Decoding of Hermitian Codes Beyond the Minimum Distance Bound [*][†]

Michael E. O'Sullivan [‡][§]

January 15, 2001

Abstract

Index Terms: Algebraic geometry codes, decoding, minimum distance.
Experimental results are given that show certain Hermitian codes can be decoded well beyond the minimum distance bound. The explanation for this behavior is that most sets of τ points on the Hermitian curve have a geometrical property that allows majority voting for unknown syndromes to proceed for an unexpectedly low check code divisor. Performance curves based on the improved decoding bound show that Hermitian codes compare favorably with Reed-Solomon codes.

[*]A preliminary report was presented at ISIT, Ulm, Germany, 1997

[†]This research was supported by NSF Small Business Innovations Research Grant #9403420.

[‡]Department of Mathematical and Computer Sciences, San Diego State University, San Diego, CA, 92182-7720, mosulliv@math.sdsu.edu

[§]Member, IEEE

1 Introduction

In one of the early reports on decoding of algebraic geometry codes [5], the authors discuss their implementation of the original decoding algorithm of Høholdt, Justesen et al [2]. They say that the algorithm applied to the code $[4096, 3705, 272]$ could actually correct error vectors of weight larger than 135 provided the error positions lie on a curve of degree at most 15. In addtion, although running times to correct an error were on the order of 15 minutes "malicious error locations have resulted in up to 45 minutes." In [4], the authors say that errors with weight larger than half the minimum distance were corrected and that "an error pattern should be rather cleverly selected to make the algorithm break down." These comments raised my curiosity to explore decoding of algebraic codes beyond the minimum distance, and the differences in decoding results between error vectors of the same weight. In this article I show that the decoding algorithm presented in [6] along with modifications for computing error evaluator polynomials [9, 8] can sometimes decode well beyond the minimum distance bound $(d-1)/2$. It should be emphasized that the algorithm is a generalization of the Berlekamp-Massey algorithm, is computationally efficient, employs no soft decisions, and does not use "most likely" guesses and multiple passes through the algorithm.

The key to understanding this behavior is that a randomly chosen set of τ rational points on the Hermitian curve will have, with overwhelming probability, a certain geometrical property. For example, three points will generally not lie on a line. It is even less likely that they will lie on a vertical line. We can say that three non-collinear points are "generic." Similar statements can be made for larger numbers of points; roughly speaking, a generic set of τ points in the plane is one which does not lie on a curve of a given degree d unless every set of τ points lies on some curve of degree less than or equal to d.

This geometrical property appears in the decoding algorithm. Let Q be the point at infinity on a Hermitian curve of genus g. Let e be an error vector of weight τ and let D_e be the set of points where e has a nonzero coordinate. The algorithm applied to e produces two sets of integers, Σ and Δ, which partition the set of nongaps at Q. The elements of Σ are exactly those nongaps such that there is a plane curve of that pole order at Q containing D_e. When D_e is generic, Δ consists of the first τ nongaps. It turns out that for $\tau < g-1$ these errors require fewer check symbols to correct than an error vector which is in more special position, lying on a curve of lower pole order at Q.

A related observation appears in [3] where the authors consider lower rate codes than those herein and find that generic ("independent") vectors of weight τ are not corrected if $\tau \geq d_{FR}/2$. On the other hand, if $\tau = \lfloor (d_{FR}-1)/2 \rfloor + 1$, then almost all of the second most common type of vector are corrected.

There has been recent work by Sudan and others [13, 11] which also concerns decoding beyond the minimum distance bound. These decoding algorithms produce a list of all the codewords within a certain distance of the received vector. There is a bound on the size of the list, but most of the time the list is much smaller than the bound. Indeed, it often contains a single code vector, which is then chosen as the decoded vector. The results are most dramatic for low to moderate rate codes, and the performance is due to the small number of vectors within *any* sphere of specified radius (say between $d/2$ and d) of a given vector. The explanation for the behaviour of the algorithm used here is slightly different. It applies to quite high rate codes. For *most* vectors e of weight τ, the sphere of radius τ centered at e contains only the 0 code vector. It would be nice to have a better understanding of the relationship between the two phenomena.

In Section 2, I tabulate my results for decoding Hermitian codes over the fields $\mathbb{F}_{16}$, $\mathbb{F}_{64}$ and $\mathbb{F}_{256}$ and comment upon the parameters involved. In Section 3, I compare the performance of Hermitian codes with with Reed-Solomon codes and show that Hermitian codes fare quite well. In Section 4 and Section 5, I briefly discuss the decoding algorithm and the dimension of the check code necessary to correct a given error vector e, as well as the number of iterations of the algorithm necessary to complete the computation of locator and evaluator polynomials. I also give formulas for the values of the dimension of the check code and for the number of iterations necessary for vectors e such that D_e is a generic set. Section 6 gives further experimental results concerning the proportions of the different sets Δ for a fixed number of errors, τ.

2 Results for Hermitian Codes

We consider the Hermitian curves over the fields $\mathbb{F}_{16}$, $\mathbb{F}_{64}$ and $\mathbb{F}_{256}$, having lengths 64, 512 and 4096, respectively. A description of the code construction and basic properties of Hermitian codes can be found in [12, 14], and the exact minimum distance of the codes is computed in [15]. The decoding algorithm—called KEY EQUATION—is described in [6, 9] for arbitrary one-point codes, and in [8] for Hermitian codes.

The decoding algorithm solves a key equation, just as the Berlekamp-Massey algorithm does. It is quite similar to Sakata's algorithm, with one main difference. The algorithm produces pairs of error locator polynomials and corresponding error evaluator polynomials. The solution of the key equation also involves voting for unknown syndromes, a technique introduced by Feng and Rao [1]. In fact, the behavior of decoding well beyond the minimum distance is simply a result of letting the algorithm continue to compute new syndromes whenever the majority voting algorithm succeeds. If the algorithm does not produce information that is somehow self-contradictory or inconclusive, the locators and evaluators continue to be updated until the final iteration. Then error evaluation proceeds.

The results of the decoding experiments are tabulated in the Tables I, II, III and IV. For a given number of errors, the code parameters $[n, k, d]$ (d the true minimum distance), the check code dimension $n - k$ and the number of iterations of KEY EQUATION are given. Then data for the given number of trials is tabulated: the number corrected, the number miscorrected, and the number where the decoder detected that it could not

number	check	code parameters	last	number of words in 1000		
errors	dimension	[n,k,d]	iteration	corrected	detected	miscorrected
2	7	[64, 57, 4]	19	952	47	0
3	10	[64, 54, 5]	21	953.9	45.9	.2
4	12	[64, 52, 8]	23	957	43	0

Table I: Results for decoding Hermitian codes over $\mathbb{F}_{16}$.

correct the error and aborted.

