Improved Power Decoding of
Algebraic Geometry Codes

Sven Puchinger, Johan Rosenkilde, Grigory Solomatov
Department of Applied Mathematics and Computer Science,
Technical University of Denmark (DTU), Denmark

Abstract—Power decoding is a partial decoding paradigm for
arbitrary algebraic geometry codes for decoding beyond half
the minimum distance, which usually returns the unique closest
codeword, but in rare cases fails to return anything. The original
version decodes roughly up to the Sudan radius, while an
improved version decodes up to the Johnson radius, but has so far
been described only for Reed–Solomon and one-point Hermitian
codes. In this paper we show how the improved version can be
applied to any algebraic geometry code.

Index Terms—Algebraic Geometry Codes, Power Decoding

I. INTRODUCTION

A. Related work

Power decoding was originally proposed in [1] and [2]
as a method for decoding Reed–Solomon codes beyond half
the minimum distance, achieving roughly the same decoding
radius as Sudan’s list decoder [3]. It is a partial unique decoder,
meaning that it either returns a unique decoding solution, or
declares failure. It has been demonstrated through numerical
simulations that power decoding has a very low probability of
failure for random errors; there are proofs of this for small
parameter choices [2], [4], but the general case remains to be
an open problem.

In [5] it was shown how to incorporate a multiplicity param-
eter into power decoding, similar to that of Guruswami–Sudan
(GS) algorithm [6], thereby matching the latter’s decoding
radius, called the Johnson radius. Both the original and the
improved power decoding with multiplicity has been applied to
one-point Hermitian codes, which belong to the large family of
algebraic geometry (AG) codes that were introduced by Goppa
in 1981 [7], [8]. They are a natural generalization of Reed–
Solomon codes and some of their subfamilies have been shown
to have minimal distance that beats the Gilbert–Varshamov
bound [9].

At the heart of power decoding lies a set of “key equations”,
which can be solved by linear algebra, or by faster, more struc-
tured approaches, e.g. by generalizing the Berlekamp–Massey
or Euclidean algorithm, see e.g. [10], [11] and their references.
This step is algebraically connected with to the “interpolation
step” of the GS algorithm, which may be computed in similar
complexity [12], [11].

After interpolation, the GS algorithm proceeds with a “root-
finding step”, which is not required by power decoding. For
Reed–Solomon and one-point Hermitian codes, root-finding is
asymptotically faster than the interpolation step \(^{1}\) [13], [12],
[14], but in hardware-implementations root-finding can take
up significant circuit area [15]. For more general AG codes,
the root-finding step is not widely studied, and it is unclear
whether it is faster than interpolation; e.g. [16] describes root-
finding for one-point codes which is not superior to performing

\(^{1}\)Except if the code length is very short compared to the field size: a sub-
routine of the root-finding step is to find \(F_q\)-roots in \(F_q[x]\) polynomials.

interpolation using linear algebra. Further, both power de-
coding and the GS algorithm generalize to interleaved codes, and
here the root-finding step is currently the bottleneck in the GS
algorithm [17], and so power decoding is asymptotically faster
[18].

Recently, [19] generalized the original power decoding to
apply to general AG codes. Their work was principally moti-
vated by their new variant of (the original) power decoding,
called power error-locating pairs, which may be applied to
attack AG-code-based cryptosystems in a setting where it
seems the GS decoder can not be used.

B. Contributions

In this paper we:

• formulate improved power decoding [5] in the language
  of function fields and adapt it to general AG codes,

• show how the resulting key equations can be solved using
  linear algebra, and

• derive the decoding radius for the proposed method
  under mild assumptions on the code and verify it using
  numerical simulations.

The most important takeaway is that our decoding radius
coincides with the one from [20] for Reed–Solomon codes and
the one from [21] for one-point Hermitian codes, achieving
the Johnson radius for suitable decoding parameters. Though
the computational complexity of our decoder can likely be
improved by replacing linear algebra with more advanced
techniques, this remains outside of the scope of this paper.

Our function field modelling revolves around “using an
extra place” \(P_\infty\) as in Pellikaan [22], which allows controlling
the “size” of functions by their pole order at \(P_\infty\). Our key
equations are in the style of Gao [23], as introduced for power
decoding in [4].

We present evidence supporting that our decoder achieves a
decoding radius similar to the GS algorithm in two ways: we
theoretically derive a radius at which the decoder will surely
fail, and then we see in simulations that for fewer random
errors, the decoder seems to almost always succeed. We have
no theoretic bound on this failure probability, which has so far
proved very challenging even in the case of Reed–Solomon
codes, see e.g. [20]. On the other hand, advantages of the power
decoding compared to GS algorithm include:

• The power decoder is structurally simpler than the GS
  algorithm since it does not have a root-finding step,
  making it easier to implement, possibly faster, and with
  less footprint in hardware.

