Reed-Solomon codes, for an AG-codeword is formed by evaluating at specified rational points a function in the function field of a curve. Given a curve of genus g, n rational points, and a specified *designed distance* $d^* \leq n-g$, one can construct an AG-code of block length n which is assured to have minimum distance at least d^*.

The encoding algorithm for an AG-code is simple once one knows the n rational points and a basis for the functions to be evaluated. However, corresponding decoding algorithms are more complicated. Several such algorithms have recently appeared [5, 7, 8, 9, 17, 18, 26, 27, 29, 30, 37, 44]. These algorithms generally decode up to $\lfloor (d^* - 1 - \ell)/2 \rfloor$ errors, where ℓ is some integer between zero and the genus of the curve. They perform between $O(n^2)$ and $O(n^3)$ operations over the base field $\mathbb{F}_q$.

Most of these algorithms fail if the number of errors exceeds the designed error-correction bound $(d^* - 1)/2$, and all of them fail if the number of errors exceeds the true error-correction bound $(d-1)/2$, where d is the minimum distance of the code.

Beyond the true error-correction bound it is in general not possible to uniquely reconstruct a codeword from a received word. However, one might hope to reconstruct a small set of candidates which provably includes the original codeword. This motivates the following definition.

Definition 1. *An $[n,k,e,b]_q$-code is an $[n,k]_q$-code such that the Hamming sphere of radius e around any $y \in \mathbb{F}_q^n$ contains at most b codewords.*

For example, note that an $[n,k,2e+1]_q$-code is an $[n,k,e,1]_q$-code.

The decoding task associated with an $[n,k,e,b]_q$-code requires methods quite different from those customarily employed for error-correction. For Reed-Solomon codes, several researchers investigated decoding algorithms that could correct more than $(d-1)/2$ errors. Welch and Berlekamp [46, 2] designed an algorithm that could correct $(d-1)/2+1$ errors in time $O(n^2)$. Dumer [6] developed an algorithm that could correct $d/2 + O(\log n)$ errors in time $O(n^2)$. Sidel'nikov [35] gave another algorithm based on computing zeros of multivariate polynomials.

A different line of thought was pursued by Sudan [40, 41], who, extending results of Ar et al. [1], investigated alternative decoding algorithms for Reed-Solomon codes. By generalizing a decoding algorithm of Welch and Berlekamp [46, 2], he derived the surprising result that an $[n,k]_q$ Reed-Solomon code is an $[n,k,e,b]_q$-code such that e is approximately $n - \sqrt{2kn}$ and b is approximately $\sqrt{2n/k}$. Hence e can be close to n while k/n is bounded away from zero and b is small. Moreover, Sudan designed a corresponding polynomial-time decoding algorithm based on factorization of bivariate polynomials. Sudan's algorithm thus has an unprecedented ability to efficiently reconstruct meaningful data in spite of very high amounts of noise. Reed-Solomon codes are however of limited utility, as they require the block length n to be no larger than the field size q.

Here we generalize Sudan's results to the class of AG-codes. Our main result is as follows:

Theorem 2. *Let C be an $[n,k,d]_q$-AG-code built over an algebraic function field K of genus g. Let $\alpha := k+g-1$ and $\beta := \lceil\sqrt{2\alpha n}+g-1\rceil$. Then C is an $[n,k,n-\beta-1,\lceil\sqrt{2n/\alpha}\,\rceil]_q$-code.*

We prove this theorem in Section 3. In Sections 3 and 4 we specify a corresponding decoding algorithm, which runs in time polynomial in n and in the size of the basis functions used for encoding, where a function's *size* is a bound on the complexity of evaluating it at a rational point. As for Sudan's method, the crucial tool for our general algorithm is polynomial factorization. However, while Sudan needs to factor a bivariate polynomial over a finite field, we must investigate the more general problem of factoring a univariate polynomial over an algebraic function field. Our algorithm for this task is an adaptation of a well-known procedure for factoring polynomials over number fields, as described in, e.g., [4, 19, 21].

Applying Theorem 2 to the asymptotically good sequence of Garcia-Stichtenoth function fields [10] allows us to construct sequences of $[n,k,e,b]_{q^2}$-codes having excellent parameters. This application is carried out in Section 5.

The bottleneck of our algorithm is the factorization routine. In Section 6 we derive for small b variants of our general algorithm in which the factorization step has been eliminated. One variant, which corresponds to the case of decoding $[n,k,e,1]_q$-codes (i.e., conventional decoding), is particularly interesting. It is slightly less powerful than existing AG-decoding algorithms, as it can correct only up to $(d^*-1)/2-g$ errors. However, its implementation only requires solving an $n\times(n+1)$ system of linear equations. Because of its simplicity, we feel that this algorithm is of independent interest. Another variant corresponds to the case of decoding $[n,k,e,2]_q$-codes. We find that the small increase from $b=1$ to $b=2$ leads to a substantial result, allowing us to reconstruct after a number of errors which may approach $2n/3$. Moreover, the associated decoding algorithm is again found to be of interest because of its enhanced efficiency.

A natural property of the high-noise decoding algorithms discussed above is that they may not return a *unique* solution. We argue that this does not limit the utility of such algorithms, as they can potentially be combined with other strategies, such as soft decision decoding, to identify the one true solution. Moreover, we will show in Section 7 that, if a large fraction of the letters of a codeword are corrupted by noise which is random in a specified sense, it may be possible to *uniquely* decode. This inspires us to speak of an $[n,k,e,1]_q^p$-code, which is defined to be an $[n,k]_q$-code such that, with probability of failure $\leq p$, a codeword may be uniquely reconstructed after up to e of its letters have been replaced with uniform-random letters. We will demonstrate that there exist constant-rate $[n,k,e,1]_q^p$-AG-codes for which e is surprisingly close to n while p is very small. We also specify a corresponding decoding algorithm, which does not require polynomial factorization and hence is particularly efficient.

In Section 8 we consider our argument that, after we have employed non-unique decoding, heuristics may be used to identify the one true solution. We demonstrate that, if an AG-code is employed in conjunction with cryptographic signatures, then we can specify codes which are in effect uniquely decodable after a large fraction of a codeword's letters have been adversarially corrupted.

In Section 9 we investigate asymptotic properties of $[n,k,e,b]_q$-codes. We define analogues of the function α_q [23, Def. 5.1.2] for our new classes of $[n,k,e,b]_q$-codes and $[n,k,e,1]_q^p$-codes, and derive lower bounds for these functions by creating asymptotically good sequences of AG-codes.

One of the crucial time-parameters of our decoding algorithms is the size of the basis functions used for encoding. In Section 10 we show that for codes based on several classes of function fields, this parameter is in fact polynomial in the block length n. Section 10 also includes a novel basis computation for the Garcia-Stichtenoth function field F_3.

Finally, Section 11 gives conclusions and poses open problems.

2 AG-Codes

Throughout this paper we will use the terminology of algebraic function fields, for which we refer the reader to [39]. Let $K/\mathbb{F}_q$ be an algebraic function field of one variable with field of constants $\mathbb{F}_q$, genus g, and distinct prime divisors of degree one $Q, P_1, \ldots, P_n$. Let α be a non-negative integer less than n. $L(\alpha Q)$ denotes the linear space of αQ, i.e., the set of all $f \in K$ which have no poles except a pole at Q of order at most α. By the Riemann-Roch Theorem, $L(\alpha Q)$ is a vector space over $\mathbb{F}_q$ of dimension $\geq \alpha - g + 1$.

The AG-code $C = C(\alpha Q; P_1, \ldots, P_n)$ is defined to be the image of the evaluation map $L(\alpha Q) \to \mathbb{F}_q^n$ mapping a function f to the vector $(f(P_1), \ldots, f(P_n))$. An application of the Riemann-Roch Theorem proves that C is an $[n, \alpha - g + 1, d]_q$-code, where $d \geq n - \alpha$. The quantity $d^* := n - \alpha$ is called the designed distance of C. For further information on AG-codes, we refer the reader to [13, 14, 32, 39, 42].

The encoding algorithm for an AG-code requires us to know a specific basis of $L(\alpha Q)$. Such bases can be constructed in polynomial time in some cases, e.g., when the code is built over a plane algebraic curve with only ordinary singularities [15, 16].

The following lemma [29, Prop. 3] shows that it suffices to know a special basis of $L((2g+1)Q)$.