The results were obtained by generating τ positions and then τ nonzero values for those positions, running the algorithm, and checking that output error values agreed with input error values. Thus the error vectors had exactly the prescribed weight, and there was no effort made to simulate a channel. For the codes over $\mathbb{F}_{16}$, I divided the set of all possible error vectors into equivalence classes. For each equivalence class I generated a single vector, tested it and assigned a weight to the outcome based on the size of the equivalence class. The only miscorrections occur when $\tau = 3$ and the error positions are collinear, on a non-vertical line. Then it is possible for e to have the same syndrome as a vector e' whose error positions are the other two points on the line.

For the codes over $\mathbb{F}_{64}$, I ran 100,000 randomly generated trials of given weight τ to ensure a reasonable level of precision. For some codes the algorithm corrected every error vector tested, although there certainly exist uncorrectable vectors of weight τ. For the codes over $\mathbb{F}_{256}$, the computing time required and the number of codes to test was so large that I restricted the test to 1000 trials. Notice that for most codes all 1000 error vectors were corrected.

There are a few comments to make about these results. First, the check code dimension needed to correct τ errors is roughly 3τ for codes over $\mathbb{F}_{64}$ and 4τ for codes over $\mathbb{F}_{256}$. Next, the ratio of τ/d (where d is the minimum distance) can be much larger than 1/2. For 6-bit codes τ/d is roughly 1 for τ between 5 and 10. These are codes with rates between 94% and 97%. When τ is 3, 4, or 11, τ/d is roughly 0.6, and for other codes it is roughly 1/2, correcting perhaps 1 or 2 errors more than $(d-1)/2$. For 8-bit codes, τ/d is roughly 2 for τ between 21 and 36. These are codes of exceptionally high rate, 96.5% to 98%. For τ between 8 and 19, τ/d is larger than 1, and for τ between 3 and 7, or 38-48, τ/d is larger than 0.6. For more than 48 errors the ratio approaches 1/2, but for certain codes it is possible to correct as many as 15 errors more than $(d-1)/2$.

Finally, *triangular numbers*, $\binom{m}{2}$, are special. The codes which produced the most miscorrections or detections are those where τ is triangular, or 1 less than a triangular number. There are also moderate jumps in the check code dimension required as τ goes from $h-1$ to h and then to $h+1$ where h is triangular. There are other places where it jumps as well, (at $\tau = 12, 19$ over $\mathbb{F}_{64}$ and $\tau = 38, 49, 61, 74$ over $\mathbb{F}_{256}$). Similarly the last iteration required normally jumps by 1 as τ increases by 1, but it jumps by 2 from $h-1$ to h and by a larger amount from h to $h+1$ where, again, h is triangular.

number errors	check dimension	code parameters [n,k,d]	last iteration	number of words in 100,000 corrected	detected	miscorrected
2	7	[512, 505, 4]	71	98571	1429	0
3	10	[512, 502,5]	73	98614	1383	3
4	16	[512, 496, 6]	79	99914	86	0
5	17	[512, 495, 6]	80	98211	1661	128
6	21	[512, 491, 7]	82	98399	1601	0
7	29	[512, 483, 8]	87	99963	37	0
8	30	[512, 482,8]	88	99970	27	3
9	31	[512, 481, 8]	89	98163	1637	200
10	36	[512, 476, 9]	91	98411	1589	0
11	38	[512, 474, 16]	95	99974	26	0
12	46	[512, 466, 24]	96	99998	0	2
13	47	[512, 465, 24]	97	99962	0	38
14	48	[512, 464, 24]	98	98179	1	1820
15	52	[512, 460, 25]	100	99981	19	0
16	54	[512, 458, 27]	103	99974	26	0
17	56	[512, 456, 32]	104	100000	0	0
18	58	[512, 454, 32]	105	99938	62	0
19	64	[512, 448, 40]	106	99959	0	41
20	65	[512, 447, 40]	107	98019	0	1981
21	68	[512, 444, 41]	109	73223	59	26718
22	70	[512, 442, 43]	111	99999	1	0
23	72	[512, 440, 45]	112	100000	0	0
24	74	[512, 438, 48]	113	99999	1	0
25	76	[512, 436, 49]	114	99976	24	0
26	78	[512, 434, 51]	115	98405	1594	1
27	82	[512, 430, 56]	116	97751	1579	670

Table II: Results for decoding Hermitian codes over $\mathbb{F}_{64}$.

number errors	check dimension	code parameters [n,k,d]	last iteration	number of words in 1000 corrected	detected	miscorrected
2	7	[4096,4089, 4]	271	1000	0	0
3	10	[4096,4086, 5]	273	1000	0	0
4	16	[4096,4080, 6]	287	1000	0	0
5	17	[4096,4079, 6]	288	996	4	0
6	21	[4096,4075, 7]	290	998	2	0
7	29	[4096,4067, 8]	303	1000	0	0
8	30	[4096,4066, 8]	304	1000	0	0
9	31	[4096,4065, 8]	305	994	6	0
10	36	[4096,4060, 9]	307	999	1	0
11	46	[4096,4050, 10]	319	1000	0	0
12	47	[4096,4049, 10]	320	1000	0	0
13	48	[4096,4048, 10]	321	1000	0	0
14	49	[4096,4047, 10]	322	995	5	0
15	55	[4096,4041, 11]	324	995	5	0
16	67	[4096,4029, 12]	335	1000	0	0
17	68	[4096,4028, 12]	336	1000	0	0
18	69	[4096,4027, 12]	337	1000	0	0
19	70	[4096,4026, 12]	338	1000	0	0
20	71	[4096,4025, 12]	339	991	9	0
21	78	[4096,4018, 13]	341	994	6	0
22	92	[4096,4004, 14]	351	1000	0	0
23	93	[4096,4003, 14]	352	1000	0	0
24	94	[4096,4002, 14]	353	1000	0	0
25	95	[4096,4001, 14]	354	1000	0	0
26	96	[4096,4000, 14]	355	1000	0	0
27	97	[4096,3999, 14]	356	1000	0	0
28	105	[4096,3991, 15]	358	997	3	0
29	121	[4096,3975, 16]	367	1000	0	0
30	122	[4096,3974, 16]	368	1000	0	0
31	123	[4096,3973, 16]	369	1000	0	0
32	124	[4096,3972, 16]	370	1000	0	0
33	125	[4096,3971, 16]	371	1000	0	0
34	126	[4096,3970, 16]	372	1000	0	0
35	127	[4096,3969, 16]	373	997	3	0
36	136	[4096,3960, 17]	375	996	4	0
37	138	[4096,3958, 32]	383	1000	0	0

