The Guruswami–Sudan Decoding Algorithm for Reed–Solomon Codes

R. J. McEliece[1]

This article is a tutorial discussion of the Guruswami–Sudan (GS) Reed–Solomon decoding algorithm, including self-contained treatments of the Kötter and Roth–Ruckenstein (RR) improvements. It also contains a number of new results, including a rigorous discussion of the average size of the decoder's list, an improvement in the RR algorithm's stopping rule, a simplified treatment of the combinatorics of weighted monomial orders, and a proof of the monotonicity of the GS decoding radius as a function of the interpolation multiplicity.

I. Introduction

In 1997 Madhu Sudan [23], building on previous work of Welch–Berlekamp [24], Ar et al. [1], and others, discovered a polynomial-time algorithm for decoding certain low-rate Reed–Solomon (RS) codes beyond the classical $d/2$ error-correcting bound. Two years later, Guruswami and Sudan [9] published a significantly improved version of Sudan's algorithm, which was capable of decoding virtually every RS code at least somewhat, and often significantly, beyond the $d/2$ limit. The main focus of these seminal articles was on establishing the existence of polynomial-time decoding algorithms, and not on devising practical implementations. However, several later authors, notably Kötter [12,13] and Roth–Ruckenstein (RR) [21], were able to find low-complexity (no worse than $O(n^2)$) realizations for the key steps in the Guruswami–Sudan (GS) algorithm, thus making GS a genuinely practical engineering alternative in storage and transmission systems requiring RS codes.

This article is a tutorial discussion of the GS algorithm, including the Kötter and Roth–Ruckenstein improvements. It also contains a number of new results, including a rigorous discussion of the average size of the decoder's list, an improvement in the RR algorithm's stopping rule, a simplified treatment of the combinatorics of weighted monomial orders, and a proof of the monotonicity of the GS decoding radius as a function of the interpolation multiplicity.

Here is an outline of the article. In Section II, we give an overview of the GS algorithm and several numerical examples. In Sections III and IV, we present a self-contained introduction to the algebraic fundamentals of two-variable polynomials, which are a key component of the GS algorithm. In Section V, we state and prove the two basic theorems that support the GS algorithm: the Interpolation Theorem and the Factorization Theorem. With this preliminary material out of the way, in Section VI we give a formal

[1] California Institute of Technology, Pasadena, California, and Communications Systems and Research Section.

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

description, and proof of correctness, of the Guruswami–Sudan algorithm. In Sections VII through IX, we describe the Kötter and Roth–Ruckenstein improvements.

Finally, in Appendices A through D, we present some miscellaneous material related to the GS algorithm: In Appendix B, for example, we explain how to modify the GS algorithm when erasures are present. In Appendix C, we present a GS-type decoding algorithm that is "conventional" in the sense that it can correct only up to $d/2$ errors, and in Appendix D we give a rigorous treatment of the average number of codewords in the GS output list.

The following notation will be used:

- $\mathbb{N}$: the nonnegative integers, i.e., $\mathbb{N} = \{0, 1, 2, \cdots\}$.
- $|X|$: the number of elements in the finite set X.
- F: a field, usually finite.
- $F[x]$: the ring of polynomials in x, with coefficients from F.
- $\deg f(x)$: the degree of the polynomial $f(x) \in F[x]$.
- $F_v[x]$: the polynomials of degree $\leq v$ from $F[x]$.
- $F[x, y]$: the ring of polynomials in x and y, with coefficients from F. A typical element of $F[x, y]$:

$$Q(x, y) = \sum_{(i,j)\in I} a_{i,j} x^i y^j$$

  where $I = I(Q)$ is a finite set of indices.
- $\mathbb{M}[x, y] = \{x^i y^j : i \geq 0,\ j \geq 0\}$: the set of bivariate ($x$ and y) monomials.
- $\deg_{u,v} Q(x, y)$: the (u, v)-weighted degree of the polynomial $Q(x, y)$, defined as

$$\deg_{u,v} Q(x, y) \triangleq \max_{(i,j)\in I} \{ui + vj\}$$

- The y-degree of $Q(x, y) \in F[x, y]$:

$$\deg_y Q(x, y) \triangleq \deg_{0,1} Q(x, y) = \max_{(i,j)\in I} \{j\}$$

- $F_L[x, y]$: the polynomials from $F[x, y]$ whose y-degree is $\leq L$

II. A First Look at the Guruswami–Sudan Algorithm

In this section, we give an overview of the GS algorithm, including a motivating example, an informal description of the algorithm, and several numerical examples.

Let us recall the definition of an (n, k) Reed–Solomon code over $F = GF(q)$, as given by Reed and Solomon in the original article [20].

Let $(\alpha_1, \cdots, \alpha_n)$ be a fixed list of n distinct elements of F, called the support set of the code. The encoding process is that of mapping a vector $(f_0, f_1, \cdots, f_{k-1})$ of k information symbols into an n-symbol codeword $(x_1, \cdots, x_n)$ by polynomial evaluation, i.e.,

$$(x_1, \cdots, x_n) = \big(f(\alpha_1), \cdots, f(\alpha_n)\big) \tag{1}$$

where

$$f(x) = f_0 + f_1 x + \cdots + f_{k-1} x^{k-1} \tag{2}$$

The corresponding Reed–Solomon code consists of all n-vectors of the form in Eq. (1), where $f(x)$ is a polynomial of degree $<k$.

It is well-known that this code has minimum Hamming distance $d = n-k+1$ and, therefore, is capable of correcting up to

$$t_0 = \left\lfloor \frac{n-k}{2} \right\rfloor \tag{3}$$

errors. Conceptually, this may be accomplished as follows. The decoder searches the Hamming sphere of radius t_0 centered at the received word for codewords. If the sphere contains a unique codeword, that is the decoder's output. Otherwise, the decoder reports failure. (This strategy is called bounded distance decoding (BDD) and dates back to Shannon's proof of the noisy-channel coding theorem [22, Section 13]. The conventional RS decoding algorithms, e.g., Berlekamp [2], Berlekamp–Massey [4,15], continued fractions [18,25], or Euclidean algorithm [16], are all BDD algorithms.) The decoding sphere cannot contain more than one codeword, since the minimum distance of the code is $>2t_0$. If we attempt to correct more than t_0 errors by increasing the decoding radius, it is possible for the decoding sphere to contain more than one codeword, in which case the decoder will fail. For this reason, conventional wisdom asserts that the code is not capable of correcting more than t_0 errors. Nevertheless, if we examine the *probability* that the decoding sphere will contain multiple codewords, rather than the *possibility*, we may reach a different conclusion.

Example 1. Consider the (32,8) RS code over $GF(32)$, with $d = 25$ and $t_0 = 12$. If the decoding radius is taken to be $t = 13$, and the transmitted codeword suffers 13 errors, it is possible for the decoding sphere to contain two codewords: the transmitted codeword (which we will call the causal codeword) and one other, a noncausal codeword at distance 12 or 13 from the received word. However, assuming all error patterns of weight 13 are equally likely, it can be shown (using methods we will describe in Appendix D) that the *probability* of this unfavorable happening is 2.08437×10^{-12}! In short, the code is capable of correcting virtually all patterns of 13 errors, despite having a conventional error-correcting capability of only 12.

Example 1 suggests that it might be possible to design a decoding algorithm for RS codes capable of correcting more than t_0 errors. The Guruswami–Sudan list decoding algorithm [23,9] does just this. It is a polynomial-time[2] algorithm for correcting (in a certain sense) up to t_{GS} errors, where t_{GS} is the largest integer strictly less than $n - \sqrt{(k-1)n}$, i.e.,

$$t_{GS} = n - 1 - \left\lfloor \sqrt{(k-1)n} \right\rfloor \tag{4}$$

[2] Conservatively, the time complexity is $O(n^2 m^4)$, where n is the code length and m is the interpolation multiplicity.

It is easy to show that $t_{GS} \geq t_0$, and often t_{GS} is considerably greater than t_0 (see the examples below). Asymptotically, for RS codes of rate R, the conventional decoding algorithms will correct a fraction $\tau_0 = (1-R)/2$ of errors, while the GS algorithm can correct up to $\tau_{\rm GS} = 1 - \sqrt{R}$.[3]

The GS decoder has an adjustable integer parameter $m \geq 1$ called the interpolation multiplicity. Associated with the interpolation multiplicity m is positive integer $t = t_m$, called the designed decoding radius. Given a received word, the $GS(m)$ decoder returns a list that includes all codewords with Hamming distance t_m or less from the received word, and perhaps a few others. The exact formula for t_m is a bit complicated, but for now it suffices to say that[4]

$$t_0 \leq t_1 \leq t_2 \leq \cdots$$

and there exists an integer m_0 such that

$$t_{m_0} = t_{m_0+1} = \cdots = t_{GS}$$

Here is an overview of the $GS(m)$ algorithm (a detailed description will be given in Section VI). Suppose $C = (f(\alpha_1), \cdots, f(\alpha_n))$ is the transmitted codeword, where $f(x)$ is a polynomial of degree $<k$, and that C is received as $R = (\beta_1, \cdots, \beta_n)$. Let $p(x)$ be any polynomial of degree $<k$ that maps to an RS codeword with Hamming distance $\leq t_m$ from R, i.e.,

$$|\{i : p(\alpha_i) \neq \beta_i\}| \leq t_m$$

The $GS(m)$ decoder "finds" $p(x)$ as follows.

(1) The interpolation step. Given the received vector $R = (\beta_1, \cdots, \beta_n)$, the decoder constructs a two-variable polynomial

$$Q(x,y) = \sum_{i,j} a_{i,j} x^i y^j$$

with the property that Q has a zero of multiplicity m (exact definition in Section IV) at each of the points (α_i, β_i), and for which the $(1, k-1)$ weighted degree (exact definition in Section III) of $Q(x,y)$ is as small as possible.

(2) The factorization step. The decoder then finds all factors of $Q(x,y)$ of the form $y - p(x)$, where $p(x)$ is a polynomial of degree $k-1$ or less. Let

$$\mathcal{L} = \{p_1(x), \cdots, p_L(x)\}$$

be the list of polynomials produced by this step. The polynomials (codewords) $p(x) \in \mathcal{L}$ are of three possible types:

[3] By the arithmetic-geometric mean inequality, $1 - \sqrt{R} \geq (1-R)/2$, with equality iff $R = 1$.

[4] We note in passing that the $GS(1)$ decoder is the original Sudan algorithm [23].

(a) Type 1. The transmitted, or causal, codeword.

(b) Type 2. Codewords with Hamming distance $\leq t_m$ from R, which we call plausible codewords.

(c) Type 3. Codewords with distance $> t_m$ from R, which we call implausible codewords.

In Section VI, we will give a proof of the following theorem.

Theorem 1. *If the $GS(m)$ decoding algorithm is used, all plausible codewords will be in $\mathcal{L}$. In particular, the transmitted codeword will be in $\mathcal{L}$ if the number of channel errors is $\leq t_m$. The list may also contain implausible codewords, but the total number of codewords in the list, plausible and implausible, will satisfy $L \leq L_m$, where the exact determination of L_m is given in Section VI, Eq. (45), but which is conservatively estimated by*

$$L_m < \left(m + \frac{1}{2}\right)\sqrt{\frac{n}{k-1}} \tag{5}$$

Example 2. Consider again the (32,8) RS code over $GF(32)$, with $r = 24$ and $d = 25$. Its conventional error-correcting capability is $t_0 = 12$ errors, but by Eq. (4), the GS algorithm can correct up to $t_{GS} = 17$ errors! The value of the designed decoding radius t_m as a function of the interpolation multiplicity m is given in the table below, together with the exact value of L_m as given in Eq. (45), and the value

$$\overline{L}(t) = q^{-r}\sum_{s=0}^{t}\binom{n}{s}(q-1)^t$$

which is the average number of codewords in a randomly chosen sphere of radius t, and which gives a heuristic upper bound on the probability that the decoding sphere will contain a noncausal codeword. Values of m that do not afford a larger value of t_m than the previous value are omitted. For example, in the present example, $t_2 = t_3 = 15$, and so $m = 3$ is omitted from the table. Similarly, $t_5 = t_6 = \cdots = t_{119} = 16$:

m	t_m	L_m	$\overline{L}(t_m)$
0	12	1	1.36305×10^{-10}
1	14	2	2.74982×10^{-07}
2	15	4	0.0000102619
4	16	8	0.000339205
$m_0 = 120$	17	256	0.00993659

It is interesting to note the growth in the required value of m as t increases from 16 ($m = 4$) to 17 ($m = 120$), which indicates that $t = 16$ is the practical limit for the GS algorithm in this case.

Example 3. Similarly, for the $(16, 4)$ RS code over $GF(16)$ ($t_0 = 6$ and $t_{GS} = 9$):

m	t_m	L_m	$\overline{L}(t_m)$
0	6	1	0.000336183
1	7	2	0.00728043
2	8	4	0.124465
$m_0 = 28$	9	64	1.68692

Here we see that for $t = t_{GS} = 9$, the interpolation multiplicity may be prohibitively large, so that $t = 8$ is the practical limit.

Example 4. For the $(31, 15)$ RS code over $GF(32)$, $t_0 = 8$ and $t_{GS} = 10$:

m	t_m	L_m	$\overline{L}(t_m)$
0	8	1	5.62584×10^{-06}
3	9	4	0.000446534
$m_0 = 21$	10	31	0.0305164

Example 5 [21, Example 7.1]. For the $(18, 2)$ RS code over $GF(19)$, $t_0 = 8$ and $t_{GS} = 13$:

m	t_m	L_m	$\overline{L}(t_m)$
0	8	1	1.74158×10^{-06}
1	12	4	0.0821209
$m_0 = 2$	13	9	0.700656

Note the large values of $\overline{L}(t)$, which may obviate the claim, e.g., that the code can correct almost all patterns of 13 errors.

Example 6. For the $(6, 4)$ RS code over $GF(7)$, $t_0 = t_{GS} = 1$:

m	t_m	L_m	$\overline{L}(t_m)$
$m_0 = 0$	1	1	0.7551

This is a rare example where the GS algorithm provides no improvement over conventional decoding.

Example 7. For the $(255, 223)$ RS code over $GF(256)$, $t_0 = 16$ and $t_{GS} = 17$:

m	t_m	L_m	$\overline{L}(t_m)$
0	16	1	2.609×10^{-14}
$m_0 = 112$	17	120	9.35×10^{-11}

Not until $m = 112$ does the GS algorithm offer an improvement over conventional decoders, and even then the improvement is only one extra error corrected. With the decoding complexity $O(m^4)$, it seems pointless to try to correct the extra error.

III. Polynomials in Two Variables I: Monomial Orders and Generalized Degree

In this section, we present a self-contained introduction to the algebraic fundamentals of two-variable polynomials. These fundamentals include weighted monomial orderings, generalized degree functions, and certain related combinatorial results.

If F is a field, we denote by $F[x, y]$ the ring of polynomials in x and y with coefficients from F. A polynomial $Q(x, y) \in F[x, y]$ is, by definition, a finite sum of monomials, viz.,

$$Q(x, y) = \sum_{i,j \geq 0} a_{i,j} x^i y^j \tag{6}$$

where only a finite number of the coefficients $a_{i,j}$ are nonzero. The summation in Eq. (6) is two-dimensional, but often it is desirable to have a one-dimensional representation instead. To do this, we need to have a linear ordering of the set of monomials

$$\mathbb{M}[x, y] = \{x^i y^j : i, j \geq 0\}$$

In this section, we will describe a general class of monomial orderings.

We first note that the set $\mathbb{M}[x, y]$ is isomorphic to the set $\mathbb{N}^2$ of pairs of nonnegative integers under the bijection $x^i y^j \leftrightarrow (i, j)$. A monomial ordering [7, Section 2.2] is a relation "$<$" on $\mathbb{M}[x, y]$ (equivalently, on $\mathbb{N}^2$) with the following three properties:[5]

$$\text{If } a_1 \leq b_1 \text{ and } a_2 \leq b_2, \text{ then } (a_1, a_2) \leq (b_1, b_2). \tag{7}$$

$$\text{The relation } \text{“} < \text{”} \text{ is a total ordering, i.e., if } \mathbf{a} \text{ and } \mathbf{b} \text{ are distinct monomials,}$$
$$\text{either } \mathbf{a} < \mathbf{b} \text{ or } \mathbf{b} < \mathbf{a}. \tag{8}$$

$$\text{If } \mathbf{a} < \mathbf{b} \text{ and } \mathbf{c} \in \mathbb{N}^2, \text{ then } \mathbf{a} + \mathbf{c} \leq \mathbf{b} + \mathbf{c}. \tag{9}$$

(Because of Property (7), "$<$" is said to be a linear extension of the partial order on $\mathbb{N}^2$ induced by the ordinary meaning of "$<$," applied componentwise.)

[5] In what follows, the symbol "$x \leq y$" will mean "either $x < y$ or $x = y$."

There are many possible monomial orderings, but for us the most important ones are the weighted degree (WD) monomial orders. A WD monomial order is characterized by a pair $\mathbf{w} = (u, v)$ of nonnegative integers, not both zero. For a fixed $\mathbf{w}$, the $\mathbf{w}$-degree of the monomial $x^i y^j$ is defined as

$$\deg_{\mathbf{w}} x^i y^j = ui + vj$$

If we order $\mathbb{M}[x, y]$ by $\mathbf{w}$-degree, i.e., declare that $\phi(x, y) < \phi'(x, y)$ if $\deg_{\mathbf{w}} \phi(x, y) < \deg_{\mathbf{w}} \phi'(x, y)$, we only get a partial order, since monomials with equal $\mathbf{w}$-degree are incomparable. It turns out that there are just two ways to break such ties so that Property (9) is satisfied: $\mathbf{w}$-lexicographic ($\mathbf{w}$-lex) order, and $\mathbf{w}$-reverse lexicographic ($\mathbf{w}$-revlex) order.