Lemma 3. *There exist $\varphi_1, \ldots, \varphi_{g+2}$ in $L((2g+1)Q)$ which have distinct pole orders at Q and thus form a basis of $L((2g+1)Q)$. Moreover, for any $j \geq 2g+1$ there is a basis of $L(jQ)$ which consists of suitable power products of the φ_i.*

PROOF. A *gap* at Q is a positive integer s such that $L(sQ) = L((s-1)Q)$. By the Riemann-Roch Theorem, there are exactly g gaps at Q, and for any gap s we have $0 < s < 2g$. As a result, functions $\varphi_1, \ldots, \varphi_{g+2}$ as specified exist and form a basis of $L((2g+1)Q)$.

For the proof of the second part we use induction on j. The induction start $j = 2g+1$ has already been proved, hence we may concentrate on the induction step. Let $j > 2g+1$. The number of positive integers $k < j$ such that either k or $j - k$ is a gap is at most $2g$. Hence, as the number of positive integers less than j is $j - 1 > 2g$, there must exist k such that k and $j - k$ are both non-gaps. The induction hypothesis now applies to prove the lemma. □

Making use of this lemma, we will assume hereafter that we have determined functions $\varphi_1, \ldots,$ φ_{n-g+1} such that, for each $j \leq n$, $L(jQ)$ has basis $\varphi_1, \ldots, \varphi_\ell$ for some ℓ.

For the rest of this paper we fix the function field $K/\mathbb{F}_q$, the divisor αQ, and the evaluation points $P_1, \ldots, P_n$, and assume that we know the functions $\varphi_1, \ldots, \varphi_{n-g+1}$.

3 Decoding AG-Codes Beyond the Error-Correction Bound

We will now prove Theorem 2 by specifying a decoding algorithm. We continue to employ the notation of the previous section. Let us first recall the statement of the theorem.

Theorem 2. *Let C be an $[n, k, d]_q$-AG-code built over an algebraic function field K of genus g. Let $\alpha := k + g - 1$ and $\beta := \lceil \sqrt{2\alpha n} + g - 1 \rceil$. Then C is an $[n, k, n - \beta - 1, \lceil \sqrt{2n/\alpha} \rceil]_q$-code.*

PROOF. Let $h \in L(\alpha Q)$ and let $x = (h(P_1), \ldots, h(P_n))$ be the image of h under the evaluation map. Suppose that $y = (y_1, \ldots, y_n) \in \mathbb{F}_q^n$ is such that x and y agree in at least $\beta + 1$ coordinate places. We will prove that the following algorithm (a generalization of Sudan's [41]) computes a list of at most $\lceil\sqrt{2n/\alpha}\,\rceil$ codewords one of which must be x:

1. (INTERPOLATION STEP) Let $s := \lfloor(\beta - g + 1)/\alpha\rfloor$. Find a nonzero polynomial $G(T) := u_s T^s + \cdots + u_1 T + u_0 \in K[T]$, where $u_j \in L((\beta - j\alpha)Q)$, such that $G(P_i, y_i) := \sum_{j=0}^{s} u_j(P_i) y_i^j$ is zero for $i = 1, \ldots, n$.

2. (FACTORIZATION STEP) Find all roots ρ of $G(T)$ in K. For each such ρ compute $x_\rho := (\rho(P_1), \ldots, \rho(P_n))$. If x_ρ is not defined or if the distance between x_ρ and y is larger than $n - \beta - 1$, discard x_ρ.

Note that x must be among the solutions x_ρ. Indeed, $G(h) \in L(\beta Q)$, and $G(h)$ has zeros at all P_i such that $y_i = h(P_i)$. By assumption there are at least $\beta + 1$ such P_i. Hence $G(h)$ has more zeros than poles, so must be zero.

Also observe that the number of codewords returned by the algorithm is at most the degree of G in T, which is at most $s \leq \lceil\sqrt{2n/\alpha}\,\rceil$. Thus it remains only to show that a nonzero polynomial G with the properties specified in step (1) must exist.

To prove this, we proceed as follows. For each j, let $u_j = u_{j,1}\varphi_1 + \cdots + u_{j,\ell}\varphi_\ell$ with unknown coefficients $u_{j,i} \in \mathbb{F}_q$. The conditions $G(P_i, y_i) = 0$ give a system of n homogeneous linear equations for the unknown coefficients. Hence a nonzero G with the properties specified in step (1) must exist if the number of unknowns is greater than n. But the number of unknowns is at least $\sum_{j=0}^{s}(\beta - g + 1 - j\alpha) = (s+1)(\beta - g + 1) - \alpha s(s+1)/2 > (\beta - g + 1)^2/(2\alpha)$, and so is greater than n by the choice of β. □

This completes our proof of Theorem 2. In the next section we will show that the specified decoding algorithm is in fact polynomial time. Furthermore, in Sections 6 and 7 we will present simplified algorithms which do not require polynomial factoring.

4 Polynomial-Time Algorithms

To complete our main algorithm, we must find the roots in K of a polynomial $G \in K[T]$, where K is a finite extension of rational function field $L := \mathbb{F}_q(X)$. We will do so by designing an efficient algorithm for factoring G completely. Our algorithm will be subject to two assumptions. First, we assume that K is a separable extension of L. Second, we assume that we know a primitive element ω of the extension $K \supset L$, as well as the minimal polynomial $f \in L[Y]$ of ω over L. These assumptions seem reasonable and, as we will demonstrate in Section 5, can be generalized to work conveniently with a tower of separable extensions. This example should serve as a prototype for how to proceed in the case of a general separable extension.

We now continue with our solution, which adapts a well-known procedure for factoring univariate polynomials over number fields [4, 19, 21].

To each $\kappa \in K$ there corresponds a unique $g_\kappa \in L[Y]$ of degree less than $\deg(f)$ such that $\kappa = g_\kappa(\omega)$. The *size* of $\lambda \in L$ is the sum of the degrees of its numerator and denominator when it is written as a rational function in lowest terms. The size of $g := \sum \gamma_i Y^i \in L[Y]$ is the sum of the sizes of the γ_i. The size of $\kappa \in K$ is the size of g_κ. The size of $G := \sum \kappa_i T^i \in K[T]$ is the sum of the sizes of the κ_i.

Remark 4. *Arithmetic operations on elements of K of size at most S can be performed in time polynomial in degree $[K\colon L]$, $\mathrm{size}(f)$, and S. It follows that sums, products, and gcd's of polynomials over K of degree at most d and size at most S can be calculated in time polynomial in $[K\colon L]$, $\mathrm{size}(f)$, S, and d.*

Let $\kappa \in K$. The *norm* of κ over L, denoted by $\mathrm{N}(\kappa)$, is defined to be $\prod_\sigma \sigma(g_\kappa(\omega))$, where σ runs over a complete set of representatives of the Galois group of the Galois closure of K modulo the fixed group of K. Alternatively, $\mathrm{N}(\kappa) = a^{\deg(f)} \cdot \mathrm{Res}(f, g_\kappa)$, where a is the leading coefficient of g_κ and $\mathrm{Res}(f, g_\kappa)$ is the *resultant* of f and g_κ [4, p. 332]. Note that $\mathrm{N}(\kappa)$ is an element of L.

Similarly, for $G := \sum \kappa_i T^i \in K[T]$, $\mathrm{N}(G) \in L[T]$ is defined to be $\prod_\sigma \sigma(G)$, where σ runs over the same set as above. If we associate the coefficients $\kappa_i \in K$ of G with the corresponding polynomials $g_{\kappa_i} \in L[Y]$, then we can regard G equivalently as a univariate polynomial $G(T) \in K[T]$ and as a bivariate polynomial $G(Y,T) \in L[Y,T]$. If $G(T)$ is monic, then the norm of G equals the resultant $\mathrm{Res}_Y(f(Y), G(Y,T))$. In the general case, write G as $\ell(G)\tilde{G}$, where $\ell(G)$ is the leading coefficient of $G(T)$ and $\tilde{G}(T)$ is monic. Since the norm is multiplicative, $\mathrm{N}(G) = \mathrm{N}(\ell(G)) \cdot \mathrm{N}(\tilde{G})$. Note that $\deg(\mathrm{N}(G)) = [K\colon L] \cdot \deg(G)$. Representing resultants as determinants, we obtain the following result.

Lemma 5. *Given $G \in K[T]$ we can compute $\mathrm{N}(G)$ in time polynomial in $[K\colon L]$, $\mathrm{size}(f)$, $\mathrm{size}(G)$, and $\deg(G)$.*

The following lemma shows that factorization in $K[T]$ can be reduced to factorization in $L[T]$. Its proof can be found in [4, Lem. 3.6.3].