• It is reasonable that a generalization to interleaved AG
codes is possible as in [18], [21].

• Potentially apply power decoding in a more efficient
  message attack on AG-based cryptosystems than in [19],
  possibly by first reformulating it in terms of power error-
  locating pairs.
II. PRELIMINARIES

In this section we briefly introduce AG codes as evaluation codes and formulate the decoding problem that we wish to solve. We assume that the reader is familiar with the language of function fields, which is presented in great detail in [24].

Let \( F \) be a function field having genus \( g \) over a base field \( \mathbb{F} \). Let \( P_1, \ldots, P_n, P_\infty \) be rational places of \( F \), and fix two divisors \( D = P_1 + \cdots + P_n \) and \( G \) with \( \deg G = \gamma \), such that \( \text{supp} \, G \cap \text{supp} \, D = \emptyset \). The code that we will be considering is

\[
C_{D,G} = \{ (f(P_1), \ldots, f(P_n)) \in \mathbb{F}^n \mid f \in \mathcal{L}(G) \},
\]

where \( \mathcal{L}(G) = \{ f \in F \mid (f) \geq -G \} \cup \{0\} \) is the Riemann-Roch space of \( G \). The dimension of this code is \( k = \#(G) \), where \( \#(A) \) denotes the dimension of \( \mathcal{L}(A) \) for any divisor \( A \); and the code's minimum distance is bounded from below by the designed minimum distance \( d^* = n - \gamma \).

The problem that we wish to address is as follows:

**Problem II.1.** Let \( f \in \mathcal{L}(G) \) be a message and let \( e \in C_{D,G} \) be the corresponding codeword. Given the received word

\[
r = (r_1, \ldots, r_n) = e + e,
\]

where \( e = (e_1, \ldots, e_n) \in \mathbb{F}^n \) is some unknown error, recover \( f \).

In the next section, in the traditional spirit of power decoding, we show how Problem II.1 can be reformulated as a highly non-linear system of equations.

III. THE KEY EQUATIONS

In this section we formulate the so-called key equations that lie at the heart of power decoding using the language of function fields.

We begin by defining the error locator and the received word interpolator, following up with some results about their sizes.

**Definition III.1.** If \( r = (r_1, \ldots, r_n) \) is the received word, then an \( r \)-interpolator of degree \( \rho \) is an element

\[
R \in \mathcal{L}(G + \rho P_\infty) \quad \text{such that} \quad R \notin \mathcal{L}(G + (\rho - 1)P_\infty),
\]

and \( R(P_i) = r_i \) for \( i = 1, \ldots, n \).

**Lemma III.2.** There exists an \( r \)-interpolator \( R \) with degree \( \rho \) satisfying \( \rho \leq n - \gamma + 2g - 1 \).

**Proof.** Consider the \( \mathbb{F} \)-linear map

\[
\phi: \mathcal{L}(G + \rho P_\infty) \to \mathbb{F}^n
\]

\[
h \mapsto (h(P_1), \ldots, h(P_n))
\]

We are guaranteed that \( R \) exists if the dimension of the image of \( \phi \), which is given by the rank-nullity theorem as

\[
\dim \, \text{im} \, \phi = \#(G + \rho P_\infty) - \#(G + \rho P_\infty - D),
\]

is \( n \). Indeed, Riemann’s theorem theorem says that

\[
\#(G + \rho P_\infty) \geq \gamma + \rho + 1 - g,
\]

and if \( \rho \geq n - \gamma + 2g - 1 \), then [24, Thm 1.5.17] promises that

\[
\#(G + \rho P_\infty - D) = \gamma + \rho - n + 1 - g,
\]

which shows that the image of \( \phi \) has dimension \( n \).

**Definition III.3.** Let \( E = \sum_{i \in \mathcal{E}} P_i \), where \( \mathcal{E} = \{ i \mid e_i \neq 0 \} \) is the set of error positions. An error locator with multiplicity \( s \) and degree \( \lambda \), is an element

\[
\Lambda_s \in \mathcal{L}(\lambda_s P_\infty - sE) \quad \text{such that} \quad \Lambda_s \notin \mathcal{L}(\lambda_s - 1)P_\infty - sE).
\]

**Lemma III.4.** For any \( \lambda_s \geq g \) there exists an error locator with degree at most \( \lambda_s \), however, if \( \lambda_s < g \), then no such error locator exists.

**Proof.** If \( \lambda_s \geq g \) + \( g \) then the Riemann-Roch theorem promises that

\[
\#(\lambda_s P_\infty - sE) \geq \lambda_s - g \geq 0,
\]

which guarantees existence of an error locator with degree at most \( \lambda_s \).

If on the other hand \( \lambda_s < g \), then \( \Lambda_s \) must have more zeroes than poles, which is impossible.