Definition 1. $\mathbf{w}$-lex order is defined as follows:

$$x^{i_1} y^{j_1} < x^{i_2} y^{j_2}$$

if either $ui_1 + vj_1 < ui_2 + vj_2$, or $ui_1 + vj_1 = ui_2 + vj_2$ and $i_1 < i_2$. $\mathbf{w}$-revlex order is similar, except that the rule for breaking ties is $i_1 > i_2$. (In the special case $\mathbf{w} = (1, 1)$, these orderings are called graded-lex, or grlex, and reverse graded-lex, or grevlex, respectively.)

Example 8. For any monomial order, we have $xy < x^2 y$ because of Property (7). Also, $xy^2 <_{\text{grlex}} x^2 y$, but $x^2 y <_{\text{grevlex}} xy^2$. Finally, if $\mathbf{w} = (1, 3)$, $x^6 <_{\text{wrevlex}} x^3 y <_{\text{wrevlex}} y^2$.

Let "$<$" be a fixed monomial ordering:

$$1 = \phi_0(x, y) < \phi_1(x, y) < \phi_2(x, y) < \cdots$$

With respect to this ordering, every nonzero polynomial in $F[x, y]$ can be expressed uniquely in the form

$$Q(x, y) = \sum_{j=0}^{J} a_j \, \phi_j(x, y) \tag{10}$$

for suitable coefficients $a_j \in F$, with $a_J \neq 0$. The integer J is called the rank of $Q(x, y)$, and the monomial ϕ_J is called the leading monomial of $Q(x, y)$. We indicate this notationally by writing $\text{Rank}(Q) = J$ and $\text{LM}(Q) = \phi_J(x, y)$. The relation $\text{LM}P = \text{LM}Q$ is an equivalence relation, which we denote by $P \equiv Q$. We can extend the order "$<$" to all of $F[x, y]$ by declaring $P < Q$ to mean $\text{LM}P < \text{LM}Q$. In this way, "$<$," which is a total order on the set of monomials, becomes a partial order on $F[x, y]$ and a total order on the equivalence classes under LM.

In the case of a WD order, the weighted degree of the leading monomial ϕ_j is also called the weighted degree, or $\mathbf{w}$-degree, of $Q(x, y)$, denoted $\deg_{\mathbf{w}} Q$. Thus,

$$\deg_{\mathbf{w}} Q(x, y) = \max\{\deg_{\mathbf{w}} \phi(x, y) : a_j \neq 0\}$$

The $\mathbf{w}$-degree function enjoys the following basic properties:

$$\deg_{\mathbf{w}} 0 = -\infty \tag{11}$$

$$\deg_{\mathbf{w}}(PQ) = \deg_{\mathbf{w}} P + \deg_{\mathbf{w}} Q \tag{12}$$

$$\deg_{\mathbf{w}}(P+Q) \leq \max(\deg_{\mathbf{w}} P, \deg_{\mathbf{w}} Q) \tag{13}$$

$$\deg_{\mathbf{w}}(P+Q) = \max(\deg_{\mathbf{w}} P, \deg_{\mathbf{w}} Q), \quad \text{if } \mathrm{LM}P \neq \mathrm{LM}\, Q \tag{14}$$

If $\phi_0(x,y) < \phi_1(x,y) < \cdots$ is a fixed monomial ordering, and $\phi = x^i y^j$ is a particular monomial, the index of ϕ, denoted $\mathrm{Ind}(\phi)$, is defined as the unique integer K such that $\phi_K(x,y) = \phi$.

Example 9. Here is a listing of the first few monomials, in the "natural" two-dimensional array, but labeled according to $(1,3)$-lex order:

$$\begin{array}{rc} i = & \begin{array}{ccccccccccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & \end{array} \\ j=0 & \left(\begin{array}{ccccccccccccccccc} 0 & 1 & 2 & 4 & 6 & 8 & 11 & 14 & 17 & 21 & 25 & 29 & 34 & 39 & 44 & 50 & \cdots \end{array}\right. \\ 1 & \begin{array}{ccccccccccccccccc} 3 & 5 & 7 & 10 & 13 & 16 & 20 & 24 & 28 & 33 & 38 & 43 & 49 & & & & \end{array} \\ 2 & \begin{array}{ccccccccccccccccc} 9 & 12 & 15 & 19 & 23 & 27 & 32 & 37 & 42 & 48 & & & & & & & \end{array} \\ 3 & \begin{array}{ccccccccccccccccc} 18 & 22 & 26 & 31 & 36 & 41 & 47 & & & & & & & & & & \end{array} \\ 4 & \left.\begin{array}{ccccccccccccccccc} 30 & 35 & 40 & 46 & & & & & & & & & & & & & \end{array}\right) \end{array}$$

Thus, we have $\phi_0 = 1$, $\phi_1 = x$, $\phi_2 = x^2$, $\phi_3 = y$, $\phi_4 = x^3, \cdots, \phi_{48} = x^9 y^2, \cdots$. Also, $\mathrm{Ind}(xy) = 5$, $\mathrm{Ind}(x^2y^2) = 15$, $\mathrm{Ind}(x^9y^2) = 48$, etc.

Example 10. Here is a listing of the first few monomials, in the "natural" two-dimensional array, but labeled according to $(1,3)$-revlex order:

$$\begin{array}{rc} i = & \begin{array}{ccccccccccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & \end{array} \\ j=0 & \left(\begin{array}{ccccccccccccccccc} 0 & 1 & 2 & 3 & 5 & 7 & 9 & 12 & 15 & 18 & 22 & 26 & 30 & 35 & 40 & 45 & \cdots \end{array}\right. \\ 1 & \begin{array}{ccccccccccccccccc} 4 & 6 & 8 & 10 & 13 & 16 & 19 & 23 & 27 & 31 & 36 & 41 & 46 & & & & \end{array} \\ 2 & \begin{array}{ccccccccccccccccc} 11 & 14 & 17 & 20 & 24 & 28 & 32 & 37 & 42 & 47 & & & & & & & \end{array} \\ 3 & \begin{array}{ccccccccccccccccc} 21 & 25 & 29 & 33 & 38 & 43 & 48 & & & & & & & & & & \end{array} \\ 4 & \left.\begin{array}{ccccccccccccccccc} 34 & 39 & 44 & & & & & & & & & & & & & & \end{array}\right) \end{array}$$

Thus, we have $\phi_0 = 1$, $\phi_1 = x$, $\phi_2 = x^2$, $\phi_3 = x^3$, $\phi_4 = y, \cdots, \phi_{48} = x^6 y^3, \cdots$. Also, $\mathrm{Ind}(xy) = 6$, $\mathrm{Ind}(x^2y^2) = 17$, $\mathrm{Ind}(x^9y^2) = 47$, etc. We shall see below that for $(1,v)$ revlex order, the numbers $\mathrm{Ind}(x^K)$ and $\mathrm{Ind}(y^L)$ are especially important, so we introduce a special notation for them:

$$A(K,v) \triangleq \mathrm{Ind}(x^K) \tag{15}$$

$$B(L,v) \triangleq \mathrm{Ind}(y^L) \tag{16}$$

it being understood that the underlying monomial order is $(1,v)$-revlex. In terms of the two-dimensional array given above, the numbers $A(K,v)$ appear in the $j=0$ row and the numbers $B(L,v)$ appear in the $i=0$ column. Thus, with $v=3$, we have

x	0	1	2	3	4	5	6	7	$\cdots$
$A(x,3)$	0	1	2	3	5	7	9	12	$\cdots$
$B(x,3)$	0	4	11	21	34	50	69	90	$\cdots$

We note that x^K is the first monomial of $(1,v)$-degree K, and y^L is the last monomial of $(1,v)$-degree vL, so that

$$A(K,v) = |\{(i,j) : i + vj < K\}| \tag{17}$$

$$B(L,v) = |\{(i,j) : i + vj \le Lv\}| - 1 \tag{18}$$

We conclude this section with a consideration of two-variable polynomials of the form

$$Q(x,y) = \sum_{j=0}^{J} a_j \phi_j(x,y)$$

where $\phi_0 < \phi_1 < \cdots$ is $(1,v)$-revlex order, and $\{a_0, a_1, \cdots, a_J\}$ are arbitrary elements of F. (N.B., We do not assume that $a_J \neq 0$.)

Two important questions that will arise are (1) what is the $(1,v)$-degree of $Q(x,y)$ and (2) what is the y-degree, i.e., the $(0,1)$-degree, of $Q(x,y)$? From Property (13), we know that

$$\deg_{1,v} Q(x,y) \le \max\{\deg_{1,v} \phi_j(x,y) : j = 0, \cdots, J\}$$

$$\deg_{0,1} Q(x,y) \le \max\{\deg_{0,1} \phi_j(x,y) : j = 0, \cdots, J\}$$

Thus, if we define (it being understood that the monomial order is $(1,v)$-revlex)

$$D(u,v;J) = \max\{\deg_{u,v} \phi_j(x,y) : j = 0, \cdots, J\} \tag{19}$$

we have the upper bounds

$$\deg_{1,v} Q(x,y) \le D(1,v;J)$$

$$\deg_{0,1} Q(x,y) \le D(0,1;J)$$

We need a definition. Let $A = \{0 = a_0 < a_1 < a_2 < \cdots\}$ be an increasing sequence of integers, and let $x \ge 0$ be a nonnegative real number. The rank of apparition[6] of x with respect to A, denoted $r_A(x)$, is the unique index K such that $a_K \le x < a_{K+1}$. Alternatively,

$$r_A(x) = \max\{K : a_K \le x\}$$

$$= \min\{L : x < a_{L+1}\}$$

[6] This amusing term was coined by Basil Gordon of UCLA.

Theorem 2. *With v fixed, define sequences $\{a_K = A(K,v)\}$ and $\{b_L = B(L,v)\}$. Then*

$$D(1,v;J) = r_A(J) \tag{20}$$

$$D(0,1;J) = r_B(J) \tag{21}$$

Proof. This is just a matter of observing that x^K is the first monomial of $(1,v)$ degree K and that y^L is the first monomial of $(0,1)$-degree L. ❒

Theorem 2 will be helpful only if we can compute the values $A(K,v)$ and $B(L,v)$.

Theorem 3. *For $K \geq 0$, let $r = K \bmod v$. Then*[7]

$$A(K,v) = \frac{K^2}{2v} + \frac{K}{2} + \frac{r(v-r)}{2v} \tag{22}$$

$$B(L,v) = \frac{vL^2}{2} + \frac{(v+2)L}{2} \tag{23}$$

Proof. The Equality (23) can be proved by induction, using the recursion

$$\begin{aligned} B(L,v) = & \ (|\{(i,j) : i + vj \leq (L-1)v\}| - 1) + |\{(i,j) : (L-1)v + 1 \leq i + vj \leq Lv\}| \\ = & \ B(L-1,v) + vL + 1 \end{aligned}$$

which follows from Eq. (18). Alternatively, Eq. (23) follows from Eq. (22), using the relationship $B(L,v) = A(vL+1,v)-1$ [see Eq. (18)]. The validity of Eq. (22) follows from Eq. (17): for each j such that $vj < K$, i must be in the range $0 \leq i < K - vj$, so that

$$\begin{aligned} A(K,v) = & \sum_{j=0}^{\lfloor K/v \rfloor} (K - jv) \\ = & \ v \sum_{j=0}^{\lfloor T \rfloor} (T - j), \quad \text{where } T = \frac{K}{v} \end{aligned}$$

Now apply Euler's summation formula [11, Section 1.2.11.2, Eq. (3)], which implies

$$\begin{aligned} \sum_{j=0}^{\lfloor T \rfloor} (T - j) = & \int_0^T (T - x)dx + \frac{T}{2} - \int_0^T \left\{ x - \frac{1}{2} \right\} dx \\ = & \ \frac{T^2}{2} + \frac{T}{2} + \frac{\{T\}(1 - \{T\})}{2} \end{aligned} \tag{24}$$

where $\{x\} \triangleq x - \lfloor x \rfloor$ is the fractional part of x. Equation (24) is equivalent to Eq. (22) since $\{T\} = r/v$. ❒

[7] Formula (22) is similar to, but simpler than, those given in [9, Lemma 6] and [14, Lemma 1].

Corollary 1. $A(n,v) = vF(n/v)$, where

$$F(x) = \frac{1}{2}\big(x^2 + x + \{x\}(1-\{x\})\big)$$

(Some important properties of the function $F(x)$ are described in Theorem A-1.)

Corollary 2. For $v \geq 1$, $K \geq 0$,

$$\frac{K^2}{2v} < A(K,v) \leq \frac{(K+v/2)^2}{2v} \tag{25}$$

Corollary 3. For $v \geq 1$, $J \geq 0$,

$$\left\lfloor \sqrt{2vJ} - \frac{v}{2} \right\rfloor \leq r_A(J) \leq \left\lfloor \sqrt{2vJ} \right\rfloor - 1$$

Proof. These inequalities follow by combining Eq. (25) with Eq. (27), to be proved below. ❒

Corollary 4. For $v \geq 1$, $J \geq 0$

$$r_B(J) = \left\lfloor \sqrt{\frac{2J}{v} + \left(\frac{v+2}{2v}\right)^2} \right\rfloor - \left(\frac{v+2}{2v}\right)$$

and hence

$$\left\lfloor \sqrt{\frac{2J}{v}} - \frac{v+2}{2v} \right\rfloor \leq r_B(J) \leq \left\lfloor \sqrt{\frac{2J}{v}} \right\rfloor$$

Proof. These facts follow by combining Eq. (23) with Eq. (26). ❒

Lemma 1. If $a_K = f(K)$, where $f(x)$ is a continuous increasing function of $x > 0$, then

$$r_A(x) = \lfloor f^{-1}(x) \rfloor \tag{26}$$

More generally, if $g(K) \leq a_K \leq f(K)$, where $f(x)$ and $g(x)$ are both continuous, increasing functions of $x > 0$, then

$$\lfloor f^{-1}(x) \rfloor \leq r_A(x) \leq \lfloor g^{-1}(x) \rfloor \tag{27}$$

Proof. Suppose $K = r_A(x)$. Then by definition,

$$g(K) \leq a_K \leq x < a_{K+1} \leq f(K+1)$$

Thus, $K \leq g^{-1}(x)$ and $f^{-1}(x) < K+1$, i.e., $r_A(x) \leq g^{-1}(x)$ and $f^{-1}(x) < r_A(x)+1$, i.e.,

$$f^{-1}(x) - 1 < r_A(x) \leq g^{-1}(x)$$

The desired result, Eq. (27), follows immediately, if we recall that $r_A(x)$ is an integer. ❒

IV. Polynomials in Two Variables II: Zeros and Multiple Zeros

In this section, we continue with our study of bivariate polynomials and focus on the notion of a zero, or a multiple zero, of such polynomials.

If $Q(x,y) \in F[x,y]$, and $Q(\alpha,\beta) = 0$, we say that Q has a *zero* at (α,β).[8] We shall be interested in polynomials with *multiple* zeros.

Definition 2. We say that $Q(x,y) = \sum_{i,j} a_{i,j}\, x^i y^j \in F[x,y]$ has a zero of multiplicity, or order m at $(0,0)$, and write

$$\text{ord}\,(Q:0,0) = m$$

if $Q(x,y)$ involves no term of total degree less than m, i.e., $a_{i,j} = 0$ if $i+j < m$. Similarly, we say that $Q(x,y)$ has a zero of order m at (α,β) and write

$$\text{ord}\,(Q:\alpha,\beta) = m$$

if $Q(x+\alpha, y+\beta)$ has a zero of order m at $(0,0)$.

Example 11. Let $Q(x,y) = x^2y + xy^3 + x^3y$. Then Q has a zero of multiplicity 3 (a "triple zero") at $(0,0)$. Similarly, $P(x,y) = (x-\alpha)^2(y-\beta) + (x-\alpha)(y-\beta)^3 + (x-\alpha)^3(y-\beta)$ has a triple zero at (α,β).

To calculate $\text{ord}\,(Q:\alpha,\beta)$, we need to be able to express $Q(x+\alpha, y+\beta)$ as a polynomial in x and y. The following theorems, due to H. Hasse [10], tell us one way to do this. We begin with the one-variable version of Hasse's theorem, both because it serves as a simplified introduction to the two-variable case and because we will need the one-variable theorem in Section IX (Lemma 7).