Lemma 6. *Suppose that $G \in K[T]$ and its norm $N \in L[T]$ are squarefree. Let $N = N_1 \cdots N_s$ be the factorization of N into irreducible elements of $L[T]$. Let G_i be the polynomial gcd of N_i and G over K. Then $G = G_1 \cdots G_s$ is the factorization of G into irreducible elements of $K[T]$.*

It is easy to make G squarefree by dividing out the gcd of G and its derivative. The following lemma assures that we can transform G so that its norm is also squarefree.

Lemma 7. *The number of $\lambda \in L$ such that the norm of $G(T - \lambda\omega)$ is not squarefree is polynomial in $[K\colon L]$ and $\deg(G)$.*

Proof. For each σ appearing in the definition of $\mathrm{N}(G)$ (there are $[K\colon L]$ such σ's), let $(\beta_{\sigma,i})$ be the roots of $\sigma(G)$. By [4, Lem. 3.6.2], $\mathrm{N}(G(T - \lambda\omega))$ is not squarefree iff there exist σ, τ, i, j such that $\lambda = (\beta_{\sigma,i} - \beta_{\tau,j})/(\tau(\omega) - \sigma(\omega))$. Hence the number of such λ is polynomial in $[K\colon L]$ and $\deg(G)$. $\square$

Theorem 8. *Let $K/\mathbb{F}_q$ be an algebraic function field of genus g, let $L := \mathbb{F}_q(X)$ be the rational subfield of K, and assume that K is generated over L by the element ω with minimal polynomial f. Let $Q, P_1, \ldots, P_n$ be distinct prime divisors of degree one of $K/\mathbb{F}_q$, and assume that we have determined basis functions $\varphi_1, \ldots, \varphi_{n-g+1}$ as specified in Section 2. Let $C := C(\alpha Q; P_1, \ldots, P_n)$ for some $\alpha < n$, and recall that C is an $[n,k,e,b]_q$-code for values of e and b as given in Theorem 2. Then, for any given $y \in \mathbb{F}_q^n$, the b or fewer codewords in the Hamming sphere of radius e around y can be found by a deterministic algorithm which performs a number of $\mathbb{F}_q$-operations polynomial in n, q, $[K\colon L]$, $\mathrm{size}(f)$, and $S := \sum_i \mathrm{size}(\varphi_i)$.*

PROOF. Evaluating φ_i at a point P_j takes time polynomial in S. Hence the interpolation of $G \in K[T]$ in the algorithm of Section 3 can be completed in time polynomial in n and S. We can assure that G is reduced to squarefree form by dividing out $\gcd(G, G')$, where G' is the formal derivative of $G(T)$. Next, by Lemma 7, we can find $\lambda \in L$ such that the norm N of $\hat{G} := G(T - \lambda\omega)$ is squarefree. By Remark 4 and Lemma 5, these steps can be carried out efficiently.

Next we compute $g \in L$ such that $N = g\tilde{N}$, where $\tilde{N} \in L[T]$ has coefficients which are *polynomials* in X having gcd 1. We may regard $\tilde{N}$ equivalently as a univariate polynomial in $L[T]$ and as a bivariate polynomial in $\mathbb{F}_q[X, T]$. We then factor $\tilde{N}(X, T) \in \mathbb{F}_q[X, T]$ using a bivariate factorization algorithm. This takes deterministic time polynomial in q and the degree of $\tilde{N}$ [11, 22].

By the Gauß Lemma [20, Chap. IV, Thm. 2.1], the irreducible factors of N in $L[T]$ exactly correspond to the irreducible factors of $\tilde{N}$ in $\mathbb{F}_q[X, T]$. Thus we have factored $N \in L[T]$, and so, by Lemma 6, we can determine the factors of $\hat{G} \in K[T]$. By Remark 4, this step is also polynomial time. Finally, knowing the factors of $\hat{G}$ we also know the factors of G. □

Remark 9. (1) *In all parts of the above algorithm except for the polynomial factorizations over $\mathbb{F}_q$, q contributes polylogarithmically to the running time. To obtain an overall polylogarithmic contribution of q one could employ a probabilistic rather than a deterministic factorization algorithm: refer to [11].*

(2) *If the curve defined by the algebraic function field K is a plane curve with only ordinary singularities, then it follows from the work in [16] that the parameter S in the above theorem is polynomial in n, $[K\colon L]$, and* $\mathrm{size}(f)$. *Furthermore, in many practical cases (e.g., for elliptic or Hermitian function fields), the parameters $[K\colon L]$,* $\mathrm{size}(f)$*, and S are all polynomial in n (as will be demonstrated in Section 10); our algorithm then takes time* $\mathrm{poly}(n)$. *Also note that, if these parameters were on the contrary very large, this would make difficult* any *encoding or decoding of the AG-code. Hence this would not be a problem specific to our algorithm.*

(3) *The algorithm we have presented is polynomial time, but is quite arduous. Its running time can likely be improved using methods to find roots of polynomials over function fields directly, rather than carrying out full polynomial factorization. Furthermore, in Sections 6–7 we will develop more efficient algorithms for certain cases.*

5 Application to Asymptotically Good AG-Codes

Garcia and Stichtenoth [10] give an explicit sequence of function fields $F_m/\mathbb{F}_{q^2}$ having more than $q^{m-1}(q^2-1)$ prime divisors of degree one and genus $g_m < q^{m-1}(q+1)$. The description of these fields is as follows: F_1 is the rational function field $\mathbb{F}_{q^2}(X_1)$, and for $m \geq 2$ we have $F_m := F_{m-1}(Z_m)$, where $Z_m^q + Z_m = X_{m-1}^{q+1}$, and $X_{m-1} := Z_{m-1}/X_{m-2}$ for $m \geq 3$.

Obviously, F_m is a separable extension of $L := F_1$. However, we cannot apply the results of Section 4 to this extension directly, as we do not have a primitive element at our disposal. One way to remedy this problem would be to find a primitive element together with its minimal polynomial. However, this approach is not very natural to the Garcia-Stichtenoth fields and does not yield a deterministic algorithm. Alternatively, we will sketch how the ideas of the Section 4 can be generalized to apply to this tower of separable extensions.

Let $n := q^{m-1}(q^2-1)$. Note that $[F_m\colon L] = q^{m-1} < n$. Each $g \in F_m$ can be represented uniquely as

$$g = \sum_{\mathbf{i}} g_{\mathbf{i}} Z_2^{i_1} \cdots Z_m^{i_{m-1}},$$

where $\mathbf{i}$ runs over all elements of $\{0, \dots, q-1\}^{m-1}$ and $g_{\mathbf{i}} \in L$. We define the size of g to be the sum of the degrees of the numerators and denominators of the $g_{\mathbf{i}}$. The size of a polynomial $G := \sum_i \kappa_i T^i$ over F_m is then defined to be the sum of the sizes of the κ_i. Observe that sums, products, and gcd's of polynomials over F_m of size at most S and degree at most d can be computed in time polynomial in n, S, and d.

For the remainder of this section, let S denote the sum of the sizes of the functions $\varphi_1, \dots, \varphi_{n-g+1}$ forming a basis of $L(nQ)$, where Q is a prime divisor of degree one of $F_m/\mathbb{F}_{q^2}$.

The polynomial G as specified in the algorithm of Section 3 can evidently be computed in time polynomial in n and S. We now have a task of polynomial factorization over F_m, which, as described in Section 4, can be reduced to a task of polynomial factorization over F_{m-1}. Repeating this reduction $m-1$ times, we arrive at the manageable task of factoring a polynomial over rational function field F_1. These repeated reductions generate a blowup in size which is polynomial in $q^{m-1} < n$. It is then easily seen that the time bound of our algorithm is polynomial in n and S.

Combining the above argument with Theorem 2, we obtain the following result.

Theorem 10. *Let C be an $[n,k,d]_{q^2}$-AG-code built over the Garcia-Stichtenoth field $F_m/\mathbb{F}_{q^2}$. Let $\theta := 1/(q-1)$, $\alpha := \lfloor k+\theta n-1 \rfloor$, and $\beta := \lceil \sqrt{2\alpha n}+\theta n-1 \rceil$. Then C is an $[n, k, n-\beta-1, \lceil \sqrt{2n/\alpha}\,\rceil]_{q^2}$-code. Equivalently, letting $R := k/n$, C is an*

$$\Big[n,\ Rn,\ \lfloor (1-\theta-\sqrt{2(R+\theta)}\,)n \rfloor,\ \lceil \sqrt{2/(R+\theta)}\,\rceil\Big]_{q^2}\text{-code.}$$

Furthermore, the codewords in the Hamming sphere of radius $\lfloor (1-\theta-\sqrt{2(R+\theta)}\,)n \rfloor$ around any given $y \in \mathbb{F}_{q^2}^n$ can be computed in time polynomial in n and S, where S is the basis size as defined above.