It will be convenient to define the following divisors:

\[
V_t = \lambda_s P_\infty + tG,
Q_t = \lambda_s P_\infty + t(G + P_\infty) \quad \text{and} \quad W_j = \lambda_s P_\infty + j(G + P_\infty - D),
\]

for \( j = 0, \ldots, s - 1 \) and \( t = 1, \ldots, \ell \).

The next lemma relates these divisors to \( f, \Lambda_s \) and \( R \).

**Lemma III.5.** If \( f \) is the sent message, \( R \) is an \( r \)-interpolator and \( \Lambda_s \) is an error locator, then for \( t = 1, \ldots, \ell \) it holds that

\[
\Lambda_s f^t \in \mathcal{L}(V_t),
\]

and for \( j = 0, \ldots, s - 1 \) it holds that

\[
\Lambda_s (f - R)^j R^{-j} \in \begin{cases} \mathcal{L}(W_j + (t - j)(G + P_\infty)) & \text{if } j < s \\ \mathcal{L}(Q_t - sD) & \text{if } j \geq s \end{cases}
\]

**Proof.** The first claim is obvious. For the second claim observe that

\[
f - R \in \mathcal{L}(G + \rho P_\infty - D + E).
\]

If \( j < s \), then

\[
\Lambda_s (f - R)^j \in \mathcal{L}(\lambda_s P_\infty + j(G + P_\infty) - sE - j(D - E)) \subset \mathcal{L}(\lambda_s P_\infty + j(G + P_\infty - D) - (s - j)E) \subset \mathcal{L}(W_j).
\]

If on the other hand \( j \geq s \), then

\[
\Lambda_s (f - R)^j R^{-j} \in \mathcal{L}(\lambda_s P_\infty + t(G + P_\infty) - sE - j(D - E)) \subset \mathcal{L}(\lambda_s P_\infty + t(G + P_\infty - D) - sE - (s - j)E) = \mathcal{L}(Q_t - sD).
\]

We are now ready to present the key equations, which express an algebraic relationship between the sent message \( f \), an \( r \)-interpolator \( R \) and a corresponding error locator \( \Lambda_s \).

**Theorem III.6 (The Key Equations).** For \( t = 1, \ldots, \ell \)

\[
\Lambda_s f^t - \sum_{j=0}^{\min\{t,s-1\}} \binom{t}{j} \Lambda_s (f - R)^j R^{-j} \in \begin{cases} \{0\} & \text{if } 1 \leq t \leq s - 1 \\ \mathcal{L}(Q_t - sD) & \text{if } s \leq t \leq \ell \end{cases}
\]
Proof. We have that
\[ \Lambda_s f^t = \Lambda_s ((f - R) + R)^t = \sum_{j=0}^{t} \binom{t}{j} \Lambda_s (f - R)^j R^{t-j}, \]
which proves the claim when \(1 \leq t \leq s-1\), while if \(s \leq t \leq \ell\), then it follows from Lemma III.5 that
\[ \Lambda_s f^t = \min_{t,s-1} \sum_{j=0}^{t} \binom{t}{j} \Lambda_s (f - R)^j R^{t-j} = \sum_{j=s}^{t} \binom{t}{j} \Lambda_s (f - R)^j R^{t-j} \in \mathcal{L}(Q_t - sD). \]
\[ \square \]

Since (1) is satisfied by the sent message, it makes sense to look for a solution of (1) in order to address Problem II.1. We explain how this can be done in the next section.

IV. SOLVING THE KEY EQUATIONS

In this section we show how to solve the following problem, which is motivated by Theorem IV.3 and Problem II.1:

**Problem IV.1.** Consider an instance of Problem II.1. Fix \( \lambda_s \geq 0 \) and let \( R \) be an \( r \)-interpolator. Find \( \phi_t \in \mathcal{L}(V_t) \) for \( t = 1, \ldots, \ell \) and \( \psi_j \in \mathcal{L}(W_j) \) for \( j = 0, \ldots, s - 1 \), not all zero, such that
\[ \phi_t = \min_{t,s-1} \sum_{j=0}^{t} \binom{t}{j} \psi_j R^{t-j} = \begin{cases} \{0\} & 1 \leq t \leq s - 1 \\ \mathcal{L}(Q_t - sD) & s \leq t \leq \ell \end{cases} \]

According to Theorem III.6, if \( f \) is the sent message and there exists an error locator \( \Lambda_s \) with degree at most \( \lambda_s \), then among the possible solutions of Problem IV.1 we would find
\[ \phi_t = \Lambda_s f^t, \quad \psi_j = \Lambda_s (f - R)^j. \tag{2} \]

Since \( \deg \Lambda_s \) is roughly proportional to the number of errors (see Lemma III.4), this means that a solution of Problem IV.1 of the type (2) for the smallest-possible \( \lambda_s \) corresponds to a codeword that is among the closest codewords to our received word.\(^2\) It might be that Problem IV.1 for small \( \lambda_s \) has solutions that do not correspond to an error locator/message pair. This might happen since Problem IV.1 poses weaker constraints on its solutions compared to (1). Previous works on (improved) power decoding indicate that such “spurious” solutions only occur with very small probability. Also our simulation results in Section VI indicate this. We will get back to types of failure in the next section.