Table III: Results for decoding Hermitian codes over $\mathbb{F}_{256}$.

number errors	check dimension	code parameters [n,k,d]	last iteration	number of words in 1000 corrected	detected	miscorrected
38	154	[4096,3942, 48]	384	1000	0	0
39	155	[4096,3941, 48]	385	1000	0	0
40	156	[4096,3940, 48]	386	1000	0	0
41	157	[4096,3939, 48]	387	1000	0	0
42	158	[4096,3938, 48]	388	1000	0	0
43	159	[4096,3937, 48]	389	1000	0	0
44	160	[4096,3936, 48]	390	995	0	5
45	168	[4096,3928, 49]	392	1000	0	0
46	170	[4096,3926, 51]	281	1000	0	0
47	172	[4096,3924, 64]	281	1000	0	0
48	174	[4096,3922, 64]	282	1000	0	0
49	188	[4096,3908, 80]	283	1000	0	0
50	189	[4096,3907, 80]	284	1000	0	0
51	190	[4096,3906, 80]	285	1000	0	0
52	191	[4096,3905, 80]	286	1000	0	0
53	192	[4096,3904, 80]	287	1000	0	0
54	193	[4096,3903, 80]	288	998	0	2
55	200	[4096,3896, 81]	289	1000	0	0
56	202	[4096,3894, 83]	297	1000	0	0
57	204	[4096,3892, 85]	297	1000	0	0
58	206	[4096,3890, 96]	298	1000	0	0
59	208	[4096,3888, 96]	299	1000	0	0
60	210	[4096,3886, 96]	300	1000	0	0
61	222	[4096,3874,112]	301	1000	0	0
62	223	[4096,3873,112]	302	1000	0	0
63	224	[4096,3872,112]	303	1000	0	0
64	225	[4096,3871,112]	304	999	0	1
65	226	[4096,3870,112]	305	999	0	1
66	232	[4096,3864,113]	306	1000	0	0
67	234	[4096,3862,115]	313	1000	0	0
68	236	[4096,3860,117]	313	1000	0	0
69	238	[4096,3858,119]	314	1000	0	0
70	240	[4096,3856,128]	315	1000	0	0
71	242	[4096,3854,128]	316	1000	0	0
72	244	[4096,3852,128]	317	1000	0	0
73	246	[4096,3850,128]	318	1000	0	0
74	256	[4096,3840,144]	319	1000	0	0
75	257	[4096,3839,144]	320	1000	0	0

Table IV: Results for decoding Hermitian codes over $\mathbb{F}_{256}$, continued.

3 Code Performance

We now consider the performance of Hermitian codes due to decoding beyond the minimum distance bound. Performance is measured using input bit error rate versus output word error rate. The word error rate for a Hermitian code correcting τ errors most of the time using the algorithm KEY EQUATION is computed as

$$1-\sum_{k=0}^{\tau-1}\binom{n}{k}p_s^k(1-p_s)^{n-k}-\alpha\binom{n}{\tau}p_s^\tau(1-p+s)^{n-\tau}$$

where p_s is the symbol error probability and α is the proportion of codewords of weight τ which are corrected. This neglects the fact that some errors of weight $\tau-1,\ldots,(d+1)/2$ are not corrected, but these are so rare that they are insignificant. In fact the proportion α is so close to 1 that the performance of these codes appears to be very close to a τ error correcting code.

The figures compare Hermitian codes with Reed-Solomon codes of roughly the same rate. In all but one of the comparisons I have used a Hermitian code of maximum length and compared it to two Reed-Solomon codes. The first is a Reed-Solomon code over a field $\mathbb{F}_{2^r}$ where r is chosen as small as possible and the length is chosen so that the code has the same length in bits as the Hermitian code. The second RS code is over a field $\mathbb{F}_q$ chosen so that $q\log_2(q)$ is equal to the length in bits of the Hermitian code. The probability of error for a field element p_s is computed from the bit error probability p_b by

$$p_s=1-(1-p_b)^{\log_2(q)}$$

Although this code is not of practical utility since the arithmetic is not in characteristic 2, it gives a fairer comparison of the Hermitian code with an RS code since neither one must be shortened. Figure 2 compares a shortened Hermitian code with a full length Reed-Solomon code.

The figures show that Hermitian codes seem roughly comparable to Reed-Solomon codes under the constraint that the length of the code in bits is held constant.

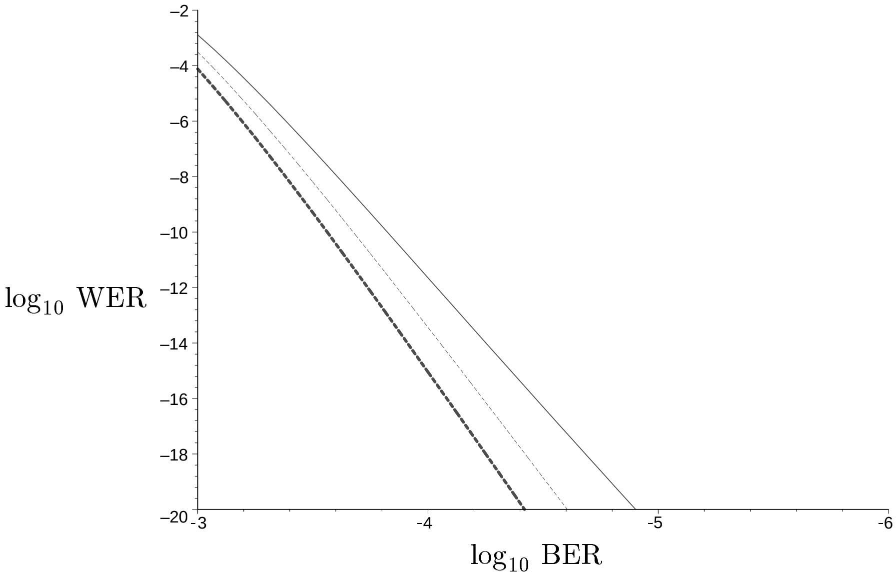


Figure 1: Output word error rate as a function of input bit error rate for codes of rate 94%: Hermitian [512,481,8] correcting 9 errors (solid), shortened RS [341,320,22] over $\mathbb{F}_{512}$ (dashed), RS [361,338,24] over $\mathbb{F}_{361}$ (dark dashed).