Proof. We need only prove that C has the parameters stated above. For this we use Theorem 2. Notice that the genus of F_m is at most $n/(q-1) = \theta n$. $\square$

Remark 11. *The running time of our decoding algorithm depends not only on n, but also on S (which of course also affects the encoding time). It is not known whether for each Garcia-Stichtenoth field one can select a divisor Q and a basis $\varphi_1, \dots, \varphi_{n-g+1}$ for $L(nQ)$ such that the size S of this basis is polynomial in n. Section 10 provides a hint in this direction: we will prove that, for the case of F_3, S is indeed polynomial in n.*

Example 12. *Using $F_m/\mathbb{F}_{101^2}$ and setting $k := \lfloor 0.01n \rfloor$, we can create an $[n, k, \lfloor 0.79n \rfloor, 10]_{101^2}$-AG-code. Equivalently, we have specified an infinite sequence of constant-rate codes over $\mathbb{F}_{101^2}$ such that, for each code, at most ten candidates for a codeword may be efficiently determined after 79% of its letters have been arbitrarily corrupted.*

6 Simplified Algorithms

Here we consider variant algorithms which in some cases are stronger and more efficient than our main algorithm of Section 3. In particular, we will specify an extremely simple and efficient algorithm for conventional decoding of AG-codes.

We begin with the following straightforward variant of Theorem 2:

Theorem 13. *Let C be an $[n,k,d]_q$-AG-code built over an algebraic function field of genus g. Then, for any positive integer b, C is an $[n,k,n-\beta-1,b]_q$-code, where $\beta := \lceil (n+1)/(b+1)+b\alpha/2+g-1\rceil$ and $\alpha := k+g-1$.*

Proof. The proof is as for Theorem 2, except that we use $s := b$ and the new value of β. $\square$

The cases $b=1$ and $b=2$ of the above theorem will be of particular interest. Recall that $[n,k,e,1]_q$-codes are those for which a codeword may be conventionally (i.e., uniquely) reconstructed after the corruption of up to e letters. Therefore, we will refer to the case $b=1$ as the *unique reconstruction* case. Similarly, we will refer to the case $b=2$ as the *binary reconstruction* case.

We start with the unique reconstruction case. Setting $b=1$ in the previous theorem yields the following.

Corollary 14. *Let C be an $[n,k,d]_q$-AG-code built over an algebraic function field of genus g, and let $\alpha := k+g-1$. Then C is an $[n,k,\lfloor (n-\alpha-1)/2-g\rfloor,1]_q$-code.*

We saw in Section 2 that AG-codes have designed distance $d^* = n-\alpha$, and thus have designed error-correction bound $e^* := \lfloor (d^*-1)/2\rfloor = \lfloor (n-\alpha-1)/2\rfloor$. Hence Corollary 14 tells us nothing new about the conventional correctability of AG-codes, and indeed falls short of the designed bound by a difference of exactly g. However, the simplified form of our algorithm in this case proves to be of interest.

With reference to our main algorithm, this simplified algorithm may be sketched as follows. Let $\beta := \lceil (n+\alpha-1)/2+g\rceil$. We find a nonzero polynomial $G(T) := u_1T+u_0$, where $u_j \in L((\beta-j\alpha)Q)$, such that $u_1(P_i)y_i+u_0(P_i)=0$ for $i=1,\ldots,n$. If there is any codeword $x=(h(P_1),\ldots,h(P_n))$ within distance $n-\beta-1$ of y, then h must be a root of $G(T)$. But then h can only be $-u_0/u_1$.

Thus our algorithm in this case simplifies to a conventional decoding algorithm giving a unique solution. Moreover, this algorithm only requires solving an $n\times(n+1)$ system of linear equations. Hence it is quite efficient. Furthermore, it seems conceptually simpler than previous algorithms for decoding AG-codes (e.g., [5, 7, 8, 9, 17, 18, 26, 27, 29, 30, 37, 44]). The utility of this algorithm is only slightly diminished by the restriction that the number of errors must be g less than the designed error-correction bound.

The binary reconstruction case is based on the following special case of Theorem 13 for $b=2$.

Corollary 15. *Let C be an $[n,k,d]_q$-AG-code built over an algebraic function field K of genus g. Then C is an $[n,k,\lfloor 2n/3-k-2g+2/3\rfloor,2]_q$-code.*

Observe that, if the rate and genus are sufficiently small, then the error-bound here may rise substantially above the conventional correction bound $e := \lfloor (d-1)/2\rfloor$. Even if we are interested only in unique decoding, we might wish to employ the binary reconstruction algorithm described below: we shall see that it is quite efficient, and, for certain codes, it will allow us to decode right up to the true correction bound.

We now proceed with the description of the binary reconstruction algorithm. Let α, β be as in the statement of Theorem 13 for $b = 2$. As above, we find functions $u_j \in L((\beta - j\alpha)Q)$, $j = 0, 1, 2$, such that $u_0(P_i) + u_1(P_i)y_i + u_2(P_i)y_i^2 = 0$ for $i = 1, \dots, n$. We now need to find the roots in $L(\alpha Q)$ of $G(T) := u_0 + u_1 T + u_2 T^2$. For this we assume that we know a prime divisor $\mathfrak{p}$ of degree $d > \alpha$ of K. This is equivalent to finding an $\mathbb{F}_{q^d}$-rational point P on the corresponding algebraic curve having at least one coordinate which is a primitive element of the extension $\mathbb{F}_{q^d}/\mathbb{F}_q$. This step depends on the presentation of the algebraic function field (or the curve) and can be performed off-line using probabilistic methods. (For a demonstration of how, for instance, to do this in the case of elliptic curves, the reader may consult [33].)

Now suppose that $\varphi_1, \dots, \varphi_\ell$ is a basis of the space $L(\alpha Q)$. We are looking for $h_1, \dots, h_\ell$ such that $h = \sum h_i \varphi_i$ satisfies

$$u_0 + u_1 h + u_2 h^2 = 0. \tag{1}$$

This implies that $u_0(\mathfrak{p}) + u_1(\mathfrak{p})h(\mathfrak{p}) + u_2(\mathfrak{p})h(\mathfrak{p})^2 = 0$. We thus start by solving the equation $u_0(\mathfrak{p}) + u_1(\mathfrak{p})x + u_2(\mathfrak{p})x^2 = 0$ for $x \in \mathbb{F}_{q^d}$. This task of solving a quadratic equation over a finite field $\mathbb{F}_{q^d}$ can be easily accomplished. E.g., using the method of [28] if q is odd, or that of [34] if q is even. Call the solutions x_1, x_2. For $i = 1, 2$, we then solve the linear system

$$\begin{pmatrix} \varphi_1(\mathfrak{p}) & \cdots & \varphi_\ell(\mathfrak{p}) \\ \varphi_1(\mathfrak{p})^q & \cdots & \varphi_\ell(\mathfrak{p})^q \\ \vdots & \ddots & \vdots \\ \varphi_1(\mathfrak{p})^{q^{d-1}} & \cdots & \varphi_\ell(\mathfrak{p})^{q^{d-1}} \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ h_\ell \end{pmatrix} = \begin{pmatrix} x_i \\ x_i^q \\ \vdots \\ x_i^{q^{d-1}} \end{pmatrix}.$$

Let h be a solution of (1), and fix i such that $h(\mathfrak{p}) = x_i$. Then $h(\mathfrak{p}^\sigma) = x_i^\sigma$ for all d distinct automorphisms σ of $\mathbb{F}_{q^d}/\mathbb{F}_q$. But this implies that h is a solution of the above linear system. We also claim that the linear system has only this one solution. Indeed, suppose that $h, \tilde{h}$ are both solutions. Then $h - \tilde{h}$ is a function in $L(\alpha Q)$ which vanishes at the $d > \alpha$ distinct points $\mathfrak{p}^\sigma$, where σ runs over the automorphisms of $\mathbb{F}_{q^d}/\mathbb{F}_q$. This function thus has more zeros than poles, so must be zero.

Solving the above linear system for each of x_1, x_2 thus suffices to efficiently identify all solutions of (1).

7 Unique Decoding for Random Noise

Our principal decoding algorithms, given high-noise corruption y of codeword x, may return more than one possible value for x. This arguably limits the utility of high-noise decoding. In the next two sections, we respond to this argument.