These arguments motivate our decoding strategy: Find a solution of Problem IV.1 for the smallest \( \lambda_s \) for which the problem has a solution. If we are successful, we get a solution of Problem IV.1 of the form in (2) (or a scalar multiple thereof), where \( \Lambda_s \) and \( f \) correspond to a close codeword to \( r \). We can extract \( f \), and thus \( c \) from this solution by division \( \frac{\phi_t}{\psi} = \frac{\Lambda_s f^t}{\Lambda_s (f - R)^j} = f \).

Since computational complexity is not a primary concern here, we describe in this section how to solve the problem for a fixed \( \lambda_s \). The “smallest solution” can then be found by iterating \( \lambda_s \) from 0 to the maximal possible value for which we expect to be able to decode.

The entire decoding strategy is outlined in Algorithm 1.

**Algorithm 1:** Decoder

**Input:** Received word \( r \in \mathbb{F}^n \) and positive integers \( s \leq \ell \)

**Output:** \( f \in \mathcal{L}(G) \) such that \( c = (f(P_1), \ldots, f(P_n)) \) is a codeword with corresponding minimal error locator degree; or “decoding failure”.

1. for \( \lambda_s = 0, \ldots, sn + g \) do
2. if Problem IV.1 with parameter \( \lambda_s \) has a unique solution up to scalar multiples then
3. \[ \{ (\phi_t)_{t=1}^{\ell}, (\psi_j)_{j=0}^{s-1} \} \leftarrow \text{a solution of Problem IV.1} \]
4. \( f \leftarrow \frac{\phi_t}{\psi} \)
5. if \( f \in \mathcal{L}(G) \) and \( \psi \) is a valid error locator for the error \( \hat{\epsilon} = r - \hat{c} \), where \( \hat{c} \) is the codeword corresponding to \( f \) then
6. return \( f \)
7. else
8. return “decoding failure”
9. return “decoding failure”

In the remainder of this section we give an explicit construction of a matrix \( U \) over \( \mathbb{F} \) whose right kernel contains vectors from which solutions to Problem IV.1 can be obtained, i.e., how to implement Lines 2 and 3 of Algorithm 1.

For any divisor \( A \) let \( \beta_A \in \mathcal{L}(A)^{l(A)} \) denote a vector whose entries form any \( \mathbb{F} \)-basis of \( \mathcal{L}(A) \) such that for any two divisors \( A \) and \( B \) it holds that
\[ \mathcal{L}(A) = \mathcal{L}(B) \iff \beta_A = \beta_B. \]

For any \( h \in \mathcal{L}(A) \) let \( h_A \in \mathbb{F}^{l(A)} \) be the unique vector such that \( \beta_A \cdot h_A = h \), and for any \( p \in \mathcal{L}(B-A) \) let \( P_{B,A} \in \mathbb{F}^{l(B-A)\times l(A)} \) be the matrix whose \( i \)-th column is \( (p_{a_i})_B \), where \( (a_1, \ldots, a_{l(A)}) = \beta_A \). It is easy to see that
\[ \beta_{B_{P_{B,A}}} h_A = p \beta_A \cdot h_A = ph \in \mathcal{L}(B). \]

In order to construct \( U \) we will need a satisfactory received word interpolator \( R \) which can be for example be obtained as follows: If \( M \in \mathbb{F}^{n \times (l(H))} \), where \( H = \rho P_\infty + G \), is the matrix whose \( i \)-th column is
\[ [b_1(P_1) \ldots b_l(P_n)]^T, \]
where \( (b_1, \ldots, b_l(H)) = \beta_H \), then \( R = \beta_H \cdot z \), where \( z \) is any solution to the linear system \( M z = r \).

The next lemma together with the subsequent Corollary IV.3 give an explicit construction of \( U \) and show its relation to Problem IV.1.