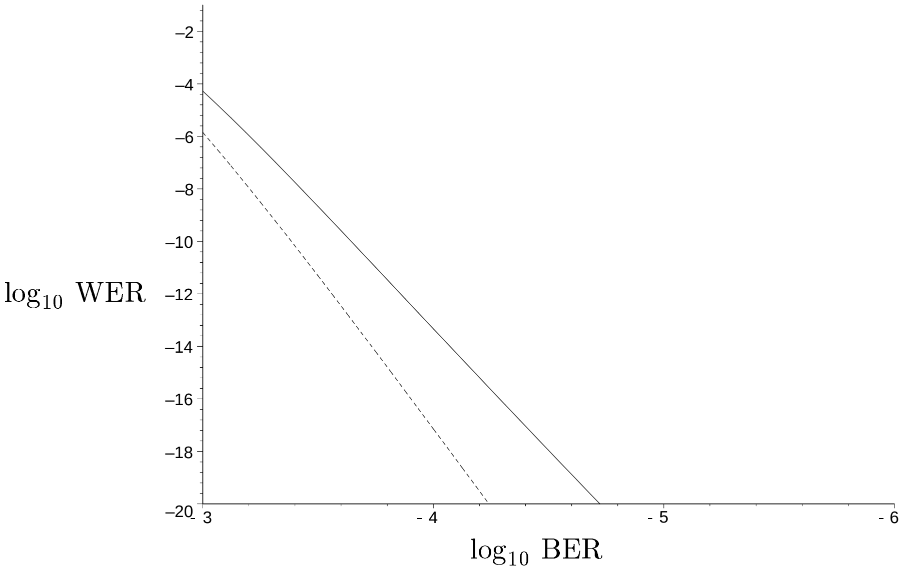


Figure 2: Output word error rate as a function of input bit error rate for codes of rate 94%: Shortened Hermitian [340,309,8] correcting 9 errors (solid) over $\mathbb{F}_{64}$, RS [256,233,24] over $\mathbb{F}_{256}$ (dashed).

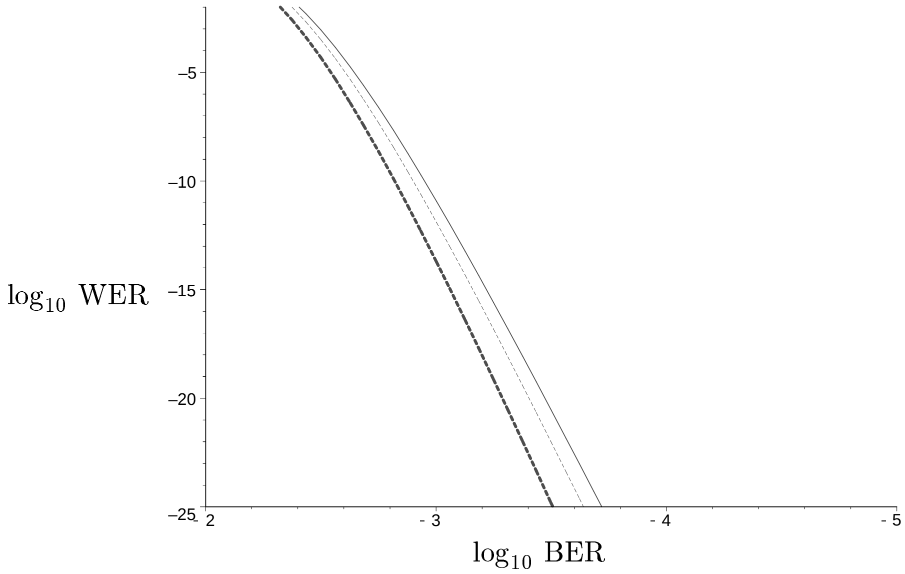


Figure 3: Output word error rate as a function of input bit error rate for codes of rate 87.5%: Hermitian [512,447,40] correcting 20 errors (solid), shortened RS [341,298,44] over $\mathbb{F}_{512}$ (dashed), RS [361,315,47] over $\mathbb{F}_{361}$ (dark dashed).

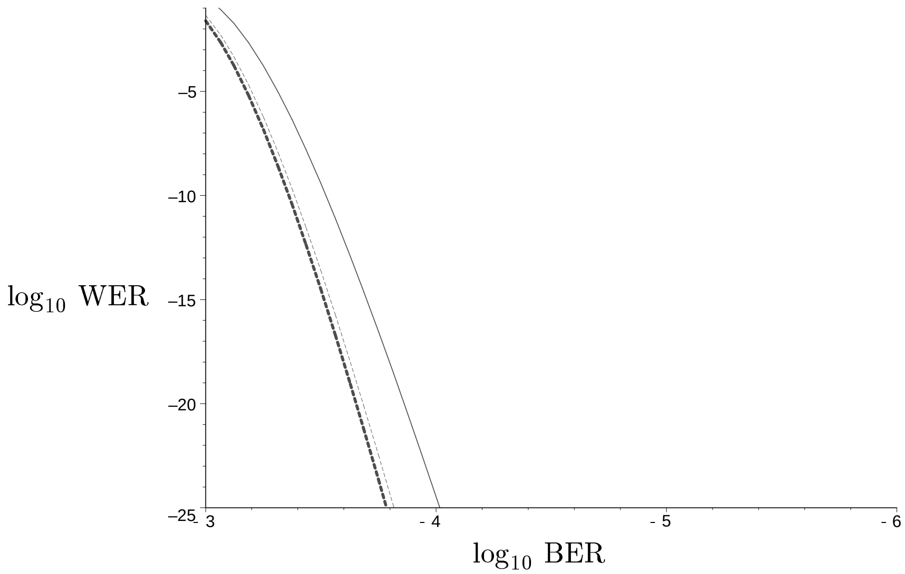


Figure 4: Output word error rate as a function of input bit error rate for codes of rate 97%. Hermitian [4096,3969,16] correcting 35 errors (solid), shortened RS [2731,2646,86] over $\mathbb{F}_{4096}$ (dashed), RS [2809,2722,88] over $\mathbb{F}_{361}$ (dark dashed).