Here we consider the possibility that, if noise is assumed to be random in a specified sense, then a codeword x may be *uniquely* reconstructed from its highly noisy corruption y. We fix the following definition:

Definition 16. *An $[n, k, e, 1]_q^p$-code is an $[n, k]_q$-code such that, after up to e letters of a codeword x have been replaced with uniform-random letters, x may be uniquely reconstructed, with probability of failure at most p.*

Our model of random noise is thus that corrupted letters can only be replaced with uniform-random members of $\mathbb{F}_q$. (Moreover, extensions to the case of near-uniform-random distributions over $\mathbb{F}_q$ should be straightforward.) We however allow the choice of *which* letters to corrupt to be adversarial.

The existence of a unique solution in the presence of large quantities of random noise has been considered previously, e.g., in [12, Sec. 6]. Here we shall accompany our existence result with an efficient decoding algorithm. Indeed, this algorithm will have the advantage of being more efficient than our main algorithm of Section 4.

The new algorithm (which extends a method of [45]) is suggested by the proof of Theorem 2. In that proof, one constructs a polynomial G such that $G(P_i, y_i) = 0$ for $i = 1, \ldots, n$. In the following, we refer to the pairs (P_i, y_i) as *data-points.* We proved that, if there are enough data-points such that $y_i = h(P_i)$, then $G(h) = 0$. Now assume that there are enough such *correct* data-points that, if we interpolate to find $\tilde{G}$ having the first $n/2$ data-points as roots, this is already sufficient to assure that $\tilde{G}(h) = 0$. But $\tilde{G}(h) = 0$ implies that any correct data-point must be a zero of $\tilde{G}$. Hence, among the second $n/2$ data-points, the correct data-points will be identifiable, because they are zeros of $\tilde{G}$ while the noisy data-points are randomized in such a manner that they are unlikely to be zeros of $\tilde{G}$. Hence we have a "filter" which we can use to identify and discard most of the noisy data-points. If this works well enough to leave us with a sufficient majority of correct data-points, we can then uniquely reconstruct h.

As a preliminary, we will need the following Chernoff bounds.

Lemma 17. (1) *Let $X_1, \ldots, X_n$ be independent random variables over $\{0,1\}$ such that $\Pr[X_i = 1] = p$ for all i. Then for any $\delta > 0$, we have $\Pr[\sum_{i=1}^{n} X_i > n(p+\delta)] \le \exp(-2\delta^2 n)$.*

(2) *Assume that e of n items are labeled "bad." Then for any $\delta > 0$, the probability that a randomly selected size-$n/2$ subset of the n items contains more than $e/2 + \delta n$ bad items is upper-bounded by $\exp(-2\delta^2 n)$.*

Proof. (1) From [25, Prob. 4.7(c), p. 98].

(2) The specified tail-probability is evidently upper-bounded by the probability, if we threw the items into two bins so that each item had independent probability 0.5 of landing in the first bin, that the first bin would receive more than $e/2 + \delta n$ bad items. Using (1) and [25, Prob. 4.6], this probability is at most $\exp(-2\delta^2 n)$. □

Theorem 18. *Let C be an $[n,k,d]_q$-AG-code built over an algebraic function field of genus g. Let $\alpha := k + g - 1$ and $\beta := \lceil \sqrt{\alpha n} + g - 1 \rceil$. Then for any $\delta > 0$, C is an $[n,k,e,1]_q^p$-code, where*

$$e := \lfloor n - \delta n - 2\beta - \beta n/(\alpha q) - \alpha - 2g - 1 \rfloor, \tag{2}$$

$$p := 2\exp(-\delta^2 n/2) + 2\exp(-\delta^2 n). \tag{3}$$

Moreover, the associated decoding task can be accomplished efficiently using Algorithm 19 below.

Proof. Assume that C is constructed using divisors $Q, P_1, \ldots, P_n$. We wish to decode a received message $y = (y_1, \ldots, y_n) \in \mathbb{F}_q^n$. At least $n - e$ of the data-points $(P_1, y_1), \ldots, (P_n, y_n)$ are correct, meaning that $y_i = h(P_i)$, for $h \in L(\alpha Q)$. The rest are noisy, meaning that y_i is a uniform-random element of $\mathbb{F}_q$. We then apply the following algorithm.

Algorithm 19. *Randomly partition the data-points into two lists of length $n/2$, which we will call $\mathfrak{D}_1, \ldots, \mathfrak{D}_{n/2}$ and $\mathfrak{D}'_1, \ldots, \mathfrak{D}'_{n/2}$. Let $s := \lfloor(\beta - g + 1)/\alpha\rfloor$. Find a nonzero $G := u_0 + u_1 T + \cdots + u_s T^s$, where $u_j \in L((\beta - j\alpha)Q)$, having $\mathfrak{D}_1, \ldots, \mathfrak{D}_{n/2}$ as zeros. Discard any of $\mathfrak{D}'_1, \ldots, \mathfrak{D}'_{n/2}$ which are not zeros of G. Similarly, find G' having $\mathfrak{D}'_1, \ldots, \mathfrak{D}'_{n/2}$ as zeros, and discard any of $\mathfrak{D}_1, \ldots, \mathfrak{D}_{n/2}$ which are not zeros of G'. Apply our unique reconstruction algorithm of Section 6, or any conventional decoding algorithm, to uniquely determine h from the remaining data-points.*

Our proof proceeds in two stages.
Claim 1. G as specified in Algorithm 19 exists and the probability that $G(h) \neq 0$ is at most $\exp(-\delta^2 n/2)$.
By Lemma 17(2) the probability that $\mathfrak{D}_1, \ldots, \mathfrak{D}_{n/2}$ include more than $e/2 + \delta n/2$ noisy data-points is at most $\exp(-\delta^2 n/2)$. By (2), $e/2 + \delta n/2$ is at most $n/2 - \beta - 1$. Mimicking the proof of Theorem 2 (using $n/2$ in place of n) now yields the claim.
Claim 2. The probability that the number of noisy data-points among $\mathfrak{D}'_1, \ldots, \mathfrak{D}'_{n/2}$ which are zeros of G is larger than $\beta + \beta n/(2\alpha q) + \delta n/2$ is at most $\exp(-\delta^2 n)$.
Since $G := u_0 + u_1 T + \cdots + u_s T^s$ is nonzero, there exists at least one u_{j^*} which is nonzero. But $u_{j^*} \in L(\beta Q)$, so at most β of the P_i can be zeros of u_{j^*}. Now let $\mathfrak{D}'_k = (P_i, y_i)$ be a noisy data-point such that u_{j^*} is nonzero at P_i. Then the independent probability that (P_i, y_i) is a zero of G is at most $s/q \leq \beta/(\alpha q)$. Hence, by Lemma 17(1), the probability that more than $\beta + \frac{n}{2}(\beta/(\alpha q) + \delta)$ of $\mathfrak{D}'_1, \ldots, \mathfrak{D}'_{n/2}$ will be zeros of G is at most $\exp(-\delta^2 n)$.

Given these claims (and the corresponding claims for G'), it follows that, once we complete our filtering stage, with probability $\geq 1 - 2\exp(-\delta^2 n/2) - 2\exp(-\delta^2 n)$ there remain in the worst case $\tilde{n} = (n - e) + (2\beta + \beta n/(\alpha q) + \delta n)$ data-points, of which at least $n - e$ are correct. By Corollary 14, we may uniquely determine h from this filtered data, provided that

$$2\beta + \beta n/(\alpha q) + \delta n \leq (\tilde{n} - \alpha - 1)/2 - g.$$

It is easily seen that (2) suffices to assure this. □

Observe that Algorithm 19 requires only simple linear algebra: we never actually factor G. Hence this algorithm has the advantage of being particularly efficient. Also note that bounds (2) and (3) could no doubt be readily improved.

Example 20. *We create an AG-code based on the Garcia-Stichtenoth field $F_2/\mathbb{F}_{101^2}$ (refer to Section 5), which has genus at most 10302 and more than 1030200 prime divisors of degree one. Theorem 18 then tells us that, using this field and picking $n := 10^6$, $k := 10^4$, $\delta := 0.01$, we can create an $[n, k, 642790, 1]^p_{101^2}$-code, where p is less than 10^{-21}. Equivalently, we have specified a long linear code of rate 0.01 over $\mathbb{F}_{101^2}$ such that, with probability very close to 1, a codeword x may be uniquely and efficiently reconstructed after 64% of its letters have been replaced with random noise.*

8 Unique Decoding via Cryptographic Signatures

While the decoding algorithm of Section 3 may not return a unique solution, we argue that this is not a limitation, as heuristics may subsequently be used to pick among the candidate solutions. Indeed, we will now see that this argument can be made rigorous if our algorithm is used in conjunction with cryptographic methods.