**Lemma IV.2.** Let
\[ u = (u_1, \ldots, u_l, w_0, \ldots, w_s) \in \mathbb{F}^n, \]
where
\[ v_r = (\phi_r)v_r, \]
\[ w_j = (\psi_j)w_j \]
and
\[ \nu = \sum_r l(V_r) + \sum_j l(W_j) \]
for \( r = 1, \ldots, \ell \) and \( j = 0, \ldots, s - 1 \), where the \( \phi_r, \psi_j \) are a solution to Problem IV.1. If \( U_t \in \mathbb{F}_q^{t \times \nu} \) is the matrix defined by
\[ U_t = [V_{t,1} \cdots | V_{t,\nu} | W_{t,0} \cdots | W_{t,s-1}], \]
where
\[ V_{t,r} = \begin{cases} 1_{Q_t},v_r & \text{if } t = r \\ 0_{Q_t},v_r & \text{if } t \neq r \end{cases} \]
and
\[ W_{t,j} = \begin{cases} (\sum_{j=0}^{\nu} l(W_{j,t}))_{Q_t} & \text{if } j \leq t \\ 0_{Q_t},w_j & \text{if } j > t. \end{cases} \]
for \( t = 1, \ldots, \ell \), then
\[ \beta_{Q_t}U_t u \subset \begin{cases} \emptyset & \text{if } 1 \leq t \leq s - 1 \\ \mathcal{L}(Q_t - sD) & \text{if } s \leq t \leq \ell. \end{cases} \]

Proof. Simply observe that
\[ \beta_{Q_t}U_t u = \beta_{Q_t} \sum_{t=1}^{\ell} V_{t,r}v_r + \beta_{Q_t} \sum_{j=0}^{s-1} W_{t,j}w_j \]
and
\[ = \phi_{Q_t} - \sum_{j=0}^{s-1} \left( \sum_{j=0}^{s-1} l(W_{j,t}) \right) \psi_j R^{t-j}. \]

Corollary IV.3 (of Lemma IV.2). If
\[ U = \begin{bmatrix} K_1 U_1 \\ \vdots \\ K_\ell U_\ell \end{bmatrix}, \]
where
\[ K_t = \begin{cases} 1_{Q_t},Q_t & \text{if } 1 \leq t \leq s - 1 \\ \text{left kernel matrix of } 1_{Q_t},Q_t - sD & \text{if } s \leq t \leq \ell, \end{cases} \]
and \( u \) is as in Lemma IV.2, then
\[ U u = 0. \tag{3} \]

In the next section we describe the conditions under which decoding is expected to fail.

V. DECODING RADIUS

We turn to investigating the decoding performance of the proposed decoder. Since the decoder returns at most one codeword while attempting to decode beyond half the minimum distance, it must obviously fail for certain received words. Theoretically bounding this probability seems a difficult problem even for the simplest case of Reed–Solomon codes, and only partial results are known \cite{1}, \cite{5}. We follow the approach of several previous papers on power decoding: we derive theoretically a number of errors at which our decoder is always guaranteed to fail, and we define this to be the “decoding radius”; this name will be supported by simulations in Section VI which indicates that the decoding algorithm succeeds with high probability whenever fewer, random errors occur.

It can be seen from Algorithm 1 that our decoder declares a decoding failure if and only if there is no value of \( \lambda_s \) for which the matrix \( U \) in Corollary IV.3 has a right kernel of \( \mathbb{F}_q \)-dimension exactly 1.

On the other hand, when we do not declare decoding failure, the returned message candidate \( f \in \mathcal{L}(G) \) must correspond to the closest codeword to the received word, as in the discussion in the preceding section.

This notion of decoding failure motivates a natural bound for when we – simply by linear algebraic arguments – would know a priori that the decoder will fail in the generic case when the smallest error locator corresponding to a codeword has degree \( \lambda_s = s|E| + g \) (and not smaller), where \( |E| \) is the distance of the codeword to the received word:

**Definition V.1.** For a given code \( C \), the decoding radius of power decoding, denoted \( \tau_{\max} \), is the greatest value of \( \tau \) such that for \( \lambda_s = s|\hat{E}| + g \), if \( \epsilon, \nu \) are the row resp. column dimensions of the matrix \( U \) of Lemma IV.2, then \( \nu > \epsilon + 1. \)

Since the proposed decoder will typically return no solutions (and therefore fail) when \( |\hat{E}| > \tau \), the decoding radius tells us how many errors we should at most expect to correct.\(^3\)

We reiterate that, at this point we have given no indications that the decoder should usually succeed up to \( \tau_{\max} \) errors, but we address this by simulations in Section VI.

For a given code \( C \), one may compute \( \tau_{\max} \) exactly, but since the equations involve dimensions of sequences of Riemann–Roch spaces, it seems impossible to give a general precise closed form. However, we may lower-bound these numbers using common function field tools:

**Lemma V.2.** For \( t = 1, \ldots, \ell \) and \( j = 0, \ldots, s - 1 \) it holds that
\[ l(V_t) \geq \lambda_s + t - g + 1, \]
\[ l(W_j) \geq \lambda_s + j(\gamma + \rho - n) - g + 1, \]
\[ l(Q_t - sD) \geq \lambda_s + t(\gamma + \rho) - sn - g + 1, \]
and if \( \gamma \geq 2g - 2 \), then
\[ l(Q_t) = \lambda_s + t(\gamma + \rho) - g + 1. \]