Theorem 4. *If $Q(x) = \sum_i a_i x^i \in F[x]$, then for any $\alpha \in F$, we have*

$$Q(x+\alpha) = \sum_r Q_r(\alpha) x^r \tag{28}$$

where

$$Q_r(x) = \sum_i \binom{i}{r} a_i x^{i-r} \tag{29}$$

[8] Alternatively, we say that the curve $Q(x,y) = 0$ *passes through* the point (α,β).

which is called the rth Hasse derivative of $Q(x)$.[9] *Note also that*

$$Q_r(\alpha) = \operatorname*{Coeff}_{x^r} Q(x+\alpha) = \sum_i \binom{i}{r} a_i \alpha^{i-r} \tag{30}$$

Note that Eq. (28) is Taylor's formula (without remainder) when F has characteristic 0, since in that case,

$$Q_r(x) = \frac{1}{r!}\frac{d^r}{dx^r} Q(x)$$

Corollary 5. We also have

$$Q(x) = \sum_{r \geq 0} Q_r(\alpha)(x-\alpha)^r$$

Theorem 5. *Let* $Q(x,y) = \sum_{i,j} a_{i,j} x^i y^j \in F[x,y]$. *For any* $(\alpha,\beta) \in F^2$, *we have*

$$Q(x+\alpha, y+\beta) = \sum_{r,s} Q_{r,s}(\alpha,\beta) x^r y^s \tag{31}$$

where

$$Q_{r,s}(x,y) = \sum_{i,j} \binom{i}{r}\binom{j}{s} a_{i,j} x^{i-r} y^{j-s} \tag{32}$$

which is called the (r,s)*th Hasse (mixed partial) derivative of* $Q(x,y)$.[10] *Note that Eq. (31) is Taylor's formula (without remainder) when* F *has characteristic* 0, *since in that case,*

$$Q_{r,s}(x,y) = \frac{1}{r!s!}\frac{\partial^{r+s}}{\partial x^r \partial y^s} Q(x,y)$$

Note also the alternative, but equivalent, formula:

$$Q_{r,s}(\alpha,\beta) = \operatorname*{Coeff}_{x^r y^s} Q(x+\alpha, y+\beta) \tag{33}$$

Proof. Using the binomial theorem, we express $Q(x+\alpha, y+\beta)$ as a polynomial in x and y:

[9] We will sometimes use the alternative notation $D_r Q(x)$ instead of $Q_r(x)$.

[10] We will sometimes use the alternative notation $D_{r,s} Q(x,y)$ instead of $Q_{r,s}(x,y)$.

$$\begin{aligned}
Q(x+\alpha, y+\beta) &= \sum_{i,j} a_{i,j}(x+\alpha)^i(y+\beta)^j \\
&= \sum_{i,j} a_{i,j}\left(\sum_r \binom{i}{r} x^r \alpha^{i-r}\right)\left(\sum_s \binom{j}{s} y^s \beta^{j-s}\right) \\
&= \sum_{r,s} x^r y^s \left(\sum_{i,j} \binom{i}{r}\binom{j}{s} a_{i,j}\alpha^{i-r}\beta^{j-s}\right) \\
&= \sum_{r,s} Q_{r,s}(\alpha,\beta) x^r y^s
\end{aligned}$$

❒

Corollary 6. We also have

$$Q(x,y) = \sum_{r,s} Q_{r,s}(\alpha,\beta)(x-\alpha)^r(y-\beta)^s$$

Corollary 7. The polynomial $Q(x,y)$ has a zero of order m at (α,β) if and only if

$$Q_{r,s}(\alpha,\beta) = 0 \qquad \text{for all } r \text{ and } s \text{ such that } 0 \le r+s < m \tag{34}$$

Proof. By definition, $\operatorname{ord}(Q:\alpha,\beta) \ge m$ iff $Q(x+\alpha, y+\beta)$ has a zero of order m at $(0,0)$. But by Eq. (31), $Q(x+\alpha, y+\beta)$ has a zero of order m at $(0,0)$ iff $Q_{r,s}(\alpha,\beta) = 0$ for all $0 \le r+s < m$. ❒

Corollary 8. If $\widetilde{Q}(x,y) = xQ(x,y)$, then

$$\widetilde{Q}_{r,s}(x,y) = Q_{r-1,s}(x,y) + xQ_{r,s}(x,y)$$

Similarly, if $\widetilde{Q}(x,y) = yQ(x,y)$, then

$$\widetilde{Q}_{r,s}(x,y) = Q_{r,s-1}(x,y) + yQ_{r,s}(x,y)$$

Proof. By definition, $\widetilde{Q}_{r,s}(\alpha,\beta) = \operatorname{coeff}_{x^r y^s}\widetilde{Q}(x+\alpha, y+\beta)$. But from Eq. (31), we have

$$\begin{aligned}
\widetilde{Q}(x+\alpha, y+\beta) &= (x+\alpha)Q(x+\alpha, y+\beta) \\
&= (x+\alpha)\sum_{r,s} Q_{r,s}(\alpha,\beta)x^r y^s \\
&= \sum_{r,s} Q_{r,s}(\alpha,\beta)x^{r+1}y^s + \sum_{r,s}\alpha Q_{r,s}(\alpha,\beta)x^r y^s \\
&= \sum_{r,s}\left(Q_{r-1,s}(\alpha,\beta) + \alpha Q_{r,s}(\alpha,\beta)\right)x^r y^s
\end{aligned}$$

Therefore, $\widetilde{Q}_{r,s}(\alpha,\beta) = Q_{r-1,s}(\alpha,\beta) + \alpha Q_{r,s}(\alpha,\beta)$. ❒

V. The Interpolation and Factorization Theorems

In this section, we will state and prove the two basic theorems that support the GS algorithm. We call these theorems the Interpolation Theorem and the Factorization Theorem.

A. The Interpolation Theorem

Suppose a nonnegative integer $m(\alpha)$ is assigned to each element $\alpha \in F$, and we are asked to construct a polynomial $f(x)$ of least degree that has a zero of multiplicity $m(\alpha)$, at $x = \alpha$, for all $\alpha \in F$. Clearly a minimum-degree solution to this one-dimensional interpolation problems is

$$f(x) = \prod_{\alpha \in F} (x - \alpha)^{m(\alpha)}$$

$$\deg f(x) = \sum_{\alpha \in F} m(\alpha)$$

We are interested in the analogous two-dimensional interpolation problem: Given a required multiplicity $m(\alpha, \beta)$ for each $(\alpha, \beta) \in F^2$, construct a low-degree polynomial $Q(x, y)$ that has zeros of the required multiplicity. This is a much harder problem, in general, but the following theorem gives a useful upper bound on the minimum required degree.

Theorem 6: The Interpolation Theorem. *Let $\{m(\alpha, \beta) : (\alpha, \beta) \in F^2\}$ be a multiplicity function as above and let $\phi_0 < \phi_1 < \cdots$ be an arbitrary monomial order. Then there exists a nonzero polynomial $Q(x, y)$ of the form*

$$Q(x, y) = \sum_{i=0}^{C} a_i \, \phi_i(x, y) \tag{35}$$

where

$$C = \sum_{\alpha, \beta} \binom{m(\alpha, \beta) + 1}{2}$$

which has a zero of multiplicity $m(\alpha, \beta)$, at $(x, y) = (\alpha, \beta)$, for all $(\alpha, \beta) \in F^2$.

Proof. By Corollary 7, $Q(x, y)$ has a zero of multiplicity m at (α, β) if and only if

$$Q_{r,s}(\alpha, \beta) = 0 \quad \text{for all } (r, s) \text{ such that } 0 \le r + s < m(\alpha, \beta) \tag{36}$$

There are $\binom{m(\alpha,\beta)+1}{2}$ choices for (r, s) in Eq. (36), and by Eq. (32), each such choice imposes one homogeneous linear constraint on the coefficients a_i. In total there are C such linear constraints imposed on the $C + 1$ coefficients $a_0, a_1, \cdots, a_C$. It follows that there must be at least one nonzero solution to this set of equations, which corresponds to a nonzero polynomial $Q(x, y)$ of the form in Eq. (35) with the required multiplicities. ❒

Corollary 9. For any (u, v), there is a nonzero polynomial $Q(x, y)$ with the required zero multiplicities whose (u, v)-degree is strictly less than $\sqrt{2uvC}$.

Proof. Take $\{\phi_j(x,y)\}$ to be (u,v)-revlex order. Then by Eq. (35),

$$\deg_{u,v} Q(x,y) \le \max\{\deg_{u,v}\phi_j(x,y) : j = 0,\cdots,C\} = \deg_{u,v}\phi_C(x,y) = r_A(C)$$

where $A = (a_K)$ is the sequence $\mathrm{Ind}(x^K)$, for (u,v)-revlex order. But $r_A(C) < \sqrt{2uvC}$ by a straightforward generalization of Corollary 3. ❒

B. The Factorization Theorem

If $Q(x,y) \in F[x,y]$, and $f(x) \in F[x]$, define the Q-score of f as

$$S_Q(f) = \sum_{\alpha\in F} \mathrm{ord}\big(Q : \alpha, f(\alpha)\big)$$

Theorem 7: The Factorization Theorem. *Suppose* $f(x) \in F_v[x]$, $Q(x,y) \in F[x,y]$, *and*

$$S_Q(f) > \deg_{1,v} Q$$

Then $y - f(x)$ *is a factor of* $Q(x,y)$.

Proof. Let $Q(x,y) = \sum_{i,j} a_{i,j}x^i y^j$. Then $Q\big(x,f(x)\big)$ is a polynomial in x:

$$Q\big(x,f(x)\big) = \sum_{i,j\ge 0} a_{i,j}x^i f(x)^j \tag{37}$$

The following three lemmas describe important properties of this polynomial.

Lemma 2. If $f(x) \in F_v[x]$, then $\deg Q\big(x,f(x)\big) \le \deg_{1,v} Q(x,y)$.

Proof. For $a_{i,j} \ne 0$, $\deg(x^i f(x)^j) \le \deg(x^i x^{vj}) = i + vj \le \max(i + vj : a_{i,j} \ne 0) = \deg_{1,v} Q(x,y)$. ❒

Lemma 3. $Q\big(x,f(x)\big) = 0$ if and only if $\big(y - f(x)\big) \,|\, Q(x,y)$.

Proof. Let us view $Q(x,y)$ as a polynomial in y over the rational field $F(x)$. Then by the division algorithm, we can write

$$Q(x,y) = Q_0(x,y)\big(y - f(x)\big) + r(x) \tag{38}$$

where $r(x) \in F(x)$. Substituting $f(x)$ for y in Eq. (38), we obtain

$$Q\big(x,f(x)\big) = r(x)$$

so that $Q\big(x,f(x)\big) = 0$ if and only if $r(x) = 0$, which is equivalent to the stated result. ❒

Lemma 4. If $\operatorname{ord}(Q:\alpha,\beta) = K$, and $f(\alpha) = \beta$, then

$$(x-\alpha)^K \mid Q\big(x, f(x)\big)$$

Proof. Using Corollary 6, express $Q(x,y)$ as a polynomial in $x-\alpha$ and $y-\beta$:

$$Q(x,y) = \sum_{i,j} b_{i,j}(x-\alpha)^i(y-\beta)^j$$

Then

$$Q\big(x, f(x)\big) = \sum_{i,j} b_{i,j}(x-\alpha)^i\big(f(x)-\beta\big)^j \tag{39}$$

since $f(\alpha) = \beta$, $f(x)-\beta$ is divisible by $x-\alpha$, so that the term $(x-\alpha)^i\big(f(x)-\beta\big)^j$ in Eq. (39) is divisible by $(x-\alpha)^{i+j}$. But $\operatorname{ord}(Q:\alpha,\beta) = K$ implies that if $b_{i,j} \neq 0$, then $i+j \geq K$. Thus, every nonzero term in Eq. (39) is divisible by $(x-\alpha)^K$, i.e., $(x-\alpha)^K \mid Q\big(x, f(x)\big)$. ❒

We can now complete the proof of Theorem 7. By Lemma 4, we know that $\prod_{\alpha\in F}(x-\alpha)^{\operatorname{ord}(Q:\alpha,f(\alpha))} \mid Q\big(x, f(x)\big)$. But by Lemma 2, the degree of $Q\big(x, f(x)\big)$ is (at most) $\deg_{1,v} Q(x,y)$, and the degree of $\prod_{\alpha\in F}(x-\alpha)^{\operatorname{ord}(Q:\alpha,f(\alpha))}$ is $S_Q(f)$. Thus, if $S_Q(f)$ exceeds $\deg_{1,v} Q$, it follows that $Q\big(x, f(x)\big) = 0$, and so by Lemma 3, $y - f(x)$ divides $Q(x,y)$. ❒

VI. A Second Look at the Guruswami–Sudan Algorithm

Armed with the preliminary material from Sections III through V, in this section we will give a formal description, and proof of correctness, of the Guruswami–Sudan algorithm.

A. Prerequisite Notation, Concepts, etc.

Here we list some of the technical details needed for a full discussion of the GS algorithm:

- $K(f,\beta) = |\{i : f(\alpha_i) = \beta_i\}|$, $D(f,\beta) = |\{i : f(\alpha_i) \neq \beta_i\}|$.
- $C(n,m) = n\binom{m+1}{2}$.
- $(1,v)$-revlex monomial order.
- The indices $A(K,v) = \operatorname{Ind}(x^K)$ and $B(L,v) = \operatorname{Ind}(y^L)$ (with respect to $(1,v)$-revlex order), with the rank of apparition functions

$$r_A(J) = \max\{K : A(K,v) \leq J\}$$

$$r_B(J) = \max\{L : B(L,v) \leq J\}$$

- The numbers K_m, t_m, and L_m:

$$K_m(n,k) \triangleq \min\{K : A(mK, v) > C(n,m)\} = 1 + \lfloor r_A(C)/m \rfloor \tag{40}$$

$$t_m(n,k) \triangleq n - K_m(n,k) = n - 1 - \lfloor r_A(C)/m \rfloor \tag{41}$$

$$L_m(n,k) \triangleq \max\{L : B(L,v) \le C(n,m)\} = r_B(C) \tag{42}$$

- Estimates of K_m and L_m (from Corollaries 3 and 4):

$$\left\lfloor \sqrt{vn\frac{m+1}{m}} - \frac{v}{2m} \right\rfloor + 1 \le K_m \le \left\lfloor \sqrt{vn\frac{m+1}{m}} \right\rfloor \tag{43}$$

$$n - \left\lfloor \sqrt{vn\frac{m+1}{m}} \right\rfloor \le t_m \le n - 1 - \left\lfloor \sqrt{vn\frac{m+1}{m}} - \frac{v}{2m} \right\rfloor \tag{44}$$

$$L_m = \left\lfloor \sqrt{\frac{n}{v}m(m+1) + \left(\frac{v+2}{2v}\right)^2} - \frac{v+2}{2v} \right\rfloor < \left(m + \frac{1}{2}\right)\sqrt{\frac{n}{v}} \tag{45}$$

B. The GS Decoding Algorithm, in Detail

Given an (n,k) RS code over the finite field F, with support set $(\alpha_1, \cdots, \alpha_n)$, and a positive integer m, the $GS(m)$ decoder accepts a vector $\beta = (\beta_1, \cdots, \beta_n) \in F^n$ as input, and produces a list of polynomials $\{f_1, \cdots, f_L\}$ as output. Here's how:

1. The *GS*(*m*) Decoder. The $GS(m)$ decoder constructs a nonzero two-variable polynomial of the form

$$Q(x,y) = \sum_{j=0}^{C(n,m)} a_j \phi_j(x,y)$$

where $\phi_0 < \phi_1 < \cdots$ is $(1,v)$-revlex monomial order, such that $Q(x,y)$ has a zero of order m at each of the n points (α_i, β_i), for $i = 1, \cdots, n$. (The Interpolation Theorem, Theorem 6, guarantees that such a polynomial exists.) The output of the algorithm is the list of y-roots of $Q(x,y)$, i.e.,

$$\mathcal{L} = \{f(x) \in F[x] : \big(y - f(x)\big) \,|\, Q(x,y)\}$$

Theorem 8. *The output list contains every polynomial of degree $\le v$ such that $K(f,\beta) \ge K_m$. Furthermore, the number of polynomials in the list is at most L_m.*

Proof. By Eq. (20), $\deg_{1,v} Q(x,y) \le \max\{\deg_{1,v} \phi_i(x,y) : i = 0, \cdots, C\} = r_A(C)$. Hence, by Theorem 7, any polynomial $f(x)$ of degree $\le v$ such that $mK(f,\beta) > r_A(C)$, will be a y-root of $Q(x,y)$. In other words, if $K(f,\beta) \ge 1 + \lfloor r_A(C)/m \rfloor = K_m$, $f(x)$ will be on the list.

On the other hand, by Eq. (21), the y-degree of $Q(x, y)$ is $\leq r_B(C(n, m)) = L_m$. Since the number of y-roots of $Q(x, y)$ cannot exceed its y-degree, it follows that the output list contains at most L_m polynomials. ❒

With the basic theory out of the way, in the following three sections we describe low-complexity algorithms for solving the interpolation and factorization problems.

VII. Kötter's Solution to the Interpolation Problem

In this section, we will give a complete description of Kötter's solution to the interpolation problem. Much of this material has apparently never before appeared in print.

In general terms, the interpolation problem is to construct a bivariate polynomial $Q(x, y)$ with minimal $(1, v)$-degree that satisfies a number of constraints of the form

$$D_{r,s}Q(\alpha, \beta) = 0$$

where $(r, s) \in \mathbb{N}^2$ and $(\alpha, \beta) \in F^2$. It turns out that the mapping

$$Q(x, y) \mapsto D_{r,s}Q(\alpha, \beta)$$

is an example of what is called a linear functional on $F[x, y]$. It is no harder mathematically, and much easier notationally, to consider the more general problem of constructing a bivariate polynomial $Q(x, y)$ of minimal weighted-degree that satisfies a number of constraints of the form

$$D_iQ(x, y) = 0, \quad \text{for } i = 1, 2, \cdots$$

where each D_i is a linear functional. The goal of this section is to describe an algorithm for solving the more general problem.

A. Linear Functionals on $F[x,y]$

A mapping $D : F[x, y] \mapsto F$ is called a linear functional if

$$D(\alpha P + \beta Q) = \alpha D(P) + \beta D(Q) \tag{46}$$

for all $P, Q \in F[x, y]$ and all $\alpha, \beta \in F$. For us, the primary example of a linear functional is the mapping that evaluates a Hasse derivative:

$$Q(x, y) \mapsto D_{r,s}Q(\alpha, \beta)$$

for fixed values of $(r, s) \in \mathbb{N}^2$ and $(\alpha, \beta) \in F^2$.