In a cryptographic signature scheme, Alice, wishing to send a message v, computes $w := S_A(v)$. Here S_A is an efficient algorithm which uses a private key known only to Alice. Alice then transmits w, which has the property that anyone may efficiently retrieve v from w and yet an adversary, given v, w, and other publicly available information, cannot efficiently (and with non-trivial probability) find any w' such that $w' = S_A(v')$ for $v' \neq v$. We say that w is a message with an *unforgeable signature.*

Now imagine that Alice encodes w into an AG-codeword x, which she transmits to Bob. During transmission, we allow a high portion of noise to be applied to x by an adversary, so that Bob receives a corrupted message y. Applying one of our non-unique decoding algorithms, Bob determines candidates $w^{(1)}, \ldots, w^{(b)}$ for w. But of these, with high probability only w will be a signed message from Alice. (Indeed, if this were not so, then we would have an efficient means of breaking the signature scheme.) Hence the adversary cannot efficiently prevent Bob from uniquely identifying the message from Alice.

Thus, by combining signatures and AG-codes, we can specify codes such that a codeword is uniquely reconstructible after a majority of its letters have been adversarially corrupted.

9 Asymptotic Considerations

One of the principal goals of asymptotic coding theory is the description of Σ_q, which is the set of all (δ, R) such that for any $n_0 \in \mathbb{N}$ there exist $[n,k,d]_q$-codes for which $n \geq n_0$ and $(d/n, k/n)$ is arbitrarily close to (δ, R). Analogously, for each positive integer b, we now define Σ_q^b to be the set of all (ϵ, R) such that for any $n_0 \in \mathbb{N}$ there exist $[n,k,e,b]_q$-codes for which $n \geq n_0$ and $(e/n, k/n)$ is arbitrarily close to (ϵ, R). We also set $\Sigma_q^\omega := \bigcup_{b=1}^\infty \Sigma_q^b$. We similarly define $\Sigma_q^{\mathcal{R}}$ to be the set of all (ϵ, R) such that for any $n_0 \in \mathbb{N}$ there exist $[n,k,e,1]_q^p$-codes for which $n \geq n_0$, $(e/n, k/n)$ is arbitrarily close to (ϵ, R), and p is upper-bounded by an exponentially small function of n.

Observe that Σ_q^1 is precisely the set of all (ϵ, R) such that $(2\epsilon, R) \in \Sigma_q$. Furthermore, we have obvious inclusions $\Sigma_q^1 \subseteq \Sigma_q^{\mathcal{R}}$ and $\Sigma_q^b \subseteq \Sigma_q^{b'}$ for $b \leq b'$.

The function α_q: $[0,1] \to [0,1]$, defined in [23, Def. 5.1.2], describes the boundary of Σ_q. To be precise, Σ_q is the set of all (δ, R) such that $R \leq \alpha_q(\delta)$. Similarly, we now define β_q to be the inverse of α_q; then Σ_q is equivalently the set of all (δ, R) such that $\delta \leq \beta_q(R)$. (For the results of the current paper, the inverted β notation will be more convenient.)

Employing a shorthand notation, we can write $\Sigma_q = (\leq \beta_q(R), R)$. Analogously, there must exist functions β_q^b, β_q^ω, and $\beta_q^{\mathcal{R}}$ such that $\Sigma_q^b = (\leq \beta_q^b(R), R)$, $\Sigma_q^\omega = (\leq \beta_q^\omega(R), R)$, and $\Sigma_q^{\mathcal{R}} = (\leq \beta_q^{\mathcal{R}}(R), R)$. Our theorems readily provide lower bounds for these functions.

Theorem 21. *Let* $\theta := 1/(q-1)$.

(1) $\beta_{q^2}^b(R) \geq 1 - 1/(b+1) - \theta - b(R+\theta)/2$.

(2) $\beta_{q^2}^\omega(R) \geq 1 - \theta - \sqrt{2(R+\theta)}$.

(3) $\beta_{q^2}^{\mathcal{R}}(R) \geq 1 - 2\theta - 2B - B/(Aq^2) - A$, *where* $A := R + \theta$ *and* $B := \sqrt{A} + \theta$.

Proof. The proof is by constructing sequences of AG-codes based on the Garcia-Stichtenoth fields and then applying: (1) Theorem 13; (2) Theorem 10; (3) Theorem 18. □

Observe that, when q is large, known upper and lower bounds on $\beta^1_{q^2}$ nearly match. Next observe that, when R is small, our lower bound on $\beta^2_{q^2}$ goes well beyond the upper bound on $\beta^1_{q^2}$. It is indeed easily seen from Theorem 21(1) that $\Sigma^2_{q^2} \not\subseteq \Sigma^1_{q^2}$ for all $q \geq 13$. Similarly, our lower bound on $\beta^{\mathcal{R}}_{q^2}$ is sufficient to demonstrate that $\Sigma^{\mathcal{R}}_{q^2}$ goes beyond $\Sigma^1_{q^2}$ when q is large ($q \geq 37$ suffices) and rate is small.

Also keep in mind that $\Sigma^2_{q^2}$ must include the region under the "$b = 1$" line as well as the region under the "$b = 2$" line. Analogously, $\Sigma^{\omega}_{q^2}$ includes the region under the curved "omega" line, which corresponds to Theorem 21(2); but observe that we can improve this lower bound by also including the known $\Sigma^b_{q^2}$ regions for all positive integers b. Correspondingly, for certain codes, the variant algorithms of Section 6 may give stronger results than our main algorithm of Section 3.

The true values of the new β functions are evidently far from known. Section 11 will summarize open questions relating to these functions.

10 Examples

The running time of our decoding algorithm of Section 4 is polynomial in the block length n, the field size q, the size of the polynomial f separably generating the function field K over a rational subfield L, the index $[K\colon L]$, and the parameter S defined to be the total size of basis $\varphi_1, \ldots, \varphi_{n-g+1}$ for $L(nQ)$, where Q is a prime divisor of degree one and g is the genus of K. In this section we will demonstrate for several classes of function fields that, for appropriately chosen Q and φ_i, all of these parameters are polynomial in n. We will consider elliptic function fields, Hermitian function fields, and the Garcia-Stichtenoth field F_3 of Section 5. In all of these cases the crucial assertion is that S is polynomial in n. To prove this, we will explicitly construct the functions φ_i. The basis computation for elliptic function fields is trivial. The basis computation for Hermitian function fields is well known: e.g., refer to [31, 38]. However, our basis computation for the case of F_3 seems to be novel.

Throughout this section we assume familiarity with the theory of algebraic function fields as presented in [39].

10.1 Elliptic Codes

For simplicity we will assume in this subsection that the characteristic of $\mathbb{F}_q$ is different from 2 or 3. Let K be generated over $L := \mathbb{F}_q(X)$ by the polynomial $f(X, Y) = Y^2 - X^3 - cX - d$, where $c, d \in \mathbb{F}_q$ and $27d^2 - c^3 \neq 0$. It is well known that K has genus 1. (For these and related facts about elliptic curves, the reader may consult standard texts such as [36].) Evidently, the size of f as well as $[K\colon L]$ are constants. Thus we need only construct basis functions $\varphi_1, \ldots, \varphi_n$ and show that S, the sum of their sizes, is polynomial in n.

By an abuse of notation, we let Y denote a root of f in the algebraic closure of L. Let Q be the common pole of X and Y. Then $X \in L(2Q)$ and $Y \in L(3Q)$. Hence we can set $\varphi_1 := 1$, $\varphi_2 := X$, and $\varphi_3 := Y$. Each φ_i can then be of form $X^{a_i}Y^{b_i}$, where $0 \leq a_i \leq i/2$ and $b_i \in \{0, 1\}$. Thus the size of φ_i is at most $i/2$, and S is $O(n^2)$. By the Hasse-Weil inequality, K has at most $q + 2\sqrt{q}$ and at least $q - 2\sqrt{q}$ prime divisors of degree one other than Q. Hence q and n have the same order of magnitude, and our decoding algorithm takes time polynomial in n.

10.2 Hermitian Codes

The Hermitian function field K is equal to the Garcia-Stichtenoth field F_2 and is generated over $L := \mathbb{F}_{q^2}(X)$ by the polynomial $f(X,Y) := Y^q + Y - X^{q+1}$. It has $q^3 + 1$ prime divisors of degree one and its genus is $q(q-1)/2$ [10]. We set $n := q^3$. Then size(f) and $[K\colon L]$ are both $o(n)$. Let Q be the common zero of X and Y. Then $1/X \in L(qQ)$ and $1/Y \in L((q+1)Q)$. Observe then that $L(nQ)$ has a basis of the following form: each pole order $\alpha = aq + b$ (for $0 \le a$, $0 \le b < q$) has a corresponding φ_i iff $b \le a$; and in this case $\varphi_i = X^{b-a}Y^{-b}$. It is easily seen that this basis has size $O(n^2)$, and hence that our decoding algorithm takes time polynomial in n.