Proof. The lower bounds on \( l(V_t) \), \( l(W_j) \) and \( l(Q_t - sD) \) are given by the Riemann-Roch theorem, while the exact value of \( l(Q_t) \) is given by \cite[Thm 1.5.17]{24} since \( \gamma \geq 2g - 2. \)

The next lemma gives bounds on the dimensions of \( U. \)

**Lemma V.3.** If \( \gamma \geq 2g - 2 \), then \( U \in \mathbb{F}_q^{\times \nu} \), where
\[ \epsilon \leq (s - 1)(\lambda_s + g + 1) + \frac{(s-1)(\gamma + \rho) + (\ell - s + 1)sn}{2}, \]
\[ \nu \geq (\ell + s)(\lambda_s - g + 1) + \frac{(\ell + 1)s}{2}(\gamma + \rho - n). \]

Proof. From Lemma V.2 it follows that
\[ \nu = \sum_{t=1}^{\ell} l(V_t) + \sum_{j=0}^{s-1} l(W_j) \]
\[ \geq \sum_{t=1}^{\ell} (\lambda_s + t\gamma - g + 1) + \sum_{j=0}^{s-1} (\lambda_s + j(\gamma + \rho) - jn - g + 1) \]
\[ = (\ell\lambda_s - g + 1) + \frac{(\ell + 1)s}{2}\gamma + s(\lambda_s - g + 1) + \frac{(s-1)n}{2}(\gamma + \rho - n). \]
\(^3\)It might succeed in rare cases in which the error locator has surprisingly low degree. Our simulation results suggest that this happens with very small probability.
Similarly, observing that \( K_t \) must have exactly
\[
|Q_t| - |Q_t - sD|
\]
rows for \( t = s - 1, \ldots, \ell \), it holds that
\[
\epsilon = \sum_{t=1}^{\ell-s} (|Q_t| - |Q_t - sD|) \\
\leq \sum_{t=1}^{\ell-s} (\lambda_s + t(\gamma + \rho) - g + 1) + \sum_{t=s}^{\ell} s n \\
= (s-1)(\lambda_s - g + 1) + \left(\frac{s-1}{2}\right)(\gamma + \rho) + (\ell - s + 1) s n .
\]

With the aid of Lemma V.3 we can deduce a lower bound on values of \( \lambda_s \) for which decoding must fail.

**Lemma V.4.** It holds that (3) has at least two linearly independent solutions (and decoding fails) if
\[
\lambda_s > s \frac{2\ell + s + 1}{2(\ell + 1)} n - \frac{\ell}{24} g + \frac{\ell}{n(\ell + 1)} + g . \tag{4}
\]

**Proof.** Keeping the notation from Lemma V.3 we have that (3) has at least two linearly independent solutions if
\[
\nu > \epsilon + 1 ,
\]
which is equivalent to (4).

**Corollary V.5.** The decoding radius as defined in Definition VI.1 is given by
\[
\tau_{\text{max}}(\ell, s) := \left[ \frac{2\ell + s + 1}{2(\ell + 1)} n - \frac{\ell}{24} g + \frac{\ell}{n(\ell + 1)} \right] .
\]

**Remark V.6.** The decoding radius \( \tau_{\text{max}}(\ell, s) \) coincides with the one in [20] in the special case of Reed–Solomon, and the one in [21] for one-point Hermitian codes.

**Remark V.7.** Although this definition of failure is convenient for theoretical analysis it means our decoder must try every sensible (sufficiently small) value of \( \tau \) before declaring failure, which is computationally inefficient. In this paper, our primary concern is not computational complexity, but we will remark that in practice, one only needs to use the largest sensible value of \( \tau \): we then pick any non-zero vector in the right kernel of \( U \) and check if the corresponding message \( f \) is in \( L(G) \). This is how the simulations in Section VI have been conducted.

### A. Parameter Choice & Asymptotic Behavior

The following theorem shows that the decoding radius achieves the Johnson radius asymptotically for large parameters \( \ell \) and \( s \). It also implies a good practical choice of the parameters \( \ell \) and \( s \).

**Theorem V.8.** Define the sequence \((\ell_i, s_i)\) by \( i \in \mathbb{Z}_{\geq 0} \) as \( \ell_i = i \) and \( s_i = \lceil \sqrt{i} \rceil + 1 \). Then, for \( i \to \infty \), we have
\[
\tau_{\text{max}}(\ell_i, s_i) = n \left( 1 - \sqrt{\frac{i}{n}} - O(\frac{1}{i}) \right) .
\]

**Proof.** This is a special case of [21, Theorem 5].

### VI. NUMERICAL RESULTS

In this section we present Monte-Carlo simulation results of the proposed decoder for a few different AG codes in order to experimentally verify our hypothesis that Algorithm I corrects up to \( \tau_{\text{max}}(\ell, s) \) errors with high probability and fails with high probability above.