If we agree on a particular monomial order, say

$$\phi_0(x, y) < \phi_1(x, y) < \cdots$$

so that any polynomial $Q(x, y)$ has a unique expansion of the form

$$Q(x,y) = \sum_{j=0}^{J} a_j \ \phi_j(x,y)$$

where $a_J \neq 0$, then any linear functional can be expressed as

$$D(Q) = \sum_{i=0}^{J} a_j d_j$$

where $d_j = D\big(\phi_j(x,y)\big)$. The kernel of D is defined to be the set

$$K = \ker D = \{Q : D(Q) = 0\} \tag{47}$$

If D is a linear functional with kernel K, the corresponding bilinear mapping $[P,Q]_D$ is defined as

$$[P,Q]_D \triangleq D(Q)P - D(P)Q \tag{48}$$

This simple mapping is a crucial part of the algorithms we present below; its key properties are given in the following lemma.

Lemma 5. For all P, Q in $F[x,y]$, $[P,Q]_D \in \ker D$. Furthermore, if $P > Q$ and $Q \notin K$, then $\text{Rank}\,[P,Q]_D = \text{Rank}\ P$.

Proof. To simplify the notation, let $\alpha = D(Q)$ and $\beta = D(P)$. Then $D([P,Q]_D) = D(\alpha P - \beta Q) = \alpha D(P) - \beta D(Q) = \alpha\beta - \beta\alpha = 0$, which proves $[P,Q]_D \in \ker D$. If $\alpha \neq 0$ and $P > Q$, the expression $[P,Q]_D = \alpha P - \beta Q$ shows that $\text{LM}\,[P,Q]_D = \text{LM}\,P$, so that $\text{Rank}\,[P,Q]_D = \text{Rank}\,P$. ❒

B. Problem Statement

Let $F_L[x,y]$ denote the set of polynomials from $F[x,y]$ whose y-degree is $\leq L$, i.e., those of the form

$$Q(x,y) = \sum_{k=0}^{L} q_k(x) y^k \tag{49}$$

where each $q_k(x) \in F[x]$. We note that $F_L[x,y]$ is an $F[x]$-module, i.e., if $Q(x,y) \in F_L[x,y]$, and $p(x) \in F[x]$, then $p(x)Q(x,y) \in F_L[x,y]$ as well.

Let $D_1, \cdots, D_C$ be C linear functionals defined on $F_L[x,y]$, and let $K_1, \cdots, K_C$ be the corresponding kernels, i.e.,

$$K_i = \{Q(x,y) \in F_L[x,y] : D_i(Q) = 0\}$$

The cumulative kernels $\overline{K}_0, \cdots, \overline{K}_C$ are defined as follows: $\overline{K}_0 = F_L[x,y]$ and for $i = 1, \cdots, C$,

$$\begin{aligned}\overline{K}_i &= \overline{K}_{i-1} \cap K_i \\ &= K_1 \cap \cdots \cap K_i \\ &= \{Q(x,y) \in F_L[x,y] : D_1(Q) = \cdots = D_i(Q) = 0\}\end{aligned}$$

Problem 1: The Generalized Interpolation Problem. Construct a minimal[11] element from

$$\overline{K}_C = K_1 \cap \cdots \cap K_C$$

i.e., calculate

$$Q_0(x,y) \in \min\{Q(x,y) : D_1(Q) = \cdots = D_C(Q) = 0\}$$

As we noted above, the linear functional D_i can be written as

$$D_i(Q) = \sum_{j \geq 0} a_j d_{j,i}$$

for suitable coefficients $d_{j,i} \in F$. Thus, the functionals $D_1, \cdots, D_C$ can be represented by a matrix with C rows and a column for each monomial $\phi_j(x,y)$:

$$\mathcal{D} = \begin{array}{c} \begin{array}{ccc} \phi_0 & \phi_1 & \cdots \end{array} \\ \begin{pmatrix} d_{1,0} & d_{1,1} & \cdots \\ \vdots & & \\ d_{C,0} & d_{C,1} & \cdots \end{pmatrix} \end{array}$$

If $\mathbf{d}^{(j)}$ denotes the jth column of $\mathcal{D}$, i.e.,

$$\mathbf{d}^{(j)} = (d_{1,j}, d_{2,j}, \cdots, d_{C,j})^T$$

the condition $Q(x,y) \in \overline{K}_C$, i.e.,

$$D_i\big(Q(x,y)\big) = 0 \quad \text{for } i = 1, \cdots, C$$

where $Q(x,y)$ is expressed as in Eq. (49), is equivalent to $a_0\mathbf{d}^{(0)} + \cdots + a_J\mathbf{d}^{(J)} = 0$. Thus, the least J such that the first J columns of $\mathcal{D}$ are linearly dependent corresponds to a polynomial of least rank that lies in $\overline{K}_C$. It turns out that there is an established algorithm, the Feng–Tseng (FT) algorithm, that is exactly suited to this formulation of the interpolation problem. We will discuss the FT algorithm in Section VIII.

[11] Here and elsewhere, "minimal" means minimal rank with respect to the given monomial order.

C. Kötter's Algorithm

Kötter (K) [12,13] noticed, in effect, that if the cumulative kernels are $F[x]$-modules, Problem 1 admits of a less complex solution than the one afforded by the FT algorithm.[12] This observation applies to the GS interpolation problem, since Corollary 8 says that if we enforce the conditions $D_{r,s}(\alpha, \beta) = 0$ for $s+r < m$ in an order in which $(r-1, s)$ always precedes (r, s), the cumulative kernels will be $F[x]$-modules. For example, $(m-1, 1)$ lex order, which orders the pairs

$$(0,0), (0,1), \cdots, (0, m-1), (1,0), (1,1), \cdots, (1, m-2), \cdots, (m-1, 0)$$

has the desired property.

In this subsection, we will describe, and prove the correctness of, Kötter's algorithm for solving this restricted class of problems.

In Kötter's algorithm, the set of monomials from $F_L[x, y]$, viz.,

$$\mathbb{M}_L[x, y] = \{x^i y^j : 0 \le i,\ 0 \le j \le L\}$$

is partitioned according to the exponent of y: $\mathbb{M}_L[x, y] = \bigcup_{j=0}^{L} \mathbb{M}_j$, where

$$\mathbb{M}_j = \{x^i y^j : i \ge 0\} \tag{50}$$

This partition of $\mathbb{M}_L$ induces a partition on $F_L[x, y]$: $F_L[x, y] = S_0 \cup \cdots \cup S_L$, where

$$S_j = \{Q \in F_L[x, y] : \mathrm{LM}(Q) \in \mathbb{M}_j\} \tag{51}$$

Kötter's algorithm generates a sequence of lists $G_0, G_1, \cdots, G_C$, with

$$G_i = (g_{i,0}, \cdots, g_{i,L})$$

where $g_{i,j}$ is a minimal element of $\overline{K}_i \cap S_j$. The algorithm's output is the polynomial

$$Q_0(x, y) = \min_{0 \le j \le L} g_{C,j}(x, y)$$

which is a minimal element of $\overline{K}_C$.

Kötter's algorithm is initialized as follows:

$$g_{0,j} = y^j, \quad j = 0, \cdots, L$$

[12] $O(n^2)$ for K versus $O(n^3)$ for FT.

Given G_i, G_{i+1} is defined recursively:

$$J_0 = \{j : D_{i+1}(g_{i,j}) = 0\}$$

$$J_1 = \{j : D_{i+1}(g_{i,j}) \neq 0\}$$

If J_1 is not empty, among the polynomials $g_{i,j}$ with $j \in J_1$, let g_{i,j^*} be the one with minimal rank, and temporarily denote g_{i,j^*} by f:

$$\left.\begin{aligned} f &= \min_{j \in J_1} g_{i,j} \\ j^* &= \operatorname*{argmin}_{j \in J_1} g_{i,j} \end{aligned}\right\} \tag{52}$$

Then using the notation of Eq. (48), $g_{i+1,j}$ is defined for $j = 0, \cdots, L$:

$$g_{i+1,j} = \begin{cases} g_{i,j} & \text{if } j \in J_0 \\ [g_{i,j}, f]_{D_{i+1}} & \text{if } j \in J_1 \text{ but } j \neq j^* \\ [xf, f]_{D_{i+1}} & \text{if } j = j^* \end{cases} \tag{53}$$

Theorem 9. *For $i = 0, \cdots, C$, we have*

$$g_{i,j} = \min\{g : g \in \overline{K}_i \cap S_j\} \quad \text{for } j = 0, \cdots, L \tag{54}$$

Proof. Induction on i: The case $i = 0$ being easily verified, let us assume the truth of Eq. (54) for the index i, and consider the index $i + 1$. We must show that Eq. (54) (with i replaced by $i + 1$) holds for $j = 0, \cdots, L$. There are three cases to consider, cf., Eq. (53).

Case 1: $j \in J_0$. In this case, we have, from Eq. (53),

$$g_{i+1,j} = g_{i,j}$$

Since $g_{i,j} \in \overline{K}_i \cap S_j$ by the induction hypothesis and $g_{i,j} \in K_{i+1}$ because $D_{i+1}(g_{i,j}) = 0$, it follows that $g_{i+1,j} \in \overline{K}_{i+1} \cap S_j$. But since $g_{i,j}$ is minimal in $\overline{K}_i \cap S_j$, it must also be minimal in the smaller set $\overline{K}_{i+1} \cap S_j$.

Case 2: $j \in J_1$ but $j \neq j^*$. In this case, we have, from Eq. (53),

$$g_{i+1,j} = [g_{i,j}, f]_{D_{i+1}}$$

Thus, $g_{i+1,j}$ is an F-linear combination of $g_{i,j}$ and f, which are both elements of $\overline{K}_i$ by the induction hypothesis. Thus, $g_{i+1,j} \in \overline{K}_i$ as well. Also, $g_{i+1,j} \in K_{i+1}$ by Lemma 5 and so $g_{i+1,j} \in \overline{K}_i \cap K_{i+1} = \overline{K}_{i+1}$.

By the induction hypothesis, $g_{i,j} \in S_j$ and $f \in S_{j^*}$, where $j \neq j^*$, which implies $\operatorname{Rank} g_{i,j} \neq \operatorname{Rank} f$. Thus, by Eq. (52), $g_{i,j} > f$. It follows from Lemma 5 that $\operatorname{Rank} g_{i+1,j} = \operatorname{Rank} g_{i,j}$ and hence (since $g_{i,j} \in S_j$) that $g_{i+1,j} \in S_j$ as well.

But since $g_{i+1,j}$ has the same rank as $g_{i,j}$, which is minimal in $\overline{K}_i \cap S_j$, it must be minimal in the smaller set $\overline{K}_{i+1} \cap S_j$ as well.

Case 3: $j = j^*$. In this case, we have, from Eq. (53),

$$g_{i+1,j} = [xf, f]_{D_{i+1}}$$

Thus, $g_{i+1,j}$ is an F-linear combination of xf and f. But $f \in \overline{K}_i$ by the induction hypothesis, and $xf \in \overline{K}_i$ because $\overline{K}_i$ is an $F[x]$-module.[13] Thus, $g_{i+1,j} \in \overline{K}_i$. Also, $g_{i+1,j} \in K_{i+1}$ by Lemma 5 and so $g_{i+1,j} \in \overline{K}_i \cap K_{i+1} = \overline{K}_{i+1}$.

Also, $f \in S_j$ by the induction hypothesis, and since S_j is closed under multiplication by x, $xf \in S_j$ as well. But clearly $xf > f$, and so by Lemma 5, $\operatorname{Rank} g_{i+1,j} = \operatorname{Rank} xg_{i,j}$, which means (since $xf \in S_j$), that $g_{i+1,j} \in S_j$ as well.

It remains only to prove that $g_{i+1,j}$ is minimal in $\overline{K}_{i+1} \cap S_j$. If it is not minimal, there exists a polynomial $h \in \overline{K}_{i+1} \cap S_j$ such that $h < g_{i+1,j}$. Also, since $h \in \overline{K}_i \cap S_j$, $f \leq h$. But $\operatorname{Rank} g_{i+1,j} = \operatorname{Rank} xf$, and since there can be no polynomial $f' \in S_j$ with $\operatorname{Rank} f < \operatorname{Rank} f' < \operatorname{Rank} xf$, it follows that $h \equiv f$. By a suitable normalization, we can arrange to have the leading coefficient of h equal to that of f. Now consider the polynomial $f' = h - f$. Clearly $f' \in \overline{K}_i$, and $f' < h \equiv f$. Also, $D_{i+1}(f') \neq 0$ since $D_{i+1}(h) = 0$ (since $h \in K_{i+1}$) and $D_{i+1}(f) \neq 0$ (since $j \in J_1$).

In summary, if $g_{i+1,j}$ is not minimal in $\overline{K}_{i+1} \cap S_j$, we can construct a nonzero polynomial f' such that

$$\left.\begin{array}{l} f' \in \overline{K}_i \setminus \overline{K}_{i+1} \\ \\ f' < f \end{array}\right\} \tag{55}$$

But this contradicts Eq. (52), which says, in effect, that f is a minimal element of $\overline{K}_i \setminus \overline{K}_{i+1}$. ❒

[13] Cf. the remarks at the beginning of this subsection.

D. Pseudocode for Kötter's Algorithm

```
/* Kötter's Algorithm -- General Formulation */
/* Complexity O(C^2) */

BEGIN (Given L, (D_i)_{i=1}^C, arbitrary monomial order)
1.  FOR j = 0 to L
2.      g_j := y^j
3.  FOR i = 1 to C DO
4.      FOR j = 0 to L DO
5.          Δ_j := D_i(g_j)            /* jth discrepancy */
6.      j := {j : Δ_j ≠ 0}
7.      IF j ≠ ∅
8.          j* := argmin {g_j : Δ; ≠ 0}      /* "min" wrt monomial order  */
9.          f := g_{j*};  Δ := Δ_{j*}
10.         FOR j ∈ J DO
11.             IF (j ≠ J*)
12.                 g_j := Δ g_j − Δ_j f   /* No change in Rank g_j */
13.             ELSE IF (j = J*)
14.                 g_j := Δ(xf) − D_i(xf) f   /* Rank g_j increases by min */
15. Q_0(x, y) := min_{j=0}^L {g_j(x, y)}
END
```

Theorem 10. *At the end of Kötter's algorithm,*

$$g_j(x,y) = \min\{g \in \overline{K}_C \cap S_j\}$$

$$Q_0(x,y) = \min\{g \in \overline{K}_C\}$$

where the "min" is with respect to the given monomial order.

```
/* Kötter's Interpolation Algorithm -- Special Case for GS Decoding */
/* Complexity O(n^2 m^4) if m_i = m for all i */

BEGIN (Given L, (α_i, β_i)_{i=1}^n, (m_i)_{i=1}^n, (1, k-1) wdeg monomial order)
1.  FOR j = 0 to L
        g_j := y^j
2.  FOR i = 1 to n DO
3.      FOR (r, s) = (0, 0) to (m_i - 1, 0) DO  /* by (m_i - 1, 1) lex order */
4.          FOR j = 0 to L DO
5.              Δ_j := D_{r,s} g_j(α_i, β_i)          /* jth discrepancy */
6.          J := {j : Δ_j ≠ 0}
7.          IF J ≠ ∅
8.              j* := argmin{g_j : j ∈ J}
9.              f := g_{j*}; Δ := Δ_{j*}
10.             FOR j ∈ J DO
11.                 IF (j ≠ j*)
12.                     g_j := Δ g_j - Δ_j f      /* No change in wdeg */
13.                 ELSE IF (j = J*)
14.                     g_j := Δ(x - α_i) f       /* wdeg increases by 1 */
15. Q_0(x, y) := min_j{g_j(x, y)}              /* The Interpolation Polynomial */
END
```

Derivation of Line 14:

$$
\begin{aligned}
g_j &= \Delta(xf) - D_i(xf)f \\
&= D_{r,s}f(\alpha_i,\beta_i)\cdot\big(xf(x,y)\big) - D_{r,s}\big(xf(x,y)\big)\Big|_{\substack{x=\alpha_i\\ y=\beta_i}}\cdot f(x,y) \\
&= D_{r,s}f(\alpha_i,\beta_i)\cdot\big(xf(x,y)\big) - \alpha_i D_{r,s}f(\alpha_i,\beta_i)\cdot f(x,y) \quad \text{(Corollary 8)} \\
&= D_{r,s}f(\alpha_i,\beta_i)f(x,y)\,(x-\alpha_i) \\
&= K(x-\alpha_i)f
\end{aligned}
$$

Theorem 11. *When Kötter's Interpolation Algorithm terminates,*

$$
\begin{aligned}
g_j(x,y) &= \min\{f \in I(m) \cap F_L[x,y] \cap S_j\} \\
Q_0(x,y) &= \min\{f \in I(m) \cap F_L[x,y]\}
\end{aligned}
$$

where

$$I(m) = \{Q(x,y) : \mathrm{ord}\,(Q : \alpha_i, \beta_i) = m_i, i = 1, \cdots, n\}$$

```
/* Kötter's Interpolation Algorithm -- Special Case m = 1 */
/* Complexity O(n^2) */

BEGIN (Given L, (α_i, β_i)_{i=1}^n, (1, k-1) wdeg to order polynomials)
1.  FOR j = 0 to L
        g_j := y^j          /* If re-encoding trick is not used */
2.  FOR i = 1 to n DO
3.      FOR j = 0 to L DO
4.          Δ_j := g_j(α_i, β_i)          /* jth discrepancy */
5.      J := {j : Δ_j ≠ 0}
6.      IF j ≠ ∅
7.          J* := argmin{g_j : j ∈ J}
8.          f := g_{j*}; Δ := Δ_{j*}
9.          FOR j ∈ J DO
10.             IF (j ≠ j*)
11.                 g_j := Δ g_j − Δ_j f        /* No change in wdeg */
12.             ELSE IF (j = J*)
13.                 g_j := (x − α_i) f      /* wdeg increases by 1 */
14. Q_0(x,y) := min_j{g_j(x,y)}              /* The Interpolation Polynomial */
END
```

VIII. An Alternative to Kötter's Solution: The Feng–Tzeng Algorithm

In this section, we will describe the Feng–Tzeng algorithm, which inspired Kötter, and which standing alone can provide a practical ($O(n^3)$) solution to the interpolation problem.