10.3 Codes over F_3

Recall from Section 5 that $F_3/\mathbb{F}_{q^2}$ is the extension of the Hermitian function field F_2 generated by the polynomial $Z^q + Z - (Y/X)^{q+1}$. (We have changed our notation slightly.) F_3 has more than $q^4 - q^2$ prime divisors of degree one [10, Prop. 3.1] and its genus is $q^3 - 2q + 1$ [10, Thm. 2.10]. Let $n := q^4 - q^2$. To show that our decoding algorithm takes time poly(n), we need only prove that basis size S is polynomial in n. Again, we will do this by exhibiting an explicit prime divisor Q and explicit basis functions φ_i. For Q we choose the unique common zero of the functions X, Y, Z [10, Lem. 2.3].

We will now employ the notation of [10]. The divisor $D^{(3)}$ can be written as $D^{(3)} = E_1 + E_2$, for effective divisors E_1, E_2 such that $E_1 \cap F_1$ is the divisor of poles of X. The discussion in Section 2 of [10] implies that we have the following divisor decompositions for X, Y, Z in F_3:

$$\begin{aligned}
(X) &= Q + D_0^{(3)} + qE_1 - q^2E_2 \\
(Y) &= (q+1)(Q + D_0^{(3)} - qE_2) \\
(Z) &= (q^2+q)Q - (q+1)(E_1 + E_2).
\end{aligned}$$

For integers a, b, c, this readily implies that

$$\begin{aligned}
(X^{c-b}Y^bZ^{-a}) = {} & (bq + c - a(q^2+q))Q + (bq+c)D_0^{(3)} + \\
& (q(c-b) + a(q+1))E_1 + (a(q+1) - q(b+cq))E_2.
\end{aligned} \tag{4}$$

Let us call $\alpha \in \mathbb{N}$ *representable* by the triple of integers (a,b,c) if

(i) $\alpha = a(q^2+q) - (bq+c)$,

(ii) $0 \le bq + c < q^2 + q$,

(iii) $-(q-2) \le c \le 1$,

(iv) $b \le a\left(1 + \frac{1}{q}\right) - q$ if $c = 1$,

(v) $b \le a\left(1 + \frac{1}{q}\right) + c$ if $c \le 0$.

If no such triple exists for α, then we call α *non-representable.* Note that if α is representable, then there exists exactly one triple (a,b,c) satisfying conditions (i)–(v). Furthermore, (4) and conditions (i)–(v) then imply that $X^{c-b}Y^bZ^{-a} \in L(\alpha Q) \setminus L((\alpha-1)Q)$.

Theorem 22. (1) *The gaps at Q are exactly the non-representable $\alpha \in \mathbb{N}$.*

(2) *For $\beta \in \mathbb{N}$, let B be the set of triples (a, b, c) satisfying conditions (i)–(v) above for any $\alpha \leq \beta$. Then $\{X^{c-b}Y^bZ^{-a} \mid (a, b, c) \in B\}$ is a basis of $L(\beta Q)$.*

PROOF. (2) follows immediately from (1). We will sketch the proof of (1), leaving the details to the reader. For each $a = 0, \ldots, q-1$, the number of integers representable by (a, b, c) is $(a+1)+\sum_{i=0}^{a-1} i$: indeed, no integer is representable by $(a, b, 1)$, $a+1$ integers are representable by $(a, b, 0)$, and $\sum_{i=0}^{a-1} i$ integers are representable by (a, b, c) such that $c < 0$. For each $a = q, \ldots, 2q-2$, the number of integers representable by (a, b, c) is $(a-q+2)+(q-1)(q+1)-\sum_{i=0}^{2q-2-a} i$: indeed, $a-q+2$ integers are representable by $(a, b, 1)$, and, for each $c \leq 0$, the number of integers representable by (a, b, c) is $\min\{q+1,\, a+c+1\}$. For each $a > 2q-2$, the number of integers representable by (a, b, c) is q^2+q. It follows that every non-representable $\alpha \in \mathbb{N}$ satisfies $0 \leq \alpha \leq (2q-2)(q^2+q)$, and that, of all α such that $0 \leq \alpha \leq (2q-2)(q^2+q)$, q^3 are representable and q^3-2q+1 non-representable. But q^3-2q+1 is exactly the genus of F_3, which by the Riemann-Roch Theorem is the number of gaps at Q. □

It is now straightforward to check that $L(nQ)$ has a basis as specified in Theorem 22(2) whose size is polynomial in n.

11 Conclusions and Open Problems

We have generalized Sudan's Reed-Solomon results [41] to algebraic-geometric codes. We have introduced the class of $[n, k, e, b]_q$-codes, and have demonstrated that there exist $[n, k, e, b]_q$-AG-codes for which e is surprisingly close to n while k/n is bounded away from zero and b is small. We have designed efficient decoding algorithms which can non-uniquely reconstruct codewords after a number of errors which may go far beyond the conventional error-correction bound. Such coding methods could appropriately be used for extremely noisy channels: e.g., in deep-space communication.

Our general decoding algorithm only requires finding the roots of a univariate polynomial over an algebraic function field. We have provided a polynomial-time algorithm for this task which completely factors the given polynomial. This gives rise to our first open problem:

(1) Is there a more efficient way to compute the roots of a univariate polynomial over an algebraic function field?

The running time of our decoding algorithm depends not only on block length n but also on S, the sum of the sizes of basis functions $\varphi_1, \ldots, \varphi_{n-g+1}$. To remove this additional parameter, we would need an affirmative answer to the following question:

(2) Does every function field have a divisor Q and a basis $\varphi_1, \ldots, \varphi_{n-g+1}$ for $L(nQ)$ such that the sum of the sizes of the φ_i is polynomial in n?

In particular, for applications to asymptotically good AG-codes the following question is of interest.

(3) Is it possible to explicitly construct polynomial-size bases for each of the Garcia-Stichtenoth function fields?

We have seen that this is indeed the case for the Garcia-Stichtenoth field F_3.

A specialization of our general algorithm has yielded a new and very simple conventional decoding algorithm for AG-codes. Its main step is an interpolation over an algebraic function field. Using standard matrix methods, this step can be performed in $O(n^3)$ $\mathbb{F}_q$-operations. But observe that univariate interpolation takes only $O(n^{1+\epsilon})$ $\mathbb{F}_q$-operations [3, Chap. 3]. This observation inspires us to ask:

(4) Can the interpolation step of our algorithm be performed in $O(n^2)$ or $O(n^{1+\epsilon})$ $\mathbb{F}_q$-operations?

An affirmative answer to this question would yield a very fast algorithm for conventional AG-decoding. However, a weakness of our conventional decoding algorithm is that it can correct only up to $(d^* - 1)/2 - g$ errors. Hence we also ask:

(5) Is there a variant of our conventional decoding algorithm which can correct up to $(d^* - 1)/2$ errors?

Restricting ourselves to a model of random noise, we have also introduced $[n, k, e, 1]_q^p$-codes, and have demonstrated that there exist $[n, k, e, 1]_q^p$-AG-codes for which e is close to n while k/n is bounded away from zero and p is very small. Equivalently, we have shown that, for noise random in a specified sense, the codewords of an AG-code are efficiently, uniquely reconstructible after a number of errors which may go far beyond the conventional error-correction bound.

In Section 9 we constructed asymptotically good sequences of $[n, k, e, b]_q$-codes and $[n, k, e, 1]_q^p$-codes over a fixed field $\mathbb{F}_q$. We introduced the functions β_q^b, β_q^ω, and $\beta_q^{\mathcal{R}}$, which are natural variants of the well-known function α_q. Manin [24] has proved the continuity of α_q. This motivates us to ask:

(6) Are β_q^b, β_q^ω, and $\beta_q^{\mathcal{R}}$ continuous?

We have proved lower bounds analogous to the Tsfasman-Vlădut-Zink bound [43] for β_q^b, β_q^ω, and $\beta_q^{\mathcal{R}}$. However, these lower bounds do not appear to be tight. In particular, it would be of pragmatic interest to find lower bounds which are non-trivial when the field size q is small or the rate R is high. Upper bounds would also be desirable.

(7) Find improved lower bounds and good upper bounds for β_q^b, β_q^ω, and $\beta_q^{\mathcal{R}}$.