To be precise, for a given code and radius \( \tau \), we test the following:

- Draw a random message \( f \) and encode it into \( c \)
- Draw an error \( e \) uniformly at random from the set of vectors in \( \mathbb{F}_q^n \) of Hamming weight \( \tau \)
- Decode \( r = c + e \) using Algorithm I
- If the algorithm does not return a decoding failure and the returned message polynomial is exactly \( f \), then count it as success.

Note that the simulation might be unsuccessful also in cases in which Algorithm I returns a valid close codeword, but not the transmitted one. This is called a miscorrection, and typically occurs only rarely (see, e.g., [25] for RS codes).

The above described simulation was tested on a few one- and two-point codes over two well known function fields:

- The Hermitian function field is defined by the equation
  \[
  H_q : y^q + y = x^{q+1}
  \]
  over \( \mathbb{F}_{q^2} \), i.e. the finite field of cardinality \( q^2 \). It has genus \( \frac{1}{2}q(q-1) \) and \( q^3 + 1 \) rational places.
- The Suzuki function field is defined by the equation
  \[
  S_q : y^q + y = x^{q^0}(x^q + x)
  \]
  over \( \mathbb{F}_{q^4} \), where \( q = 2^b \) and \( q^2 + 1 \) rational places.
- The function from [26, Lemma 3.2] field is defined by
  \[
  T_q : y^q + y = x^{q^0} - 1 + x
  \]
  over \( \mathbb{F}_{q^2} \). It has genus \( (q-1)^2 \) and \( q^3 - q^2 + 2q \) rational places.

Table I contains simulation results for various function field family, code, and decoder parameters. All results confirm our hypothesis.

| Curve | \( P \) | \( \gamma \) | \( n \) | \( k \) | \( d^* \) | \( \ell \) | \( s \) | \( \tau \) | OFR |
|-------|-------|-------|-------|-------|-------|-------|-------|------|-----|
| \( H_4 \) | 4^2 | 15 | 64 | 10 | 49 | 4 | 2 | 29+ | 0.00 | 10^2 |
| \( H_4 \) | 4^2 | 10 | 5 | 10 | 5 | 4 | 2 | 28+ | 1.00 | 10^2 |
| \( H_5 \) | 5^2 | 55 | 125 | 46 | 70 | 5 | 2 | 36+ | 0.00 | 10^2 |
| \( H_5 \) | 5^2 | 30 | 25 | 124 | 46 | 69 | 3 | 2 | 35+ | 0.94 | 10^2 |
| \( S_4 \) | 2^4 | 12 | 24 | 12 | 2 | 2 | 2 | 5+ | 0.00 | 10^4 |
| \( S_4 \) | 2^4 | 26 | 30 | 24 | 12 | 2 | 2 | 5+ | 0.00 | 10^4 |
| \( S_4 \) | 2^4 | 4 | 24 | 20 | 6 | 2 | 2 | 12+ | 1.00 | 10^{-4} |
| \( S_4 \) | 2^4 | 22 | 20 | 24 | 19 | 2 | 2 | 11+ | 0.00 | 10^3 |
| \( T_3 \) | 4^2 | 15 | 55 | 7 | 40 | 4 | 2 | 21+ | 1.74 | 10^{-3} |
| \( T_3 \) | 4^2 | 24 | 44 | 4 | 20 | 2 | 2 | 12+ | 1.00 | 10^4 |

Code parameters \( \gamma, n, k, d^* \). \( \gamma = a + b \) means that \( G = aP + bP' \) for some rational places \( P \) and \( P' \). Decoder parameters \( \ell, s, \tau \). Number of errors \( \tau \) means that \( \tau = \tau_{\text{max}} \). Observed failure rate OFR. Each simulation was repeated at least \( N \) times.
REFERENCES

[1] G. Schmidt, V. Sidorenko, and M. Bossert, “Decoding Reed-Solomon Codes Beyond Half the Minimum Distance Using Shift-Register Synthesis,” in IEEE International Symposium on Information Theory, 2006, pp. 459–463.

[2] ——, “Syndrome Decoding of Reed-Solomon Codes Beyond Half the Minimum Distance Based on Shift-Register Synthesis,” IEEE Transactions on Information Theory, vol. 56, no. 10, pp. 5245–5252, 2010.

[3] M. Sudan, “Decoding of Reed–Solomon Codes beyond the Error-Correction Bound,” Journal of Complexity, vol. 13, no. 1, pp. 180–193, 1997.

[4] J. S. R. Nielsen, “Power Decoding of Reed–Solomon Codes Revisited,” in International Castle Meeting on Coding Theory and Applications, Sep. 2014. [Online]. Available: http://jsrn.dk/publications.html

[5] J. Rosenkilde, “Power Decoding of Reed–Solomon Up to the Johnson Radius,” Advances in Mathematics of Communications, vol. 12, no. 1, pp. 81–106, Feb. 2018.