Let $A = (a_{i,j})$ be an $m \times n$ matrix over F with $m < n$. Let $\mathbf{a}_i = (a_{i,1}, \cdots, a_{i,n})$ denote the ith row of A, and let $\mathbf{a}^{(j)} = (a_{1,j}, \cdots, a_{m,j})$ denote the jth column. The FT algorithm [8] finds the largest integer L such that the first L columns of A are linearly independent. It also produces an $L+1$-vector $\mathbf{d} = (d_1, \cdots, d_L, 1)$ such that $d_1\mathbf{a}^{(1)} + \cdots + d_L\mathbf{a}^{(L)} + \mathbf{a}^{(L+1)} = 0$.

```
/* The Feng-Tzeng Algorithm */
/* Complexity O(n^3) */
BEGIN
FOR u = 1 to n DO
        ρ(u) = 0; δ(u) = 0;
s := 0
DO
1.      s := s + 1; r := 0; b := 0^(s-1)1; columnblocked := false
/* any b of the form b_1 b_2 ··· b_{s-1} 1 will do.*/
        DO
2.          r := r + 1
3.          Δ := a_r · b
4.          IF (Δ ≠ 0)
5.                  IF (there is a u < s for which ρ(u) = r)
6.                      b := b − (Δ/δ(u)) c_u /* now a_i · b = 0 for all 1 ≤ i ≤ r.*/
7.                  ELSE (there is no u < s for which ρ(u) = r)
8.                      ρ(s) := r; δ(s) := Δ; c_s := b
                        columnblocked := true
        WHILE (r < m and columnblocked = false)
WHILE (columnblocked = true)
9.  c_s := b; L := s − 1
END
```

The following theorem describes the most important properties of the FT algorithm.

Theorem 12. *First,*

$$\rho(1), \cdots, \rho(L) \quad \textit{are distinct elements of} \quad \{1, \cdots, m\} \tag{56}$$

Next, for $s = 1, \cdots, L$,

$$\mathbf{a}_r \cdot \mathbf{c}_s = \begin{cases} 0 & \text{for } 1 \le r < \rho(s) \\ \delta(s) \ne 0 & \text{for } r = \rho(s) \end{cases} \tag{57}$$

Finally,

$$\mathbf{a}_r \cdot \mathbf{c}_{L+1} = 0 \quad \text{for } 1 \le r \le m \tag{58}$$

Proof. See [8]. ❒

Corollary 10. The first L columns of A are linearly independent. The first $L+1$ columns of A are linearly dependent. In fact, if $\mathbf{c}_{L+1} = (d_1, \cdots, d_L, 1)$, then

$$d_1\mathbf{a}^{(1)} + \cdots + d_L\mathbf{a}^{(L)} + \mathbf{a}^{(L+1)} = 0 \tag{59}$$

Proof. Denote by A_{L+1} the $m \times (L+1)$ matrix formed by the first $L+1$ columns of A, i.e.,

$$A_{L+1} = \begin{pmatrix} \mathbf{a}^{(1)} & \cdots & \mathbf{a}^{(L+1)} \end{pmatrix}$$

and by C the $(L+1) \times (L+1)$ matrix whose rows are the vectors $\mathbf{c}_1, \cdots, \mathbf{c}_L, \mathbf{c}_{L+1}$, i.e.,

$$C = \begin{pmatrix} \mathbf{c}_1 \\ \vdots \\ \mathbf{c}_L \\ \mathbf{c}_{L+1} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ * & 1 & 0 & \vdots \\ \vdots & \vdots & \vdots & 0 \\ * & * & * & 1 \end{pmatrix}$$

Now consider the $m \times (L+1)$ matrix $D = A_{L+1}C^T$. Because C has the lower triangular form shown, the first s columns of D are column-equivalent to the first s columns of A_{L+1}, for $s = 1, \cdots, L+1$. But the (r,s)th entry in D is $\mathbf{a}_r \cdot \mathbf{c}_s$, and so by Eq. (57), the sth column of D has $\rho(s) - 1$ leading 0's followed by the nonzero entry $\delta(s)$, for $s = 1, \cdots, L$. But by Eq. (56), the indices $\rho(1), \cdots, \rho(L)$ are distinct, which implies that the first L columns of D, and hence also those of A, are linearly independent. Finally, Eq. (58) says that the $L+1$st column of D is identically zero, which implies the first $L+1$ columns of D, and hence also of A, are linearly dependent. Equation (58) shows that Eq. (59) is true. ❒

Example 12. (Taken from [8]). Let A be the following 6×6 matrix over $GF(2)$:

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

The result of running the FT algorithm is

$$C = \begin{array}{c} \\ \mathbf{c}_1 \\ \mathbf{c}_2 \\ \mathbf{c}_3 \\ \mathbf{c}_4 \\ \mathbf{c}_5 \end{array} \begin{array}{c} \begin{array}{ccccc} & & & & \end{array} \\ \begin{pmatrix} 1 & & & & \\ 0 & 1 & & & \\ 0 & 0 & 1 & & \\ 0 & 1 & 1 & 1 & \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \end{array}$$

$$D = A_5 C^T = \begin{array}{c} \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \\ \mathbf{a}_4 \\ \mathbf{a}_5 \end{array} \begin{array}{c} \begin{array}{ccccc} \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 & \mathbf{c}_4 & \mathbf{c}_5 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

The first four columns of D show that $\rho(1) = 2$, $\rho(2) = 4$, $\rho(3) = 1$, and $\rho(4) = 5$. Thus, $L = 4$ and the leftmost linear dependence among the columns of A is

$$\mathbf{a}^{(2)} + \mathbf{a}^{(3)} + \mathbf{a}^{(4)} + \mathbf{a}^{(5)} = 0$$

Example 13. Let $F = GF(8)$, with primitive root γ satisfying $\gamma^3 = \gamma + 1$. Let's use the FT algorithm to construct a two-variable polynomial $Q(x, y)$ of minimal $(1, 3)$-revlex rank such that $\text{ord}\,(Q : 1, \gamma) \geq 1$, and $\text{ord}\,(Q : \gamma, \gamma^6) \geq 2$. There are four constraints: $Q_{0,0}(1, \gamma) = Q_{0,0}(\gamma, \gamma^6) = Q_{0,1}(\gamma, \gamma^6) = Q_{1,0}(\gamma, \gamma^6) = 0$, so we know that some linear combination of the first five monomials listed in $(1, 3)$-revlex order, viz., $1, x, x^2, x^3, y$, will suffice. Thus, for the matrix A we will take the 4×5 matrix whose columns are indexed by the first five monomials $\{1, x, x^2, x^3, y\}$ in $(1, 3)$-revlex order, and whose rows are indexed by the four coefficient constraints corresponding to the conditions (1) $Q_{0,0}(1, \gamma) = 0$; (2) $Q_{0,0}(\gamma, \gamma^6) = 0$; (3) $Q_{0,1}(\gamma, \gamma^6) = 0$; and (4) $Q_{1,0}(\gamma, \gamma^6) = 0$. Here is a summary of the work:

$$A = \begin{array}{c} \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \\ \mathbf{a}_4 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & x & x^2 & x^3 & y \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 1 & \gamma \\ 1 & \gamma & \gamma^2 & \gamma^3 & \gamma^6 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & \gamma^2 & 0 \end{pmatrix} \end{array}$$

$$C = \begin{array}{c} \\ \mathbf{c}_1 \\ \mathbf{c}_2 \\ \mathbf{c}_3 \\ \mathbf{c}_4 \end{array} \begin{array}{c} \\ \begin{pmatrix} 1 & & & \\ 1 & 1 & & \\ \gamma & \gamma^3 & 1 & \\ \gamma^2 & \gamma^2 & 1 & 1 \end{pmatrix} \end{array}$$

$$D = A_4 C^T = \begin{array}{c} \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \\ \mathbf{a}_4 \end{array} \begin{array}{c} \begin{array}{cccc} \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 & \mathbf{c}_4 \end{array} \\ \begin{pmatrix} \underline{1} & 0 & 0 & 0 \\ 1 & \underline{\gamma^3} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & \underline{\gamma^3} & 0 \end{pmatrix} \end{array}$$

The first three columns of D tell us that $\rho(1) = 1$, $\delta(1) = 1$; $\rho(2) = 2$, $\delta(2) = \gamma^3$; and $\rho(3) = 4$, $\delta(3) = \gamma^3$. Thus, the algorithm returns $L = 3$, and $\mathbf{c}_4 = (\gamma^2, \gamma^2, 1, 1)$, i.e., $Q(x, y) = \gamma^2 + \gamma^2 x + x^2 + x^3$ is the unique (up to scalar multiplication) polynomial of minimal $(1, 3)$-revlex rank satisfying the given conditions.

Example 14. [16, Example 9.3]. Let us use the FT algorithm to find the "linear complexity" of the sequence $(S_1, S_2, \cdots, S_8) = (1, 1, 1, 0, 1, 0, 1, 0)$. The appropriate matrix A is as shown below. It has the general form

$$A = \begin{pmatrix} S_1 & S_2 & S_3 & \cdots & S_7 & S_8 \\ S_2 & S_3 & S_4 & \cdots & S_8 & \\ \vdots & & & & & \\ S_7 & S_8 & & & & \\ S_8 & & & & & \end{pmatrix}$$

(The missing entries are "don't cares" that do not enter into the calculations.) According to [8], the algorithm stops when $s + r \geq n + 1$, i.e., when otherwise the next entry of A to be processed is a don't care. The final vector $\mathbf{c}_5 = (0, 0, 1, 0, 1)$ indicates that the "shortest" linear recurrence relation satisfied by the given sequence is

$$S_j = S_{j-2} \quad \text{for } j \geq 5$$

(This example is somewhat pathological, since in general an 8-term sequence $(S_1, S_2, \cdots, S_8)$ will satisfy a degree-4 recurrence relation of the form

$$S_j = \sigma_1 S_{j-1} + \sigma_2 S_{j-2} + \sigma_3 S_{j-3} + \sigma_4 S_{j-4} \quad \text{for } j \geq 5$$

It appears to be an accident that $\sigma_3 = \sigma_4 = 0$ in this case.)

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{array} \begin{array}{c} \begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & \\ 1 & 0 & 1 & 0 & 1 & 0 & & \\ 0 & 1 & 0 & 1 & 0 & & & \\ 1 & 0 & 1 & 0 & & & & \\ 0 & 1 & 0 & & & & & \\ 1 & 0 & & & & & & \\ 0 & & & & & & & \end{pmatrix} \end{array}$$

$$C = \begin{array}{c} \\ \mathbf{c}_1 \\ \mathbf{c}_2 \\ \mathbf{c}_3 \\ \mathbf{c}_4 \\ \mathbf{c}_5 \end{array} \begin{array}{c} \\ \begin{pmatrix} 1 & & & & \\ 1 & 1 & & & \\ 1 & 0 & 1 & & \\ 1 & 0 & 1 & 1 & \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix} \end{array}$$

$$D = A_4 C^T = \begin{array}{c} \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \\ \mathbf{a}_4 \\ \mathbf{a}_5 \end{array} \begin{array}{c} \begin{array}{ccccc} \mathbf{c}_1 & \mathbf{c}_2 & \mathbf{c}_3 & \mathbf{c}_4 & \mathbf{c}_5 \end{array} \\ \begin{pmatrix} \underline{1} & 0 & 0 & 0 & 0 \\ 1 & 0 & \underline{1} & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

IX. The Roth–Ruckenstein Solution to the Factorization Problem

In this section, we will present the most efficient algorithm currently known for solving the factorization problem, due to Roth and Ruckenstein [21]. Our exposition includes an improved stopping rule and a "depth first search" implementation.

The factorization problem is this: given a polynomial $Q(x, y) \in F[x, y]$, find all polynomials $f(x)$ of degree $\leq v$ such that $(y - f(x)) \,|\, Q(x, y)$. Alternatively, by Lemma 3, find all $f(x) \in F_v[x]$ such that

$$Q\big(x, f(x)\big) \equiv 0 \tag{60}$$

If Eq. (60) holds, we call $f(x)$ a y-root of $Q(x, y)$. In this section, we will describe an algorithm due to Roth and Ruckenstein [21] for finding y-roots.

If $Q(x,y)$ is a two-variable polynomial such that $x^m \,|\, Q(x,y)$, but $x^{m+1} \not| Q(x,y)$, define

$$\langle\langle Q(x,y)\rangle\rangle = \frac{Q(x,y)}{x^m}$$

Although $Q(0,y)$ might be identically zero, nevertheless $\langle\langle Q(0,y)\rangle\rangle$ is a nonzero polynomial in y (e.g., if $Q(x,y) = xy$, $Q(0,y) = 0$ but $\langle\langle Q(0,y)\rangle\rangle = y$).

Suppose

$$f(x) = a_0 + a_1 x + \cdots + a_v x^v \tag{61}$$

is a y-root of $Q(x,y)$. We will see that the coefficients $a_0, a_1, \cdots, a_v$ can be "picked off," one at a time. As a start, the following lemma shows how to determine a_0.

Lemma 6. If $\big(y - f(x)\big) \,|\, Q(x,y)$, then $y = f(0) = a_0$ is a root of the equation

$$Q_0(0,y) = 0$$

where $Q_0(x,y) = \langle\langle Q(x,y)\rangle\rangle$.

Proof. By definition, $Q(x,y) = x^m Q_0(x,y)$ for some $m \geq 0$. Thus, if $(y - f(x)) \,|\, Q(x,y)$, then $(y - f(x)) \,|\, Q_0(x,y)$ as well, so that $Q_0(x,y) = \big(y - f(x)\big)T_0(x,y)$ for some polynomial $T_0(x,y)$. Thus, $y = f(0)$ is a solution of the equation $Q_0(0,y) = 0$. ❒

We now proceed by induction, defining three sequences of polynomials, $f_j(x)$, $T_j(x,y)$, and $Q_j(x,y)$, for $j = 0, 1, \cdots, v$, as follows.

Initially, $f_0 := f(x)$, $Q_0(x,y) := \langle\langle Q(x,y)\rangle\rangle$. For $j \geq 1$, define

$$f_j(x) := \big(f_{j-1}(x) - f_{j-1}(0)\big)/x = a_j + \cdots + a_v x^{v-j} \tag{62}$$

$$T_j(x,y) := Q_{j-1}(x, xy + a_{j-1}) \tag{63}$$

$$Q_j(x,y) := \langle\langle T_j(x,y)\rangle\rangle \tag{64}$$

Theorem 13. *Given $f(x) = a_0 + a_1 x + \cdots + a_v x^v \in F_v[x]$, and $Q(x,y) \in F[x,y]$, define the sequences $f_j(x)$ and $Q_j(x,y)$ as in Eqs. (62) and (64). Then for any $j \geq 1$, $(y - f(x)) \,|\, Q(x,y)$ if and only if $(y - f_j(x)) \,|\, Q_j(x,y)$.*

Proof. We will show that if $j \geq 1$, $\big(y - f_j(x)\big) \,|\, Q_j(x,y) \leftrightarrow \big(y - f_{j-1}(x)\big) \,|\, Q_{j-1}(x,y)$.

1. ($\rightarrow$). Assuming $\big(y - f_j(x)\big) \,|\, Q_j(x,y)$, by Eq. (64), $T_j(x,y) = x^m Q_j(x,y)$ for some $m \geq 0$. Then

$$\big(y - f_j(x)\big) \,|\, x^m Q_j(x,y) = T_j(x,y) = Q_{j-1}(x, xy + a_{j-1})$$

Therefore,

$$Q_{j-1}(x, xy + a_{j-1}) = \big(y - f_j(x)\big)U(x, y) \tag{65}$$

for some $U(x, y) \in F[x, y]$. Now substitute $(y - a_{j-1})/x$ for y in Eq. (65):

$$Q_{j-1}(x, y) = \left(\frac{y - a_{j-1}}{x} - f_j(x)\right) U\left(x, \frac{y - a_{j-1}}{x}\right) \tag{66}$$

Multiplying both sides of Eq. (66) by a sufficiently large power of x, we obtain

$$x^L Q_{j-1}(x, y) = \big(y - f_{j-1}(x)\big)V(x, y)$$

for some $V(x, y) \in F[x, y]$. Thus, $y - f_{j-1}(x)$ divides $x^L Q_{j-1}(x, y)$; but x^L and $y - f_{j-1}(x)$ are relatively prime, so $y - f_{j-1}(x)$ divides $Q_{j-1}(x, y)$, as asserted.

2. (←). Assuming $\big(y - f_{j-1}(x)\big) \,|\, Q_{j-1}(x, y)$,

$$Q_{j-1}(x, y) = \big(y - f_{j-1}(x)\big)U(x, y)$$

for some polynomial $U(x, y)$. Thus, by Eq. (63),

$$\begin{aligned} T_j(x, y) &= \big(xy + a_{j-1} - f_{j-1}(x)\big) \cdot U(x, xy + a_{j-1}) \\ &= x\big(y - f_j(x)\big) \cdot U(x, xy + a_{j-1}) \end{aligned}$$

which proves that $\big(y - f_j(x)\big) \,|\, T_j(x, y)$. But since $T_j(x, y) = x^m Q_j(x, y)$, it follows that $\big(y - f_j(x)\big) | Q_j(x, y)$ as well. ❒

Here is the "picking off" theorem.