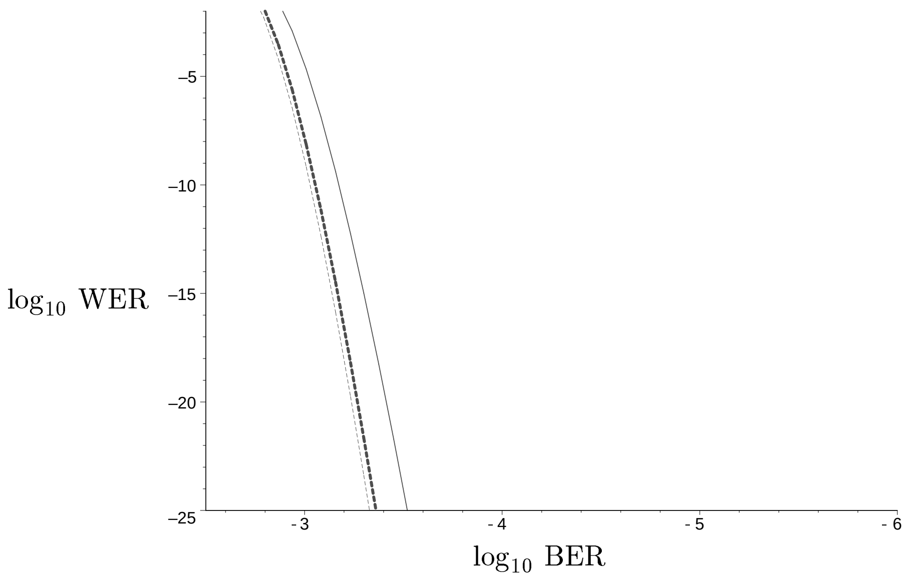


Figure 5: Output word error rate as a function of input bit error rate for codes of rate 95%. Hermitian [4096,3892,85] correcting 57 errors (solid), shortened RS [2731,2595,137] over $\mathbb{F}_{4096}$ (dashed), RS [2809,2669,141] over $\mathbb{F}_{361}$ (dark dashed).

4 Derivation of Parameters: The Last Iteration

In this section and the next we consider the question of what code parameters are necessary to correct vectors of a particular weight τ. Let $m^\star Q$ be the divisor defining the check code. Let M be the last iteration of KEY EQUATION. Then the question is to determine the value of $m^\star$ such that majority voting is ensured to proceed correctly, and the value of M such that the compution of error locators as well as the computation of error evaluators is complete. We consider M in this section and $m^\star$ in the next. We derive results for an arbitrary given vector e and then specialize to a generic vector. The analysis here is valid for any one-point code, only the computation of specific formulas for $m^\star$ and M are specific to the Hermitian curve.

It is assumed that the reader is familar with certain notation and results from [6] and [8] which we now summarize. Let Λ be the set of nongaps at Q. For f a function on the curve, let $o(f) \in \Lambda$ be its pole order at Q. Let $L(mQ)$ be the space of functions with pole order at most m at Q. For a given vector e, let $\Sigma \subset \Lambda$ be the set of pole orders of all the locators for e, and let Δ be the complement of Σ in Λ. We will call these the sigma and delta sets of e. For a function f which is not a locator we let $\operatorname{span} f$ be the nongap c such that the syndrome of fg is zero when $o(g) < c$ but is nonzero when $o(g) = c$. We define $\operatorname{fail} f = \operatorname{span} f + o(f)$. A key result is that the set of spans of non-locators is exactly the set Δ.

Definition 4.1. The vector e of weight τ is *generic* if the set Δ consists of the first τ nongaps.

At the end of the mth iteration of KEY EQUATION, a set of nongaps σ_m has been computed and for each $s \in \sigma_m$ a function f has been computed which has order s and fail larger than m. If f is not a locator then there must be some $m' > m$ such that $\operatorname{span} f = m' - s \in \Delta$. Looked at another way, suppose that for any $m' > m$, $m' - s \notin \Delta$. Then f must be a locator. Thus we have established the following.

Proposition 4.2. *Let M be an integer such that for any nongap $m > M$, and for any $s \in \sigma$, $m - s \notin \Delta$. Then $\sigma_M = \sigma$ and the candidate locators at the end of the Mth iteration are in fact locators.*

The algorithm also produces one evaluator polynomial for each locator polynomial. The coefficients of the evaluator associated to a locator f are computed as recursions of the syndrome vector determined by shifts of f that occur at an iteration m where $m - o(f)$ is a *gap*. Assuming that f is a locator, the computation of the corresponding evaluator is complete at the mth iteration provided $m - o(f) \geq 2g - 1$, since $2g - 1$ is the largest gap. Thus we have the following result.

Proposition 4.3. *Let $s_{\max}$ be the largest element of σ and let $c_{\max}$ be the largest element of Δ. Let*

$$M = s_{\max} + \max\{c_{\max}, 2g - 1\}$$

Then the computation of error locators and error evaluators is complete at the end of the Mth iteration.

PROOF: For $m > M$, and $s \in \sigma$, $m - s > M - s_{\max} \geq c_{\max}$, so that $m - s \notin \Delta$. Thus the error locators are completed after the Mth iteration. In addition, $m - s \geq 2g - 1$ so the computation of error evaluators is complete after the Mth iteration. $\square$

Now suppose that e is generic, so that the set Δ consists of the first τ nongaps. Let λ_τ be the τth nongap, and therefore the largest element of Δ. Write $\lambda_\tau = ar + b(r+1)$ with a, b nonnegative integers and $a \leq r$. If $a = 0$, then the elements of σ are $\{ir + j(r+1) : 0 \leq i \leq r \text{ and } i + j = b + 1\}$. The largest element of σ is $(b+1)(r+1) = \lambda_\tau + r + 1$. If $a \neq 0$ then the elements of σ are $\{ir + j(r+1) : 0 \leq i < a \text{ and } i + j = b\} \cup \{ir + j(r+1) : a + 1 \leq i \leq r \text{ and } i + j = b + 1\}$. The largest element of σ is $(a+1)r + b(r+1) = \lambda_\tau + r$.

In the next section Proposition 5.4 shows that decoding beyond half the minimum distance only occurs for $\tau < g - 1$, so we may assume that $\lambda_\tau < 2g - 1$. From the proposition we have $M = s_{\max} + 2g - 1$. Using $2g - 1 = r^2 - r - 1$ we have the following result.

Proposition 4.4. *Let e be a generic error of weight $\tau < g$ and let λ_τ be the τth nongap. Write $\tau = ar + b(r+1)$ where $0 \leq a \leq r$. Then the computation of the locators and evaluators is complete at the end of the Mth iteration where*

$$M = \begin{cases} \lambda_\tau + r^2 - 1 & \text{if } a \neq 0 \\ \lambda_\tau + r^2 & \text{if } a = 0 \end{cases}$$

We can now explain the jumps in the value of M in Tables I II, and III. Note that $\lambda_{\tau+1} = \lambda_\tau + 1$ when τ is not triangular. When τ is triangular, $\lambda_\tau = b(r+1)$ and $\lambda_{\tau+1} = (b+1)r$, so $\lambda_{\tau+1} - \lambda_\tau = r - b$. Consequently, as the number of errors to correct changes from τ to $\tau + 1$, the value of M will generally increase by 1 with two types of exceptions: if $\tau + 1$ is triangular, then the two values of M differ by 2; if τ is triangular, then they differ by $r - b - 1$.