[6] V. Guruswami and M. Sudan, “Improved Decoding of Reed–Solomon and Algebraic-Geometric Codes,” in IEEE Annual Symposium on Foundations of Computer Science, 1998, pp. 28–37.

[7] V. D. Goppa, “Codes on algebraic curves,” Dokl. Akad. Nauk SSSR, vol. 259, no. 6, pp. 1289–1290, 1981.

[8] ——, “Algebraico-Geometric Codes,” Izvestiya: Mathematics, vol. 21, no. 1, pp. 75–91, 1983.

[9] M. A. Tsfasman, S. G. Vladut, and Z. Zink, “Modular curves, Shimura curves, and Goppa codes, better than Varshamov-Gilbert bound,” Mathematische Nachrichten, vol. 109, no. 1, pp. 21–28, 1982.

[10] V. Sidorenko and G. Schmidt, “A Linear Algebraic Approach to Multisequence Shift-Register Synthesis,” Problems of Information Transmission, vol. 47, no. 2, pp. 149–165, 2011.

[11] J. Rosenkilde and A. Storjohann, “Algorithms for simultaneous Hermite–Padé approximations,” Journal of Symbolic Computation, vol. In press, Oct. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0747717119301300

[12] M. Chowdhury, C.-P. Jeannerod, V. Neiger, E. Schost, and G. Villard, “Faster Algorithms for Multivariate Interpolation With Multiplicities and Simultaneous Polynomial Approximations,” IEEE Transactions on Information Theory, vol. 61, no. 5, pp. 2370–2387, May 2015.

[13] V. Neiger, J. Rosenkilde, and E. Schost, “Fast Computation of the Roots of Polynomials Over the Ring of Power Series,” in International Symposium on Symbolic and Algebraic Computation, Jul. 2017. [Online]. Available: https://hal.inria.fr/hal-01457954/document

[14] J. Nielsen and P. Boelen, “Sub-Quadratic Decoding of One-Point Hermitian Codes,” IEEE Transactions on Information Theory, vol. 61, no. 6, pp. 3225–3240, Jun. 2015.

[15] A. Ahmed, R. Koetter, and N. R. Shanbhag, “VLSI Architectures for Soft-Decision Decoding of Reed-Solomon Codes,” IEEE Transactions on Information Theory, vol. 57, no. 2, pp. 648–667, Feb. 2011.

[16] X.-W. Wu and P. H. Siegel, “Efficient Root-Finding Algorithm With Application to List Decoding of Algebraic-Geometric Codes,” IEEE Transactions on Information Theory, vol. 47, no. 6, pp. 2579–2587, 2001.

[17] H. Cohn and N. Heninger, “Approximate Common Divisors via Lattices,” The Open Book Series, vol. 1, no. 1, pp. 271–293, 2013.

[18] S. Puchinger and J. Rosenkilde né Nielsen, “Decoding of Interleaved Reed-Solomon Codes Using Improved Power Decoding,” IEEE International Symposium on Information Theory, 2017.

[19] A. Couvreur and J. Panaccione, “Power error locating pairs,” Designs, Codes and Cryptography, vol. 88, no. 8, pp. 1561–1593, 2020.

[20] J. Rosenkilde, “Power decoding Reed-Solomon codes up to the Johnson radius,” Advances in Mathematics of Communications, vol. 12, no. 1, p. 81, 2018.

[21] S. Puchinger, J. Rosenkilde, and I. Bouw, “Improved Power Decoding of Interleaved One-Point Hermitian Codes,” Designs, Codes and Cryptography, vol. 87, no. 2, pp. 598–607, 2019.

[22] S. C. Porter, B.-Z. Shen, and R. Pelikkaan, “Decoding geometric goppa codes using an extra place,” IEEE transactions on information theory, vol. 38, no. 6, pp. 1663–1676, 1992.

[23] S. Gao, “A New Algorithm for Decoding Reed-Solomon Codes,” in Communications, Information and Network Security, ser. The Springer International Series in Engineering and Computer Science. Springer, Jan. 2003, no. 712, pp. 55–68.

[24] H. Stichtenoth, Algebraic Function Fields and Codes, 2nd ed. Springer, 2009.

[25] G. Schmidt, V. R. Sidorenko, and M. Bossert, “Collaborative Decoding of Interleaved Reed–Solomon Codes and Concatenated Code Designs,” IEEE Transactions on Information Theory, vol. 55, no. 7, pp. 2991–3012, 2009.

[26] A. Garcia and H. Stichtenoth, “On the asymptotic behaviour of some towers of function fields over finite fields,” Journal of number theory, vol. 61, no. 2, pp. 248–273, 1996.