Corollary 11. [21, Lemma 5.1]. If $\big(y - f(x)\big) \,|\, Q(x, y)$, then $y = a_j$ is a root of the equation

$$Q_j(0, y) = 0, \quad \text{for } j = 0, \cdots, v$$

Proof. By Theorem 13, $y - f_j(x)$ divides $Q_j(x, y)$ for all $j \geq 0$. Substituting $x = 0$ yields the stated result, since $f_j(0) = a_j$. ❒

Corollary 12. If $y \,|\, Q_{v+1}(x, y)$, i.e., if $Q_{v+1}(x, 0) = 0$, then $f(x) = a_0 + \cdots + a_v x^v$ is a y-root of $Q(x, y)$.

Proof. Note that, by Eq. (62), $f_j(x) = 0$ for all $j \geq v + 1$, so that the hypothesis $y \,|\, Q_{v+1}(x, y)$ says that $\big(y - f_{v+1}(x)\big) \,|\, Q_{v+1}(x, y)$. Now apply Theorem 13. ❒

The following lemma provides some insight into the all-important transformation $Q(x, y) \rightarrow Q(x, xy + a)$.

Lemma 7. If

$$Q(x,y) = \sum_i x^i g_i(y)$$

$$= \sum_{i,j} x^i y^j D_j g_i(0)$$

then

$$Q(x, xy + a) = \sum_{i,j} x^i y^j D_j g_{i-j}(a)$$

where D_i denotes the ith one-dimensional Hasse derivative.

Proof. By Theorem 4, we have

$$g_s(z + a) = \sum_r z^r D_r g_s(a)$$

and so

$$Q(x, z + a) = \sum_s x^s \sum_r z^r D_r g_s(a) \tag{67}$$

Substituting xy for z in Eq. (67), we have

$$Q(x, xy + a) = \sum_{s,r} x^{s+r} y^r D_r g_s(a)$$

$$= \sum_{i,j} x^i y^j D_j g_{i-j}(a)$$

❒

Symbolically, Lemma 7 can be summarized as follows:

$$Q(x,y) = \begin{pmatrix} g_0(0) & g_1(0) & g_2(0) & g_3(0) & \cdots \\ D_1 g_0(0) & D_1 g_1(0) & D_1 g_2(0) & D_1 g_3(0) & \\ D_2 g_0(0) & D_2 g_1(0) & D_2 g_2(0) & D_2 g_3(0) & \\ \vdots & & & & \end{pmatrix}$$

$$Q(x, xy + a) = \begin{pmatrix} g_0(a) & g_1(a) & g_2(a) & g_3(a) & \cdots \\ 0 & D_1 g_0(a) & D_1 g_1(a) & D_1 g_2(a) & \\ 0 & 0 & D_2 g_0(a) & D_2 g_1(a) & \\ 0 & 0 & 0 & D_3 g_0(a) & \\ \vdots & & & & \end{pmatrix}$$

In words: if the entries of column j of $Q(x,y)$ are interpreted as the coefficients of a polynomial, say $g_j(z)$, then the entries of the jth diagonal of $Q(x, xy+a)$ are the coefficients of the polynomial $g_j(z+a)$.

We now give a pseudocode representation of the RR algorithm. It takes as input a bivariate polynomial $Q(x,y)$ and positive integer D, and returns as output the set of all y-roots of $Q(x,y)$ of degree $\leq D$. The strategy adopted by the algorithm is "depth-first search," as described, for example, in [6, Section 23.3].

```
/* The Roth-Ruckenstein Algorithm */
/* Input: {Q(x,y), D}; Output: {f(x) : (y − f(x)) | Q(x,y); deg f(x) ≤ D}. */
```

```
    BEGIN
1.  π[0] = NIL; deg[0] = −1;    Q_0(x,y) = Q(x,y);
        t = 1; u = 0
2.  DFS[u]
    END

    /* DFS[u]:  Depth-first search beginning at u */
    BEGIN
3.  IF (Q_u(x,0) = 0)
4.      Output f_[u](x)   /* using traceback  */
5.  ELSE IF (deg[u] < D)   /* explore edges from vertex u */
6.      R = RootList[Q_u(0,y)]
7.          FOR (α ∈ R) DO
8.              v = t; t = t + 1;
9.              π[v] = u; deg[v] = deg[u] + 1; Coeff[v] = α;
10.             Q_v(x,y) = ⟨⟨Q_u(x, xy + α)⟩⟩
11.             DFS[v]
    END
```

Glossary:

$\pi[u]$ = Parent of u

$\deg[u]$ = "degree" of u = distance from root - 1

$\text{Coeff}[u]$ = polynomial coefficient at u

$f_{[u]}(x)$ = "partial" polynomial at u :

$= \text{Coeff}[u]x^{\deg[u]} + \text{Coeff}[\pi[u]]x^{\deg[\pi[u]]} + \cdots$

$Q_u(x,y)$: See Eq. (64)

Example 15. [21, Example 7.1]. Let $F = GF(19)$ and let us use the RR algorithm to find the y-roots of the degree ≤ 1 of the polynomial $Q(x,y)$ given below.

$$Q(x,y) = (4 + 12x + 5x^2 + 11x^3 + 8x^4 + 13x^5)$$
$$+ (14 + 14x + 9x^2 + 16x^3 + 8x^4)y$$
$$+ (14 + 13x + x^2)y^2$$
$$+ (2 + 11x + x^2)y^3$$
$$+ 17y^4$$

In general, the RR algorithm navigates its way through a tree structure, with the potential y-roots forming paths from the root (vertex 0). This particular example is summarized in Table 1 and Fig. 1. Each vertex is labeled with a "timestamp," which indicates the order in which the vertices are visited. The edge labels descending from vertex $[u]$ correspond to the roots of the equation $Q_u(0, y) = 0$, i.e., `RootList`$[u]$. Similarly, the label on the edge going up from u to $\pi[u]$ is Coeff $[u]$. Finally, the termination symbols indicate terminal vertices, i.e., either vertices corresponding to y-roots of $Q(x, y)$ (unshaded icon) or vertices whose depth exceeds v (shaded icon).

Table 1. Summary of Example 15.

u	$\pi[u]$	$\deg[u]$	Coeff $[u]$	`RootList`$[u]$
0	`NIL`	-1	`NIL`	$\{18, 18, 14, 8\}$
1	0	0	18	$\{14, 15\}$
2	1	1	14	–
3	1	1	15	–
4	0	0	14	$\{16\}$
5	4	1	16	–
6	0	0	8	$\{8\}$
7	6	1	8	–

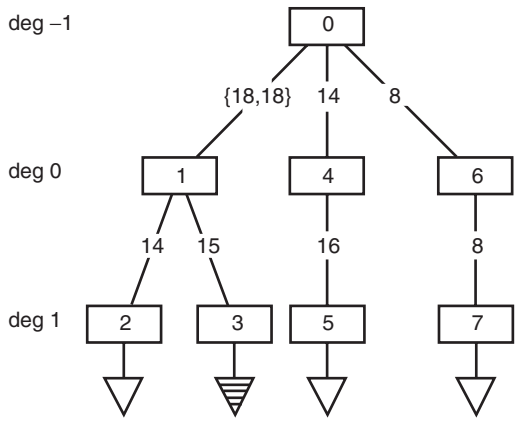

Fig. 1. The RR tree for Example 15.

• We begin with Vertex 0 ($\pi[0] = \texttt{NIL}$; $\deg[0] = -1$):

$$Q_0(x,y) = \langle\langle Q(x,y) \rangle\rangle$$

$$= \begin{array}{c} \\ 1 \\ y \\ y^2 \\ y^3 \\ y^4 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & x & x^2 & x^3 & x^4 & x^5 \end{array} \\ \begin{pmatrix} 4 & 12 & 5 & 11 & 8 & 13 \\ 14 & 14 & 9 & 16 & 8 & \\ 14 & 13 & 1 & & & \\ 2 & 11 & 1 & & & \\ 17 & & & & & \end{pmatrix} \end{array}$$

$$Q_0(x,0) = 4 + 12x + \cdots \neq 0; \deg[0] < 1;$$

$$Q_0(0,y) = 4 + 14y + 14y^2 + 2y^3 + 17y^4$$

$$= 17(y-18)^2(y-14)(y-8)$$

$$\texttt{RootList}[0] = \{18, 14, 8\}$$

• Vertex 1 ($\pi[1] = 0$; $\deg[1] = 0$; Coeff $[1] = 18$):

$$Q_1(x,y) = \langle\langle Q_0(x, xy+18) \rangle\rangle$$

$$= \begin{array}{c} \\ 1 \\ y \\ y^2 \\ y^3 \\ y^4 \end{array} \begin{array}{c} \begin{array}{cccc} 1 & x & x^2 & x^3 \end{array} \\ \begin{pmatrix} 15 & 14 & 0 & 13 \\ 2 & 10 & 16 & 8 \\ 15 & 18 & 17 & \\ & 10 & 11 & 1 \\ & & 17 & \end{pmatrix} \end{array}$$

$$Q_1(x,0) = 15 + 14x + \cdots \neq 0, \deg[1] < 1$$

$$Q_1(0,y) = 15 + 2y + 15y^2$$

$$= 15(y-14)(y-15) = 0,$$

$$\texttt{Rootlist}[1] = \{14, 15\}$$

- Vertex 2 ($\pi[2] = 1$; $\deg[2] = 1$; Coeff$[2] = 14$):

$$Q_2(x,y) = \langle\langle Q_1(x, xy+14)\rangle\rangle$$

$$= \begin{array}{c} \\ 1 \\ y \\ y^2 \\ y^3 \\ y^4 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & x & x^2 & x^3 & x^4 & x^5 \end{array} \\ \begin{pmatrix} & & & & & \\ 4 & 10 & 18 & 7 & & \\ & 15 & 1 & 8 & 4 & \\ & & & 10 & 13 & 1 \\ & & & & & 17 \end{pmatrix} \end{array}$$

$$Q_2(x,0) = 0;$$

$$\text{Output } f_{[2]}(x) = 14x + 18$$

- Vertex 3 ($\pi[3] = 0$; $\deg[3] = 1$; Coeff$[3] = 15$):

$$Q_3(x,y) = \langle\langle Q_1(x, xy+15)\rangle\rangle$$

$$= \begin{array}{c} \\ 1 \\ y \\ y^2 \\ y^3 \\ y^4 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & x & x^2 & x^3 & x^4 & x^5 \end{array} \\ \begin{pmatrix} 2 & 18 & 12 & 0 & & \\ 15 & 4 & 8 & 18 & & \\ & 15 & 12 & 16 & 7 & \\ & & & 10 & 5 & 1 \\ & & & & & 17 \end{pmatrix} \end{array}$$

$$Q_3(x,0) = 2 + 18x + \cdots \neq 0; \deg[3] \not< 1;$$

(No Output)

- Vertex 4 ($\pi[4] = 1$; $\deg[4] = 0$; Coeff$[4] = 14$):

$$Q_4(x,y) = \langle\langle Q_0(x, xy+14)\rangle\rangle$$

$$= \begin{array}{c} \\ 1 \\ y \\ y^2 \\ y^3 \\ y^4 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & x & x^2 & x^3 & x^4 \end{array} \\ \begin{pmatrix} 13 & 12 & 7 & 6 & 13 \\ 17 & 6 & 17 & 16 & 8 \\ & 7 & 0 & 5 & 0 \\ & & 4 & 11 & 1 \\ & & & 17 & \end{pmatrix} \end{array}$$

$$Q_4(x,0) = 13 + 12x + \cdots \neq 0; \deg[4] < 1;$$

$$Q_4(0,y) = 13 + 17y$$

$$\texttt{Rootlist}[4] = \{16\}$$

- Vertex 5 ($\pi[5] = 4$; $\deg[5] = 1$; Coeff $[5] = 16$):

$$
\begin{aligned}
Q_5(x,y) = {} & \langle\langle Q_4(x, xy+16)\rangle\rangle \\
& \begin{array}{c} \\ 1 \\ y \\ y^2 \\ y^3 \\ y^4 \end{array}
\begin{array}{c}
\begin{array}{ccccccc} 1 & x & x^2 & x^3 & x^4 & x^5 & x^6 \end{array} \\
\begin{pmatrix}
 & & & & & & \\
17 & 2 & 11 & 5 & 16 & & \\
 & & 7 & 2 & 7 & 10 & \\
 & & & & 4 & 16 & 1 \\
 & & & & & & 17
\end{pmatrix}
\end{array} \\
Q_5(x,0) = {} & 0 \\
\text{Output } f_{[5]}(x) = {} & 16x + 14
\end{aligned}
$$

- Vertex 6 ($\pi[6] = 0$; $\deg[6] = 0$; Coeff $[6] = 8$):

$$
\begin{aligned}
Q_6(x,y) = {} & \langle\langle Q_0(x, xy+8)\rangle\rangle \\
& \begin{array}{c} \\ 1 \\ y \\ y^2 \\ y^3 \\ y^4 \end{array}
\begin{array}{c}
\begin{array}{ccccc} 1 & x & x^2 & x^3 & x^4 \end{array} \\
\begin{pmatrix}
14 & 7 & 6 & 15 & 13 \\
3 & 16 & 8 & 16 & 8 \\
 & 16 & 11 & 6 & 0 \\
 & & 14 & 11 & 1 \\
 & & & 17 &
\end{pmatrix}
\end{array} \\
Q_6(x,0) = {} & 14 + 7x + \cdots \neq 0; \deg[6] < 1; \\
Q_6(0,y) = {} & 14 + 3y \\
\texttt{RootList}[6] = {} & \{8\}
\end{aligned}
$$

- Vertex 7 ($\pi[7] = 6$; $\deg[7] = 1$; Coeff $[7] = 8$):

$$
\begin{aligned}
Q_7(x,y) = {} & \langle\langle Q_6(x, xy+8)\rangle\rangle \\
& \begin{array}{c} \\ 1 \\ y \\ y^2 \\ y^3 \\ y^4 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 1 & x & x^2 & x^3 & x^4 & x^5 \end{array} \\
\begin{pmatrix}
 & & & & & \\
3 & 6 & 3 & 9 & 10 & \\
 & & 16 & 5 & 15 & 5 \\
 & & & 14 & 4 & 1 \\
 & & & & & 17
\end{pmatrix}
\end{array} \\
Q_7(x,0) = {} & 0 \\
\text{Output } f_{[7]}(x) = {} & 8x + 8
\end{aligned}
$$

Thus, the output of the RR algorithm in this case is

$$\{14x + 18, 16x + 14, 8x + 8\}$$

References

[1] S. Ar, R. Lipton, R. Rubinfeld, and M. Sudan, "Reconstructing Algebraic Functions from Mixed Data," *SIAM J. Computation,* vol. 28, no. 2, pp. 488–511, 1999.

[2] E. R. Berlekamp, *Algebraic Coding Theory,* New York: McGraw-Hill, 1968.

[3] E. R. Berlekamp and J. L. Ramsey, "Readable Erasures Improve the Performance of Reed–Solomon Codes," *IEEE Trans. Inform. Theory,* vol. 24, no. 5, pp. 632–633, September 1978.

[4] R. E. Blahut, *Theory and Practice of Error-Control Codes,* Reading, Massachusetts: Addison-Wesley, 1983.

[5] K.-M. Cheung, "More on the Decoder Error Probability for Reed–Solomon Codes," *IEEE Trans. Inform. Theory,* vol. 35, no. 4, pp. 895–900, July 1989.

[6] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms,* Cambridge, Massachusetts: MIT Press, 1990.

[7] D. Cox, J. Little, and D. O'Shea, *Ideals, Varieties, and Algorithms,* New York: Springer-Verlag, 1992.

[8] G.-L. Feng and K. K. Tzeng, "A Generalization of the Berlekamp–Massey Algorithm for Multisequence Shift-Register Synthesis with Applications to Decoding Cyclic Codes," *IEEE Trans. Inform. Theory,* vol. 37, no. 5, pp. 1274–1287, September 1991.

[9] V. Guruswami and M. Sudan, "Improved Decoding of Reed–Solomon Codes and Algebraic Geometry Codes," *IEEE Trans. Inform. Theory,* vol. 45, no. 6, pp. 1757–1767, September 1999.

[10] H. Hasse, "Theorie der höheren Differentiale in einem algebraishen Funcktionenkörper mit vollkommenem Konstantenkörper nei beliebeger Charakteristic," *J. Reine. Ang. Math.*, vol. 175, pp. 50–54, 1936.

[11] D. E. Knuth, *The Art of Computer Programming, vol. 1: Fundamental Algorithms,* Reading, Massachusetts: Addison-Wesley, 1973.

[12] R. Kötter, *On Algebraic Decoding of Algebraic-Geometric and Cyclic Codes*, *Linköping Studies in Science and Technology,* no. 419 (Ph.D. Dissertation, Department of Electrical Engineering), Linköping U., 1996.

[13] R. Kötter, "Fast Generalized Minimum-Distance Decoding of Algebraic-Geometry and Reed–Solomon Codes," *IEEE Trans. Inform. Theory,* vol. 42, no. 3, pp. 721–736, May 1996.

[14] R. Kötter and A. Vardy, "Algebraic Soft-Decision Decoding of Reed–Solomon Codes," submitted to IEEE Trans. Inform. Theory, preprint dated May 31, 2000, and preprint dated August 31, 2001.

[15] J. L. Massey, "Shift-Register Synthesis and BCH Decoding," *IEEE Trans. Inform. Theory,* vol. 15, no. 1, pp. 122–127, January 1969.