5 Derivation of Parameters: The Check Code

In this section we derive a general property that will define what check code $C_{m^\star}$ is needed to correct a given error vector e. The key question is to determine when majority voting for a new syndrome is guaranteed to be correct.

We will use the notation $m - \Sigma$ for $\{m - s : s \in \Sigma\}$. Let $\Lambda_{\preceq m} = \Lambda \cap (m - \Lambda)$. This is the set of nongaps s such that $m - s$ is a nongap. Recall that at the end of the mth iteration of KEY EQUATION there are two sets Σ_m and Δ_m that partition the set of nongaps Λ. For each $s \in \Sigma_m$ there exists a function f with order s such that fail $f > m$. Let $\Gamma_m = \Sigma_{m-1} \cap (m - \Sigma_{m-1})$. At the beginning of the mth iteration, when majority voting for the new syndrome is done, a vote is cast by each element of Γ_m. Let V_m be the correct voters and let W_m be the incorrect voters. The incorrect voters are spans of functions which fail at the mth iteration, so these incorrect voters become members of the new delta set, $\Delta_m = W_m \cup \Delta_{m-1}$. The following proposition gives a sufficient value for $m^\star$ such that for $m > m^\star$, $\#V_m > \#W_m$, so correct voting is assured.

Proposition 5.1. *Let e be a vector with locator pole orders Σ and let $\Delta = \Lambda \setminus \Sigma$. Let $m^\star$ be the smallest nongap such that for all $m > m^\star$,*

$$\#(\Lambda_{\preceq m} \cap \Sigma) > \#(\Lambda_{\preceq m} \cap \Delta) \tag{1}$$

Then for all $m > m^\star$, majority voting will produce the correct synrome value. Thus the vector e can be corrected by KEY EQUATION applied to the code $C_{m^\star}$.

PROOF: For $m > m^\star$ we consider the following partition of $\Lambda_{\preceq m}$.

$$\begin{aligned} T_m &= \Sigma \cap (m - \Sigma) \\ U_m &= \Sigma \cap (m - \Delta) \\ X_m &= \Delta \cap (m - \Sigma) \\ Y_m &= \Delta \cap (m - \Delta) \end{aligned}$$

We have,

$$\begin{aligned} \Lambda_{\preceq m} \cap \Sigma &= T_m \cup U_m \\ \Lambda_{\preceq m} \cap \Delta &= X_m \cup Y_m \end{aligned}$$

Since $\#U_m = \#X_m$, the assumption (1) is equivalent to

$$\#T_m > \#Y_m \tag{2}$$

We have $V_m \supset T_m$ and $Y_m \supset W_m$, so we can conclude that $\#V_m > \#W_m$. $\square$

Definition 5.2. We will say that decoding of a vector e is *guaranteed* by the code $C_{m^\star}$ if (1), or equivalently, (2) holds. Otherwise decoding is not guaranteed.

Now we consider generic error vectors. Let $\Sigma + \Sigma = \{s + s' : s, s' \in \Sigma\}$. Note that the set $T_m = \Sigma \cap (m - \Sigma) = \{s + s' : s, s' \in \Sigma\}$ is non-empty if and only if $m \in \Sigma + \Sigma$. The corollary below shows that for generic vectors, $m^\star$ is the minimal nongap that is not in $\Sigma + \Sigma$.

Corollary 5.3. *Let Σ be the set of locator pole orders for a generic error of weight τ. Let $m^\star$ be the smallest nongap such that for all nongaps $m > m^\star$, $\Sigma \cap (m - \Sigma) \neq \emptyset$. Then the smallest code for which decoding of generic errors of weight τ is guaranteed is $C_{m^\star}$.*

PROOF: Let λ_τ be the τth nongap. Since all elements of Σ are larger than λ_τ, $T_m = \Sigma \cap (m - \Sigma)$ is empty for $m \leq 2\lambda_\tau$. Thus majority voting cannot be guaranteed for $m \leq 2\lambda_\tau$. On the other hand, for $m > 2\lambda_\tau$, $Y_m = \Delta \cap (m - \Delta)$ is empty. Thus if we assume that T_m is nonempty for $m > m^\star$, then (2) holds, and that is equivalent to the hypothesis of the proposition. □

The following two propositions compute $m^\star$ for generic errors and Hermitian codes. The first shows that significant decoding beyond the minimum distance bound should not be expected for codes with $d > 2g$.

Proposition 5.4. *Let $\tau \geq g - 1$ and let λ_τ be the τth nongap. Let Σ be the locator pole orders for a generic error of weight τ. Then the minimal code for which decoding of generic errors of weight τ is guaranteed is $C_{m^\star}$ where*

$$m^\star = \begin{cases} 2\lambda_\tau + 1 & \text{if } \tau \geq g+1 \\ 2\lambda_\tau + 3 & \text{if } \tau = g \text{ or } g-1 \end{cases}$$

The minimum distance of these codes is $2\tau + 1$. Thus, decoding generic errors of weight $\tau \geq g-1$ errors is only guaranteed when it is guaranteed by the minimum distance bound.

PROOF: Suppose first that $\tau \geq g + 1$. Then from the Riemann-Roch theorem, $\lambda_\tau = \tau + g - 1$, $\Sigma = \{s \in \mathbb{Z} : s \geq \lambda_\tau + 1\}$ and $\Sigma + \Sigma = \{s \in \mathbb{Z} : s \geq 2\lambda_\tau + 2\}$. Thus the last nongap not in $\Sigma + \Sigma$ is $m^\star = 2\lambda_\tau + 1$.

If $\tau = g$ then $\lambda_\tau = 2g - 2$ and $\Sigma = \{s \in \mathbb{Z} : s \geq 2g\}$. Then $\Sigma + \Sigma = \{s : s \geq 4g\}$ so $m^\star = 4g - 1 = 2\lambda_\tau + 3$.

Finally, if $\tau = g - 1$ then $\lambda_\tau = 2g - 3$ and $\Sigma = \{s \in \mathbb{Z} : s = 2g - 2 \text{ or } s \geq 2g\}$. Then $\Sigma + \Sigma = \{s : s = 4g - 4 \text{ or } s \geq 4g - 2\}$ so $m^\star = 4g - 3 = 2\lambda_\tau + 3$.

In each case, the minimum distance of the code is $2\lambda_\tau + 1$, see the table in [15] or computations in [6].