One might also consider sequences of $[n, k, e, b]_q$-codes for which b is allowed to increase with n: we require only that b must be bounded by a polynomial function of n. Such codes fit within a reasonable definition of efficient non-unique decoding. With reference to Section 9, corresponding to such codes we can define a function $\beta_q^{\mathcal{P}}$ which is analogous to and evidently at least as large as β_q^ω. Hence we pose the following problem:

(8) Find bounds which distinguish $\beta_q^{\mathcal{P}}$ from β_q^ω.

Finally, all our results are for AG-codes. It would be of interest to derive analogous results for other classes of error-correcting codes.

Acknowledgements

The authors would like to thank Ian Blake, Ilya Dumer, Tom Høholdt, Ralf Kötter, Ruud Pellikaan, Madhu Sudan, and Alexander Vardy for helpful comments and discussions.

References

[1] S. Ar, R. Lipton, R. Rubinfeld, and M. Sudan. Reconstructing algebraic functions from mixed data. In *Proc. 33rd FOCS*, pages 503–512, 1992.

[2] E.R. Berlekamp. Bounded distance + 1 soft decision Reed-Solomon decoding. *IEEE Trans. Inform. Theory*, 42:704–720, 1996.

[3] P. Bürgisser, M. Clausen, and M.A. Shokrollahi. *Algebraic Complexity Theory*, volume 315 of *Grundlehren der Mathematischen Wissenschaften.* Springer Verlag, Heidelberg, 1996.

[4] H. Cohen. *A Course in Computational Algebraic Number Theory*, volume 138 of *Graduate Texts in Mathematics.* Springer Verlag, 1993.

[5] C. Dahl. Fast decoding of codes from algebraic curves. *IEEE Trans. Inform. Theory*, 40:223–229, 1994.

[6] I.I. Dumer. Two algorithms for the decoding of linear codes. *Problems Inform. Transmission*, 25:17–23, 1989.

[7] I. Duursma. Algebraic decoding using special divisors. *IEEE Trans. Inform. Theory*, 39:694–698, 1993.

[8] D. Ehrhard. Achieving the designed error capacity in decoding algebraic-geometric codes. *IEEE Trans. Inform. Theory*, 39:743–751, 1993.

[9] G.L. Feng and T.R.N. Rao. Decoding algebraic-geometric codes up to the designed minimum distance. *IEEE Trans. Inform. Theory*, 39:37–45, 1993.

[10] A. Garcia and H. Stichtenoth. A tower of Artin-Schreier extensions of function fields attaining the Drinfeld-Vladut bound. *Invent. Math.*, 121:211–222, 1995.

[11] J. von zur Gathen and E. Kaltofen. Factorization of multivariate polynomials over finite fields. *Math. Comp.*, 45:251–261, 1985.

[12] O. Goldreich, R. Rubinfeld, and M. Sudan. Learning polynomials with queries: the highly noisy case. In *Proc. 36th FOCS*, pages 294–303, 1995.

[13] V.D. Goppa. Codes on algebraic curves. *Sov. Math. Dokl.*, 24:170–172, 1981.

[14] V.D. Goppa. *Geometry and Codes.* Kluwer Academic Publishers, 1988.

[15] M. Huang and D. Ierardi. Efficient algorithms for the Riemann-Roch problem and for addition in the Jacobian of a curve. In *Proc. 32nd FOCS*, pages 678–687, 1991.

[16] M. Huang and D. Ierardi. Efficient algorithms for the Riemann-Roch problem and for addition in the Jacobian of a curve. *J. Symb. Comp.*, 18:519–539, 1994.

[17] J. Justesen, K.J. Larsen, A. Havemose, H.E. Jensen, and T. Høholdt. Construction and decoding of a class of algebraic geometry codes. *IEEE Trans. Inform. Theory*, 35:811–821, 1989.

[18] J. Justesen, K.J. Larsen, H.E. Jensen, and T. Høholdt. Fast decoding of codes from algebraic plane curves. *IEEE Trans. Inform. Theory*, 38:111–119, 1992.

[19] S. Landau. Factoring polynomials over algebraic number fields. *SIAM J. Comput.*, 14:184–195, 1985.

[20] S. Lang. *Algebra.* Addison-Wesley, third edition, 1993.

[21] A.K. Lenstra. Factoring polynomials over algebraic number fields. Technical Report IW213/82, Dept. Comp. Science, Stichting Mathematisch Centrum, Amsterdam, 1982.

[22] A.K. Lenstra. Factoring multivariate polynomials over finite fields. In *Proc. 15th STOC*, pages 189–192, 1983.

[23] J.H. van Lint. *Introduction to Coding Theory*, volume 86 of *Graduate Texts in Mathematics*. Springer Verlag, 1982.

[24] Yu.I. Manin. What is the maximum number of points on a curve over $\mathbb{F}_2$? *J. Fac. Sci. Tokio*, 28:715–720, 1981.

[25] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[26] M.E. O'Sullivan. Decoding of codes defined by a single point on a curve. *IEEE Trans. Inform. Theory*, 41:1709–1719, 1995.

[27] R. Pellikaan. On a decoding algorithm for codes on maximal curves. *IEEE Trans. Inform. Theory*, 35:1228–1232, 1989.

[28] A. Peralta. A simple and fast probabilistic algorithm for computing square roots modulo a prime number. *IEEE Trans. Inform. Theory*, 32:846–847, 1986.

[29] S.C. Porter, B.Z. Shen, and R. Pellikaan. Decoding geometric Goppa codes using an extra place. *IEEE Trans. Inform. Theory*, 38:1663–1676, 1992.

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

[31] M.A. Shokrollahi. Codes on Hermitian curves. In Th. Beth and M. Clausen, editors, *Proc. AAECC-4*, number 307 in Lecture Notes in Computer Science, pages 168–176. Springer Verlag, 1988.

[32] M.A. Shokrollahi. Beiträge zur Codierungs- und Komplexitätstheorie mittels algebraischer Funktionenkörper. *Bayreuth. math. Schriften*, 39:1–236, 1992.

[33] M.A. Shokrollahi. Efficient randomized generation of algorithms for multiplication in certain finite fields. *Comp. Compl.*, 2:67–96, 1992.

[34] M.A. Shokrollahi and K. Werther. Generation of optimal bilinear multiplication algorithms: theory and implementation. Technical report, Department of Computer Sciene, Universität Bonn, 1992.

[35] V.M. Sidel'nikov. Decoding the Reed-Solomon code when the number of errors is greater than $(d-1)/2$, and zeros of polynomials in several variables. *Problems Inform. Transmission*, 30:44–59, 1994.

[36] J.H. Silverman. *The Arithmetic of Elliptic Curves*, volume 106 of *Graduate Texts in Mathematics*. Springer Verlag, 1986.

[37] A.N. Skorbogatov and S.G. Vladut. On decoding of algebraic-geometric codes. *IEEE Trans. Inform. Theory*, 36:1051–1060, 1990.

[38] H. Stichtenoth. A note on Hermitian codes over $GF(q^2)$. *IEEE Trans. Inform. Theory*, 34:1345–1348, 1988.

[39] H. Stichtenoth. *Algebraic Function Fields and Codes*. Universitext. Springer Verlag, 1993.

[40] M. Sudan. Maximum likelihood decoding of Reed-Solomon codes. In *Proc. 37th FOCS*, pages 164–172, 1996.

[41] M. Sudan. Decoding of Reed-Solomon codes beyond the error-correction bound. *J. Compl.*, 13:180–193, 1997.

[42] M.A. Tsfasman and S.G. Vladut. *Algebraic-Geometric Codes*. Mathematics and Its Applications. Kluwer Academic Publishers, Dordrecht, 1991.

[43] M.A. Tsfasman, S.G. Vladut, and Th. Zink. Modular curves, Shimura curves, and Goppa codes better than the Varshamov-Gilbert bound. *Math. Nachrichten*, 109:21–28, 1982.

[44] S.G. Vladut. On the decoding of algebraic-geometric codes over $\mathbb{F}_q$ for $q \geq 16$. *IEEE Trans. Inform. Theory*, 36:1461–1463, 1990.

[45] H. Wasserman. Reconstructing randomly sampled multivariate polynomials from highly noisy data. In *Proc. 9th SODA*, 1998.

[46] L.R. Welch and E.R. Berlekamp. Error correction for algebraic block codes. U.S. Patent 4,633,470, issued Dec. 30, 1986.

M. Amin Shokrollahi
International Computer Science Institute
1947 Center Street, Suite 600
Berkeley, CA 94704–1198
`amin@icsi.berkeley.edu`
`http://www.icsi.berkeley.edu/~amin`

Hal Wasserman
Computer Science Division
University of California
Berkeley, CA 94720
`halw@cs.berkeley.edu`
`http://http.cs.berkeley.edu/~halw`