[16] R. J. McEliece, *The Theory of Information and Coding,* 2nd ed. Cambridge, England: Cambridge U. Press, 2002.

[17] R. J. McEliece and L. Swanson, "On the Decoder Error Probability for Reed–Solomon Codes," *IEEE Trans. Inform. Theory,* vol. IT-32, no. 5, pp. 701–703, September 1986.

[18] W. H. Mills, "Continued Fractions and Linear Recurrences," *Mathematics of Computation*, vol. 29, no. 129, pp. 173–180, January 1975.

[19] R. Nielsen and T. Hoeholdt, "Decoding Reed–Solomon Codes beyond Half the Minimum Distance," in *Cryptography and Related Areas,* J. Buchmann, T. Hoeholdt, H. Stichenoth, and H. Tapia-Recillas, eds., Springer-Verlag, pp. 221–236, 2000.

[20] I. S. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields," *J. Soc. Industrial Appl. Math.,* vol. 8, pp. 300–304, 1960.

[21] R. Roth and G. Ruckenstein, "Efficient Decoding of Reed–Solomon Codes beyond Half the Minimum Distance," *IEEE Trans. Inform. Theory,* vol. 46, no. 1, pp. 246–257, January 2000.

[22] C. E. Shannon, *The Mathematical Theory of Communication,* Urbana Illinois: University of Illinois Press, 1949.

[23] M. Sudan, "Decoding of Reed–Solomon Codes beyond the Error-Correction Bound," *J. Complexity*, vol. 13, pp. 180–193, 1997.

[24] L. R. Welch and E. R. Berlekamp, "Error Correction for Algebraic Block Codes," U.S. Patent no. 4,633,470, December 30, 1986.

[25] L. R. Welch and R. A. Scholtz, "Continued Fractions and Berlekamp's Algorithm," *IEEE Trans. Inform. Theory,* vol. 25, no. 1, pp. 19–27, January 1979.

Appendix A
The Function $F(x)$ and the Numbers K_m

In this appendix, we collect for reference a number of technical results that are needed in order to give precise descriptions of various parameters associated with the $GS(m)$-decoder.

I. The Function $F(x)$

In this subsection, we collect for reference the important properties of the function $F(x)$, defined for $x \geq 0$ by

$$F(x) \triangleq \frac{1}{2}\big(x^2 + x + f(x)\big) \tag{A-1}$$

where

$$f(x) \triangleq \{x\}\big(1 - \{x\}\big) \tag{A-2}$$

and $\{x\}$ denotes the fractional part of x, i.e., $\{x\} = x - \lfloor x \rfloor$. As we will see below, Eq. (A-5), the function $F(x)$ is piecewise linear and its graph consists of a sequence of chords connecting the points

$$\left\{(0,0),(1,1),(2,3),(3,6),\cdots,\left(m,\binom{m+1}{2}\right),\cdots\right\}$$

as shown in Fig. A-1.

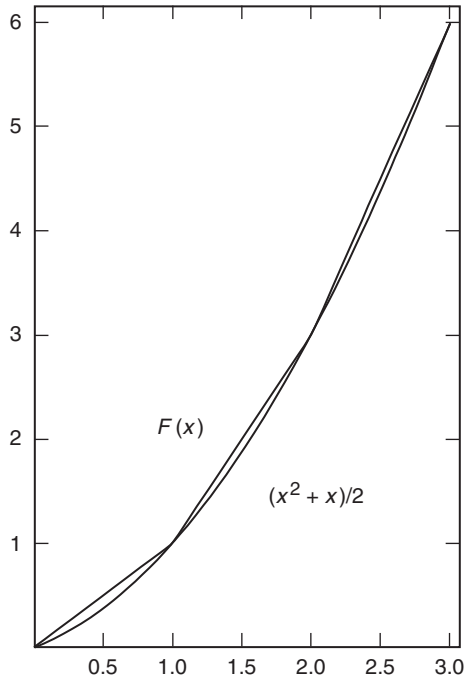

Fig. A-1. The functions $F(x)$ and $(x^2 + x)/2$, $0 \leq x \leq 3$.

Of course the reason $F(x)$ interests us is Corollary 1:

$$A(K,v) = vF\left(\frac{K}{v}\right) \tag{A-3}$$

Here are the important properties of $F(x)$.

Theorem A-1. *$F(x)$ has the following properties.*

$$\frac{1}{2}\big(x^2+x\big) \le F(x) \le \frac{1}{2}\left(x+1/2\right)^2, \quad \textit{for all } x>0 \tag{A-4}$$

$$F(x) = mx - \binom{m}{2}, \quad \textit{for } m-1 \le x \le m \tag{A-5}$$

$$F'(x) = \lceil x \rceil, \quad \textit{if } x \notin \mathbb{N} \tag{A-6}$$

$$F(x) \ge mx - \binom{m}{2}, \quad \textit{for all } m \ge 1 \tag{A-7}$$

$$F(x) \le \frac{m+1}{2m}x^2, \quad \textit{for } x \ge m \tag{A-8}$$

$$F(mx) \le \frac{m}{m+2}F\big((m+1)x\big), \quad \textit{if } x \ge 1 \tag{A-9}$$

Proof of Eq. (A-4). These inequalities follow immediately from Definition (A-1) and the observation that $0 \le f(x) \le 1/4$. ❒

Proof of Eq. (A-5). This follows from Definition (A-1) and the observation that $m-1 \le x \le m$ implies $\{x\}(1-\{x\}) = (x-m+1)(m-x) = -x^2+(2m-1)x-m(m-1)$, so $F(x) = (1/2)(x^2+x-x^2+(2m-1)x-m(m-1)) = mx-m(m-1)/2$. ❒

Proof of Eq. (A-6). This follows immediately from Eq. (A-5). ❒

Proof of Eq. (A-7). This follows from Eq. (A-5) and the fact that $F(x)$ is convex $\cup$. ❒

Proof of Eq. (A-8). If m and m' are positive integers, define

$$P_{m,m'}(x) \triangleq (m'+1)x - \binom{m'+1}{2} - \frac{m+1}{2m}x^2 \tag{A-10}$$

Note that $P_{m,m'}(0) = -\binom{m'+1}{2} < 0$. The discriminant of $P_{m,m'}(x)$ is

$$\Delta_{m,m'} = \frac{m'+1}{m}\left(m-m'\right)$$

which is ≤ 0 if $m' \ge m$. Hence, if $m' \ge m$, $P_{m,m'}(x)$ never changes sign and therefore must be ≤ 0 for all x. But by Eq. (A-5), $P_{m,m'}(x) = F(x) - ([m+1]/2m)x^2$ for $m' \le x \le m'+1$. ❒

Proof of Eq. (A-9). We have, by routine algebra,

$$(m+2)F(mx) - mF\big((m+1)x\big) = \frac{m}{2}\left(x - x^2 + \frac{m+2}{m}f(mx) - f\big((m+1)x\big)\right)$$

where $f(x)$ is as defined in Eq. (A-2). Thus, Eq. (A-9) is equivalent to

$$F_m(x) \le x^2 - x \quad \text{for } x \ge 1 \tag{A-11}$$

where

$$F_m(x) \triangleq \left(1 + \frac{2}{m}\right) f(mx) - f\big((m+1)x\big) \tag{A-12}$$

Since the function $F_m(x)$ is periodic of period 1, Eq. (A-11) is equivalent to

$$F_m(x) \le (x+1)^2 - (x+1) = x^2 + x \quad \text{for } x \ge 0 \tag{A-13}$$

as illustrated in Fig. A-2. To prove Eq. (A-13), we need a lemma.

Lemma A-1. $F_m(x)$ satisfies the following (see Fig. A-3):

$$-f(x) \le F_m(x) \le (1 + 2/m)\, f(x) \quad \text{for } x \ge 0 \tag{A-14}$$

$$F_m(x) = (x+1)x \quad \text{for } 0 \le x \le 1/(m+1) \tag{A-15}$$

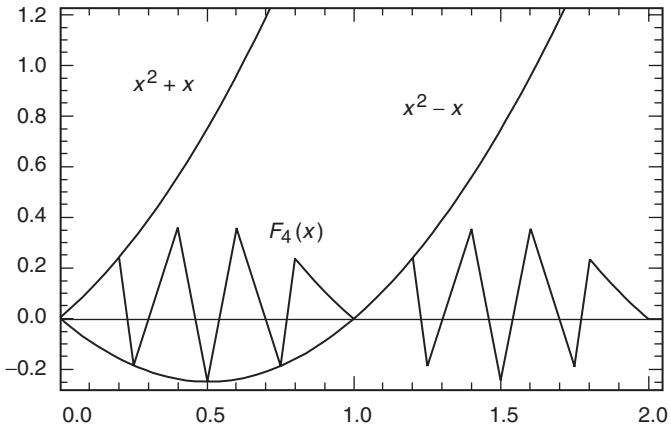

Fig. A-2. The functions $F_4(x)$, $x^2 + x$, and $x^2 - x$, $0 \le x \le 2$.

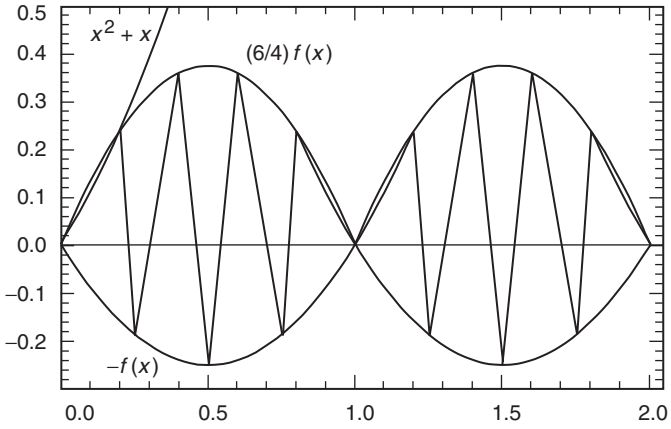

Fig. A-3. The function $F_4(x)$, and friends, $0 \le x \le 2$.

Proof. A tedious exercise in algebra yields the following: For $x \in [k/m, (k+1)/(m+1)]$, $k = 0, \cdots, m-1$:

$$mF_m(x) = mx^2 + m(2k+1)x - 2k(k+1) \tag{A-16}$$

$$mF_m(x) - (m+2)f(x) = 2\big((m+1)x - (k+1)\big)(x+k) \tag{A-17}$$

$$mF_m(x) + mf(x) = 2(mx-k)(k+1) \tag{A-18}$$

and for $x \in [k/(m+1), k/m]$, $k = 1, \cdots, m$:

$$mF_m(x) = mx^2 - (2m^2 - 2km + 3m)x + 2k(m-k+1) \tag{A-19}$$

$$mF_m(x) - (m+2)f(x) = 2\big((m+1)x - k\big)\big(x - (m-k+1)\big) \tag{A-20}$$

$$mF_m(x) + mf(x) = -2(mx-k)(m-k+1) \tag{A-21}$$

Note that Eq. (A-15) follows immediately from Eq. (A-16) if we let $k = 0$. To prove Eq. (A-14), we consider two cases:

Case A-1: $x \in [k/m, (k+1)/(m+1)]$, for $k = 0, \cdots, m-1$. Here the right side of Eq. (A-17) is negative, which proves the right inequality of Eq. (A-14). On the other hand, the right side of Eq. (A-18) is positive, which proves the left inequality of Eq. (A-14).

Case A-2: $x \in [k/(m+1), k/m]$, for $k = 1, \cdots, m$. Here the right side of Eq. (A-20) is negative, which proves the right inequality of Eq. (A-14). On the other hand, the right side of Eq. (A-21) is positive, which proves the left inequality of Eq. (A-14).

Finally,

$$F_m(x) \overset{\text{(A-15)}}{=} x^2 + x \quad \text{for } 0 \le x \le \frac{1}{m+1}$$

$$\overset{\text{(A-14)}}{\le} (1 + 2/m)\, f(x) < x^2 + x \quad \text{for } x > \frac{1}{m+1}$$

which completes the proof of Eq. (A-13) and therefore Eq. (A-9). ❒

II. The Numbers K_m

The $GS(m)$-decoder for an $(n, k = v+1)$ RS code is guaranteed to correct any pattern of up to t errors if there exists a positive integer D such that[14]

$$\left.\begin{aligned} m(n-t) &\ge D \\ n\binom{m+1}{2} &< A(D, v) \end{aligned}\right\} \tag{A-22}$$

which is equivalent to the condition

$$A\big(m(n-t), v\big) > n\binom{m+1}{2} \tag{A-23}$$

Using K (for "Korrect," or "Koradius") for $n - t$, an equivalent statement is that K correct received symbols are sufficient to guarantee that $GS(m)$ can identify the transmitted codeword if

$$A(mK, v) > n\binom{m+1}{2} \tag{A-24}$$

Now define for $1 \le v < n$,

$$K_0 \triangleq \left\lceil \frac{n+v+1}{2} \right\rceil \tag{A-25}$$

$$K_m \triangleq \min\left\{ K : A(mK, v) > n\binom{m+1}{2} \right\} \quad \text{for } m \ge 1 \tag{A-26}$$

$$K_\infty \triangleq \lfloor \sqrt{vn} \rfloor + 1 \quad \left((K_\infty - 1)^2 \le vn < K_\infty^2 \right) \tag{A-27}$$

The most important properties of the sequence $K_0, K_1, \cdots, K_m, \cdots, K_\infty$ are given in the following theorem. We might summarize these properties as follows:

[14] Here $D = 1 + \deg_{1,v} Q(x, y)$.

$$
\begin{array}{ll}
& K_0 \quad \text{(Conventional decoder)} \\
\geq & K_1 \quad \text{(Sudan decoder)} \\
\vdots & \\
\geq & K_m \quad \text{(GS decoder with interpolation multiplicity } m\text{)} \\
\vdots & \\
= & K_\infty \quad \text{(Most powerful, most complex GS decoder)}
\end{array}
$$

Theorem A-2.

$$K_0 \geq K_\infty \geq v + 1 \tag{A-28}$$

$$K_0 \geq K_1 \tag{A-29}$$

$$K_m \geq K_\infty \quad \text{if } m \geq 1 \tag{A-30}$$

$$K_m \geq K_{m+1} \quad \text{if } m \geq 1 \tag{A-31}$$

$$K_m = K_\infty \quad \text{for all sufficiently large } m \tag{A-32}$$

Proof of Eq. (A-28). First note that, if x and y are real numbers,

$$x > y \quad \text{implies} \quad \lceil x \rceil \geq \lfloor y \rfloor + 1 \tag{A-33}$$

Now by the arithmetic-geometric inequality,

$$\frac{n + v + 1}{2} \geq \sqrt{(v+1)n} > \sqrt{vn} \tag{A-34}$$

Combining Eq. (A-34) with Eq. (A-33), we get Eq. (A-28). ❒

Proof of Eq. (A-29). Since K_1 is defined in Eq. (A-26) as the least integer K such that $A(K, v) > n$, it is sufficient to show that $A(K_0, v) > n$. But

$$A(K_0, v) \overset{\text{(A-3)}}{=} vF\left(K_0/v\right) \overset{\text{(A-7)}_{m=2}}{\geq} v\left(2K_0/v - 1\right) = 2K_0 - v \overset{\text{(A-25)}}{\geq} n + 1$$

❒

Proof of Eq. (A-30). We will show that for all $m \geq 1$,

$$A\big(m(K_\infty - 1), v\big) \leq n\binom{m+1}{2} \tag{A-35}$$

which implies $K_m \geq K_\infty$ for all $m \geq 1$. Here we go:

$$A\big(m(K_\infty-1),v\big) \overset{\text{(A-3)}}{=} vF\left(m\frac{K_\infty-1}{v}\right)$$
$$\overset{\text{(A-8)}}{\leq} v\frac{m+1}{2m}m^2\frac{(K_\infty-1)^2}{v^2} \quad \big(\text{since } m(K_\infty-1)/v \geq m \text{ by Eq. (A-28)}\big)$$
$$\leq n\binom{m+1}{2} \quad \big(\text{since } (K_\infty-1)^2 \leq vn \text{ by Eq. (A-27)}\big)$$

❒

Proof of Eq. (A-31).

$$n\binom{m+1}{2} \overset{\text{(A-26)}}{<} A(mK_m,v)$$
$$\overset{\text{(A-3)}}{=} vF\left(mK_m/v\right)$$
$$\overset{\text{(A-9)}}{\leq} v\frac{m}{m+2}F\big((m+1)K_m/v\big) \quad (\text{using } K_m \geq v+1)$$
$$\overset{\text{(A-3)}}{=} \frac{m}{m+2}A\big((m+1)K_m,v\big)$$

Thus,

$$A\big((m+1)K_m,v\big) > n\binom{m+1}{2}\frac{m+2}{m} = n\binom{m+2}{2}$$

which implies [see Eq. (A-26)] that $K_{m+1} \leq K_m$. ❒

Proof of Eq. (A-32). We need to show that, for all sufficiently large m,

$$A(mK_\infty,v) > n\binom{m+1}{2} \tag{A-36}$$

We have

$$A(mK_\infty,v) \overset{\text{(A-3)}}{=} vF\left(\frac{mK_\infty}{v}\right) \overset{\text{(A-4)}}{>} \frac{m^2K_\infty^2}{2v} = n\binom{m+1}{2}\left\{\frac{m}{m+1}\frac{K_\infty^2}{vn}\right\}$$

which proves that Eq. (A-36) holds for $(m/[m+1])(K_\infty^2/vn) > 1$, i.e.,

$$m > \left(\frac{K_\infty^2}{vn}-1\right)^{-1}$$

❒

Appendix B
Decoding when Erasures are Present

In this appendix, we briefly discuss the modifications necessary in the $GS(m)$-algorithm if erasures are present in the garbled codeword.