□

Proposition 5.5. *Let $\tau < g - 1$ and let $\lambda_\tau = ar + b(r + 1)$, where $0 \leq a \leq r$. Then the minimal code guaranteeing correction of a generic vector of weight τ is $C_{m^\star}$ where*

$$m^\star = \begin{cases} 2\lambda_\tau + r + 1 & \text{if } a = 0 \text{ and } b < (r-1)/2 \\ 2\lambda_\tau + 2(r-b) - 1 & \text{if } a = 0 \text{ and } b \geq (r-1)/2 \\ 2\lambda_\tau + r - b & \text{if } a \neq 0 \text{ and } 2a + b < r \\ 2\lambda_\tau + 1 & \text{if } a \neq 0 \text{ and } 2a + b \geq r \end{cases} \tag{3}$$

PROOF: Note first that since $\tau < g - 1$, $\lambda_\tau < 2g - 2 = r^2 - r - 2$. This forces $a + b < r - 2$. Suppose first that $a = 0$. Then the set of minimal elements of Σ is $\sigma = \{ir + j(r + 1) : 0 \leq i \leq r, i + j = b + 1\}$. Now $\Sigma + \Sigma$ contains

$$\{ir + j(r + 1) : i + j = 2(b + 1)\} \tag{4}$$

and all larger nongaps. If $b < (r-1)/2$ then $2b+2 < r+1$, and the largest nongap not in $\Sigma+\Sigma$ is $(2b+1)(r+1) = 2\lambda_\tau + r + 1$.

If $b \geq (r-1)/2$ then $2(b+1)r$ may also be written $(2b+2-r-1)r + r(r+1)$. All nongaps larger than this are also in $\Sigma+\Sigma$, so the smallest nongap not in $\Sigma+\Sigma$ is $(2b+2-r)r + (r-1)(r+1) = 2br + 2r - 1 = 2\lambda_\tau + 2(r-b) - 1$.

Now suppose that $a \neq 0$. Then the set of minimal elements of Σ is

$$\begin{aligned}\sigma = &\{ir + j(r+1) : 0 \leq i \leq a-1 \text{ and } i+j = a+b\} \\ &\cup \{ir + j(r+1) : a+1 \leq i \leq r \text{ and } i+j = a+b+1\}\end{aligned}$$

Now $\Sigma+\Sigma$ contains

$$\{ir + j(r+1) : 0 \leq i \leq 2a-2 \text{ and } i+j = 2a+2b\}, \tag{5}$$

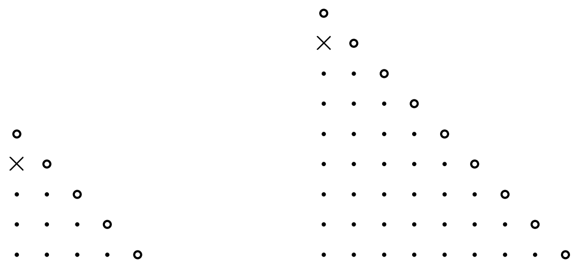
$$\{ir + j(r+1) : 0 \leq i \leq 2a+b \text{ and } i+j = 2a+2b+1\}, \tag{6}$$

$$\{ir + j(r+1) : i+j = 2a+2b+2\} \tag{7}$$

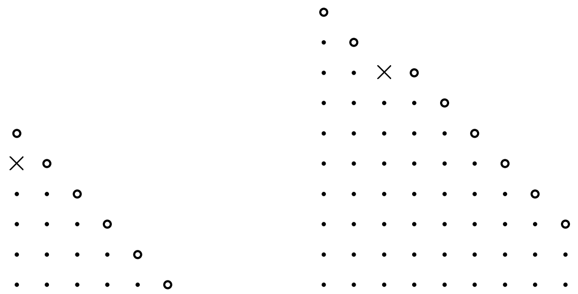
and all larger nongaps. If $2a+b < r$ then all nongaps larger than $(2a+b)r + (b+1)(r+1)$ are in $\Sigma+\Sigma$ but $(2a+b+1)r + b(r+1)$ is not. Thus $m^\star = (2a+b+1)r + b(r+1) = 2\lambda_\tau + r - b$.

If $2a+b \geq r$ then $(2a+b+1)r + b(r+1)$ may also be written $(2a+b-r)r + (b+r)(r+1)$. The same reasoning shows that $ir + j(r+1) \in \Sigma+\Sigma$ for any i, j such that $i+j = 2a+2b+1$. The smallest nongap not in $\Sigma+\Sigma$ is $(2a-1)r + (2b+1)(r+1) = 2\lambda_\tau + 1$. $\square$

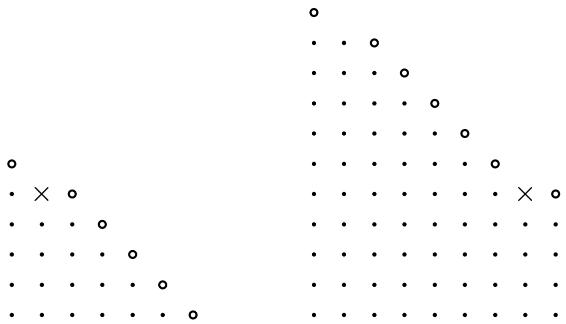
The 4 cases of the Proposition are illustrated in Figure 6 for the Hermitian curve $x^9 = y^8 + y$ over $\mathbb{F}_{64}$. Each nongap λ is represented as an ordered pair (a, b) with $0 \leq a \leq 8$.



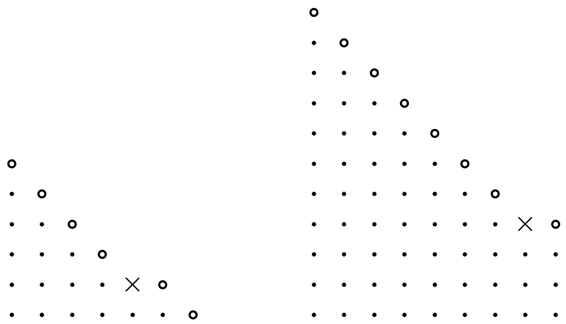
Case $a = 0$ and $b < r(r-1)/2$: $\tau = 10$. $\lambda_\tau = 3(9) = 27$. $m^\star = 7(9) = 63 = 2(27) + 9$.



Case $a = 0$ and $b \geq r(r-1)/2$: $\tau = 15$. $\lambda_\tau = 4(9) = 36$. $m^\star = 2(8) + 7(9) = 79 = 2\lambda_\tau + 2(8-4) - 1$.



Case $a \neq 0$ and $2a + b < r$: $\tau = 20$. $\lambda_\tau = 1(8) + 4(9) = 44$. $m^\star = 7(8) + 4(9) = 92 = 2\lambda_\tau + (8 - 4)$.



Case $a \neq 0$ and $2a + b \geq r$: $\tau = 17$. $\lambda_\tau = 4(8) + 1(9) = 41$. $m^\star = 7(8) + 3(9) = 83 = 2\lambda_\tau + 1$.

Figure 6: The 4 cases in Proposition 5.5. On the left, the elements of σ are shown with circles, $c_{\max}$ with a $\times$ and the other elements of Δ with a dot. On the right, the minimal elements of $\Sigma + \Sigma$ are shown with a circle, $m^\star$ with a $\times$ and the elements of $(\Sigma + \Sigma)^c$ with a dot.

delta set	proportion $\mathbb{F}_{16}$	proportion $\mathbb{F}_{64}$	proportion $\mathbb{F}_{256}$
$\lambda_1, \ldots, \lambda_\tau$	930-952	984	996
$\lambda_1, \ldots, \lambda_{\tau-1}, \lambda_{\tau+1}$	46-62	15	3.8
$\lambda_1, \ldots, \lambda_{\tau-1}, \lambda_{\tau+2}$	2-7	.24	.01-.02
$\lambda_1, \ldots, \lambda_{\tau-2}, \lambda_\tau, \lambda_{\tau+1}$	1.5-3	.24	.01-.02
Others combined	$< .2$	.01 - .03	$< .001$

Table V: Proportions of different delta sets expressed per thousand.

6 Proportions of different delta sets

For a fixed number of errors, τ, a vector of weight τ may have any of a number of different delta sets, Δ. The most likely is the generic delta set which consists of the first τ nongaps. In this section, I give experimental results for the relative proportion of generic delta sets and for the next most common ones.

Suppose first that there are 2 errors. If we fix one location and consider choosing the second, then it is evident that there are $r-1$ choices such that the two points lie on a vertical line. Since there are $r^3 - 1$ possible choices, we have a proportion $(r-1)/(r^3-1) = 1/(r^2+r+1)$ which are not generic errors. The percentage of generic errors is thus 95.2% over $\mathbb{F}_{16}$, 98.6% over $\mathbb{F}_{64}$, and 99.6% over $\mathbb{F}_{256}$. Remarkably, these percentages continue to be roughly valid for a larger number of errors.

One could do the same type of computation for 3, 4 or even more errors. For $\tau = 3$ there are 3 possible delta sets $\{0, r, r+1\}, \{0, r, 2r\}, \{0, r+1, 2r+2\}$ and for $\tau = 4$ there are 5. The complexity of analysing the possibilities and computing the proportions quickly becomes overwhelming. The surprising result is that the relative proportions of the various types of delta sets seem to be roughly the same for different values of τ. These proportions are tabulated in Table V. The figures were obtained by decoding 10^6 randomly generated error vectors of specified weight with a code that could correct all the vectors. Over the field $\mathbb{F}_{16}$ weights between 2 and 7 were tested. The delta sets listed in the table were not always applicable in the case of 2 and 3 errors (for example, $\{0, 5, 8\}$ is not a valid delta set) so I considered those figures in the most reasonable category. For higher number of errors there was no ambiguity. Over $\mathbb{F}_{64}$, about a dozen different weights between 8 and 26 were tested. Over $\mathbb{F}_{256}$ only weights 19, 26 and 34 were tested thoroughly.

An important conclusion from these results is that a code which corrects generic vectors of weight τ will also correct vectors of lower weight which are either generic or which have the second most likely type of delta set. Thus the lower weight vectors have neglible influence on the performance of the code. One need only consider the proportion of vectors of weight τ decoded and the performance on vectors of weight larger than τ. An additional improtant issue for performance which was not discussed in this article, but was evident in experiments, is that miscorrections are exceedingly rare.

References

[1] G. L. Feng and T. R. N. Rao, "Decoding algebraic-geometry codes up to the designed minimum distance," *IEEE Trans. Inform. Theory*, **39**, no. 1, (Jan. 1993) 37-45.

[2] J. Justesen, K. Larsen, H. E. Jensen and T. Høholdt, "Fast decoding of codes from algebraic plane curves," *IEEE Trans. Inform. Theory*, **38** no. 1 (1992), 111-119.

[3] H. E. Jensen, R. R. Nielsen, T. Höholdt, "Recent results in the decoding of algebraic geometry codes," *Proc. Inform. Theory Workshop, Killarney, Ireland*, June, 1998, 35-6.

[4] H. E. Jensen, T. Høholdt, J. Justesen, "On the number of correctable errors for some AG codes," *Trans. Inform. Theory*, vol. 39, no. 2, Mar. 1993, pp. 681-694.

[5] J. Justesen, K. J. Larson, "A program for decoding a code derived from an algebraic curve," Technical Report IT-107, Technical University of Denmark, Sept. 1988.

[6] M. E. O'Sullivan, "Decoding of codes defined by a single point on a curve," *IEEE Trans. Inform. Theory*, **41** no. 6, (1995) 1709-1719.

[7] M. E. O'Sullivan, "Decoding of Hermitian codes beyond $(d_{min} - 1)/2$," in *Proc. IEEE 1997 International Symposium on Information Theory* p. 384.

[8] M. E. O'Sullivan, "Decoding of Hermitian codes: the key equation and efficient error evaluation," submitted to *Trans. Inform. Theory.*

[9] M. E. O'Sullivan, "The key equation for one-point codes and efficient error evaluation," submitted to *Finite Fields Appl.*

[10] S. Sakata, "Extension of the Berlekamp-Massey algorithm to N dimensions," *Informat. Comput.* **84**, Feb. 1990, 207-239.

[11] A. Shokrollahi, H. Wasserman, "List decoding of algebraic-geometric codes," *IEEE Trans. Inform. Theory*, **45** no. 2, (1999) 432-437.

[12] H. Stichtenoth, "A note on Hermitian codes over GF(q^2)," *IEEE Trans. Inform. Theory*, vol. 34, no. 5, pp. 1345-1348, Sept. 1988.

[13] M. Sudan, "On the decoding of Reed-Solomon codes beyond the error-correction bound," *Journal of Complexity*, 13(1), Mar. 1997, 180-193.

[14] H. J. Tiersma, "Remarks on codes from Hermitian curves," *Trans. Inform. Theory*, vol. 33, no. 4, July 1987, pp. 605-609.

[15] C. Yang, P. V. Kumar, "On the true minimum distance of Hermitian codes" in *Coding Theory and Algebraic Geometry, proceedings Luminy, 1991*, Stichtenoth, Tsfasman ed., Springer-Verlag Lecture notes # 1518.