Suppose the codeword suffers e erasures, which for notational convenience we assume are in positions $n-e+1, \cdots, n$. Then the received word is $(\beta_1, \cdots, \beta_{n-e}, *, *, \cdots, *)$. In this case, the $GS(m)$ decoder constructs a polynomial with a zero of multiplicity m at each of the $n-e$ points $(\alpha_1, \beta_1), \cdots, (\alpha_{n-e}, \beta_{n-e})$. It is then easy to show that the list of y-roots of $Q(x, y)$ will contain every codeword that agrees with the transmitted codeword in at least $K_m(n-e, k)$ of the unerased positions. Hence, we have the following.

Theorem B-1. *For an (n, k) RS code, the $GS(m)$ decoder will correct any pattern of e erasures and t errors, provided*

$$e \leq n - k$$

and

$$t \leq t_m(n-e, k)$$

Conventionally, of course,

$$e \leq n - k$$

and

$$t \leq \left\lfloor \frac{(n-k-e)}{2} \right\rfloor$$

Thus, for example, for the $GS(\infty)$-decoder, the "achievable pairs" are

$$P_\infty(n,k) \overset{\text{(A-27)}}{=} \left\{ (e,t) : \frac{(n-e-t)^2}{n-e} > k-1 \right\}$$

versus the conventional

$$P_0 = \big\{ (e,t) : n-e-2t > k-1 \big\}$$

Sanity check:

$$(n-e-2t) = \frac{(n-e-t)+t}{n-e}\big((n-e-t)-t\big) = \frac{(n-e-t)^2-t^2}{n-e} \leq \frac{(n-e-t)^2}{n-e}$$

The relative sizes of these regions is illustrated in Fig. B-1.

Example B-1. For the $(32, 8)$ RS code over $GF(32)$, if the codeword suffers e erasures, the following table shows how many errors also can be corrected, if (1) conventional decoding; (2) a $GS(3)$ decoder; and (3) a $GS(\infty)$ decoder is used.

e :	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
t_0 :	12	11	11	10	10	9	9	8	8	7	7	6	6	5	5	4	4	3	3	2	2	1	1	0	0
t_3 :	15	15	14	13	12	12	11	10	10	9	8	8	7	6	6	5	4	4	3	3	2	1	1	0	0
t_{GS} :	17	16	15	14	13	13	12	11	11	10	9	8	8	7	6	6	5	5	4	3	2	2	1	1	0

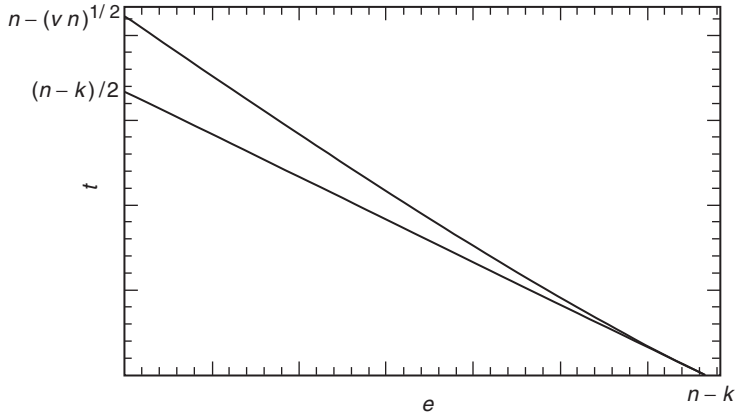

Fig. B-1. Combined erasure, e, and error, t, correction, conventional and $GS(\infty)$.

Appendix C
A GS-Type Conventional RS Decoding Algorithm

If Kötter's interpolation algorithm is specialized to $m = 1$ and $L = 1$, the result is an algorithm that can be used as an RS decoding algorithm in the conventional sense. (A similar observation was made in [21].) In the following pseudocode, $(\beta_1, \cdots, \beta_n)$ is a noisy codeword from an (n, k) RS code consisting of all codewords of the form $\big(f(\alpha_1), \cdots f(\alpha_n)\big)$, where $(\alpha_1, \ldots, \alpha_n)$ is a list of n distinct elements of F, and $f(x) \in F_{k-1}[x]$. This algorithm involves no syndrome calculation or error value/location evaluation.

```
/* An O(n^2) GS-like RS Decoding Algorithm */

BEGIN (Given {(α_i, β_i)}_{i=1}^n, (1, k−1)-revlex order)
1.  g_0(x, y) := 1; δ_0 = 0.
2.  g_1(x, y) := y; δ_1 = k − 1.
3.  FOR i = 1 to n DO
4.        FOR j = 0 to 1 DO
5.            Δ_j := g_j(α_i, β_i)            /* jth discrepancy */
6.        J := {j : Δ_j ≠ 0}
7.        If J ≠ ∅
8.            j* := argmin {g_j : j ∈ J}
9.            f := g_{j*};  Δ := Δ_{j*}
10.           FOR j ∈ J DO
11.                IF (j ≠ J*)
12.                     g_j := Δ g_j − Δ_j f
13.                ELSE IF (j = J*)
14.                     g_j := (x − α_i) g_j
15.                     δ_j := δ_j + 1   /* (1, k−1)-deg increases by 1 */
16. Q(x, y) := min_j{g_j(x, y)} = P_1(x)y − P_0(x)
17. f(x) rem r(x) := P_0(x) ÷ P_1(x)
18. IF r(x) == 0 and deg f(x) ≤ k − 1
19.       Print "transmitted codeword was f(x)"
20. ELSE
21.       Print "uncorrectable error pattern"
END
```

Example C-1. Consider a (5,2) RS code over $F = GF(5)$, which is conventionally capable of correcting 1 error. Suppose we are given

$i:$	1	2	3	4	5
$\alpha_i:$	0	1	2	3	4
$\beta_i:$	1	3	3	2	4

(Here $D(2x+1, \beta) = 1$.) The following table shows how the algorithm proceeds.

i	(α_i, β_i)	$g_0(x,y)$	$g_1(x,y)$
0	—	1	y
1	(0,1)	x	$y+4$
2	(1,3)	x^2+4x	$y+3x+4$
3	(2,3)	$3y+3x^2+x+2$	$y(x+3)+(3x^2+3x+2)$
4	(3,2)	$y(3x+1)+(3x^3+2x^2+4x+4)$	$y(x+3)+(3x^2+3x+2)$
5	(4,4)	$y(3x^2+4x+1)+(3x^4+x^2+3x+4)$	$y(x+3)+(3x^2+3x+2)$

Hence, $Q(x,y) = g_1(x,y) = y(x+3) - (2x^2+2x+3)$, $f(x) = 2x+1$, $r(x) = 0$, so the decoder's output is $2x+1$, which indeed produces a codeword that differs from the received word in only one position, viz., $i = 3$. (Interestingly, if $g_0(x,y)$ is taken instead, $f(x) = 4x^2+3x+1$ results, with $D(f,\beta) = 2$.)

Example C-2. Consider the same $(5,2)$ RS code over $F = GF(5)$, but now we are given

$i:$	1	2	3	4	5
$\alpha_i:$	0	1	2	3	4
$\beta_i:$	1	3	3	3	4

(Here $D(2x+1, \beta) = 2$.) The following table shows how the algorithm proceeds.

i	(α_i, β_i)	$g_0(x,y)$	$g_1(x,y)$
0	—	1	y
1	(0,1)	x	$y+4$
2	(1,3)	x^2+4x	$y+3x+4$
3	(2,3)	$3y+3x^2+x+2$	$y(x+3)+(3x^2+3x+2)$
4	(3,3)	$y(3x+1)+(3x^3+2x^2+4x+4)$	$xy+2x$
5	(4,4)	$y(x+4)+(2x^3+3x^2+4x+1)$	$y(x^2+x)+(2x^2+2x)$

Hence, $Q(x,y) = g_0(x,y) = y(x+4) - (3x^3+2x^2+x+4)$, $f(x) = 3x^2+1$, $r(x) = 0$, so the decoder's output is "uncorrectable error pattern," even though $D(f,\beta) = 1$. (Interestingly, if $Q(x,y)$ is chosen to be $g_1(x,y)$ instead, $f(x) = 3$, which has $D(f,\beta) = 2$, is output.)

A mathematical discussion of the algorithm's behavior follows. We introduce the notation

$$\Delta_0(n,k) = \left\lceil \frac{n+k}{2} \right\rceil - 1$$

$$\Delta_1(n,k) = \left\lfloor \frac{n-k}{2} \right\rfloor$$

and

$$K(f;\beta) = |\{i : f(\alpha_i) = \beta_i\}| \quad \text{(agreements)}$$

$$D(f;\beta) = |\{i : f(\alpha_i) \neq \beta_i\}| \quad \text{(disagreements)}$$

Lemma C-1.

$$\Delta_0(n,k) + \Delta_1(n,k) = n - 1 \tag{C-1}$$

$$k - 1 \leq \Delta_0(n,k) - \Delta_1(n,k) \leq k \tag{C-2}$$

Lemma C-2. $y - f(x)$ divides $Q(x,y) = P_1(x)y - P_0(x)$ if and only if $P_1(x) \,|\, P_0(x)$, in which case $f(x) = P_0(x)/P_1(x)$.

Theorem C-1. *This algorithm will return a polynomial* $Q(x,y) = yP_1(x) - P_0(x)$ *with* $\deg P_0(x) \leq \Delta_0$ *and* $\deg P_1(x) \leq \Delta_1$.

Proof. The polynomial $Q(x,y)$ must satisfy the n constraints $Q(\alpha_i, \beta_i) = 0$, for $i = 1, \cdots, n$. Thus, $Q(x,y)$ will be a linear combination of the first $n+1$ monomials from $F_L[x,y]$ in $(1, k-1)$-revlex order. By Eq. (C-1), there are $\Delta_0 + \Delta_1 + 2 = n + 1$ monomials in the sets

$$\left\{1, x, \cdots, x^{\Delta_0}\right\}$$

and

$$\left\{y, xy, \cdots, x^{\Delta_1} y\right\}$$

and these are the first $n+1$ monomials from $F[x, y^{(1)}]$ in $(1, k-1)$-revlex order, since $x^{\Delta_0+1} > x^{\Delta_1} y$ by the left side of Eq. (C-2) and $x^{\Delta_1+1} y > x^{\Delta_0}$ by the right side of Eq. (C-2). ❒

Theorem C-2. *The algorithm returns* $f(x)$ *if and only if* $f(x) \in F_{k-1}[x]$ *and* $D(f;\beta) \leq \Delta_1(n,k) = \lfloor (n-k)/2 \rfloor$. *If there is no such* $f(x)$, *it prints "uncorrectable error pattern."*

Proof. (Only if.) Suppose that the algorithm returns $f(x)$, i.e., $f(x) \in F_{k-1}[x]$ and $y - f(x)$ divides $Q(x,y)$. By Lemma C-2, $y - f(x)$ divides $Q(x,y) = P_1(x)y - P_0(x)$ if and only if $f(x) = P_0(x)/P_1(x)$. This algorithm, as a special case of Kötter's algorithm, guarantees that

$$Q(\alpha_i, \beta_i) = P_1(\alpha_i)\beta_i - P_0(\alpha_i) = 0 \quad \text{for } i = 1, \cdots, n$$

Thus, if $P_1(\alpha_i) \neq 0$, we have

$$\beta_i = \frac{P_0(\alpha_i)}{P_1(\alpha_i)} = f(\alpha_i) \tag{C-3}$$

But since $\deg P_1(x) \leq \Delta_1$, there can be at most Δ_1 exceptions to Eq. (C-3), i.e., $D(f;\beta) \leq \Delta_1 = \lfloor (n-k)/2 \rfloor$.

(If.) Suppose $f(x) \in F_{k-1}[x]$ and $D(f;\beta) \leq \lfloor (n-k)/2 \rfloor$. By Theorem C-1 and Eq. (C-2),

$$\deg_{1,k-1} Q(x,y) \leq \max\left(\Delta_1 + (k-1), \Delta_0\right) \leq \Delta_0$$

It follows from Theorem 7 that, if the score of $f(x)$ exceeds Δ_0, then $y - f(x)$ divides $Q(x,y)$. But in this case, the score of $f(x)$ is

$$S_1(f) = K(f;\beta) = n - D(f;\beta)$$

so that $y - f(x)$ divides $Q(x,y)$ provided $D(f;\beta) \leq n - \Delta_0 - 1$. But

$$n - \Delta_0 - 1 = n - \left\lceil \frac{(n+k)}{2} \right\rceil = \left\lfloor \frac{(n-k)}{2} \right\rfloor$$

❒

Appendix D
The Average Size of the List

Technically, the GS decoding algorithm is a "list" decoder, i.e., the decoder's output is a list of candidate codewords. In this appendix, we will show that, for the GS decoder, the list is unlikely to contain more than one codeword. (The only previous work on this topic we are aware of is [19], which presents an upper bound on the probability of having more than one codeword on the list. However, the expression obtained is difficult to evaluate and appears to provide little insight.)

We have seen that the $GS(m)$ decoding algorithm returns a list that is guaranteed to include all codewords within distance t_m of the received word. Let us denote by L the number of such codewords; L is a random variable that depends on the channel noise. We have seen that in the worst case L cannot exceed L_m, defined in Eqs. (42) and (45) and closely bounded above by

$$L_m < \left(m + \frac{1}{2}\right)\sqrt{\frac{n}{k-1}}$$

But what about the *average* number of codewords on the list? If the number of channel errors is t_m or less, the causal codeword will certainly be on the list. Let us take the presence of the causal codeword for granted and consider the average number of noncausal codewords on the list. To do this, we need to explore the combinatorics of RS codes a bit.

Thus, let $\mathcal{C}$ be an (n, k) RS code over $GF(q)$, with redundancy $r = n - k$ and minimum distance $d = n - k + 1$. Let $\mathcal{C}^*$ be the set of nonzero codewords, and let E be an arbitrary vector of length n over $GF(q)$. We make the following definitions:

$$f(E, t) = |\{C \in \mathcal{C}^* : |E - C| \le t\}| \tag{D-1}$$

$$D(u, t) = \sum_{|E|=u} f(E, t) \tag{D-2}$$

The interpretation is this. If $(0, \cdots, 0)$ is the transmitted codeword, and E is received, $f(E, t)$ represents the number of nonzero codewords with distance t or less from E. If $f(E, t) = m$, we say that E is m-tuply falsely decodable. By linearity, if C is the transmitted codeword, and E is the error pattern, $f(E, t)$ is also the number of noncausal codewords at distance $\le t$ from the received word $R = C + E$. Thus, $D(u, t)$ is the total number of falsely decodable words of weight u, where an m-tuply falsely decodable word is counted m times.

Theorem D-1. *Consider a bounded distance decoder with decoding radius t. If $|E| = u$, then the average number of noncausal codewords (averaged over all error patterns of weight u) in the decoding sphere is given by*

$$\overline{L}(u, t) = \frac{D(u, t)}{\binom{n}{u}(q-1)^u} \tag{D-3}$$

If $P(u, t)$ denotes the probability that there exists at least one noncausal codeword within distance t of the received word R,

$$P(u,t) \le \overline{L}(u,t) \quad for\ all\ u\ and\ t \tag{D-4}$$

$$P(u,t) = \overline{L}(u,t) \quad if\ 2t \le r \tag{D-5}$$

Proof. If E is the error pattern, then the number of noncausal codewords at distance t or less from R is, by definition, $f(E,t)$. Since there are $\binom{n}{u}(q-1)^u$ error patterns of weight u, the average of $f(E,t)$ over all error patterns of weight u is

$$\frac{\sum_{E:|E|=u} f(E,t)}{\binom{n}{u}(q-1)^u} = \frac{D(u,t)}{\binom{n}{u}(q-1)^u}$$

which proves Eq. (D-3). To prove Eqs. (D-4) and (D-5), we note that if X is a random variable assuming nonnegative integer values, and $p_i = \Pr\{X = i\}$, then

$$\Pr\{X > 0\} = \sum_{i \ge 1} p_i \le \sum_{i \ge 1} i p_i = E(X)$$

with equality iff $\Pr\{X \ge 2\} = 0$. If X represents the number of noncausal codewords within distance t of R, the above inequality is equivalent to Eq. (D-4). To prove Eq. (D-5), we note that, if $2t \le r$, it is impossible for a sphere of radius t to contain two or more codewords, i.e., $\Pr\{X \ge 2\} = 0$. ❒

Theorem D-1 tells that, to compute the average number of noncausal codewords within distance t of R, it is enough to know the numbers $D(u,t)$. Fortunately, several previous authors have considered these numbers.[15] Of course we know from prehistory that

$$D(u,t) = 0 \quad \text{if } u + t \le r = d - 1 \tag{D-6}$$

Less trivially, in 1978 Berlekamp and Ramsey [3] proved that

$$\overline{L}(u,t) = \frac{1}{(q-1)^{u-1}} \binom{n-u}{t} \quad \text{if } u + t = r + 1 = d \tag{D-7}$$

In 1986, McEliece and Swanson [17] proved that for all (u,t),

$$\overline{L}(u,t) \le \overline{L}_1(u,t) := \frac{1}{(q-1)^r} \sum_{s=d-u}^{t} (q-1)^s \left\{ \sum_{w=d-u}^{s} \binom{n-u}{w} \binom{u}{s-w} \right\} \tag{D-8}$$

$$\le \overline{L}_2(u,t) := \frac{1}{(q-1)^r} \sum_{s=d-u}^{t} (q-1)^s \binom{n}{s} \tag{D-9}$$

[15] The articles [3], [17], and [5] predate GS and ostensibly apply only to the "conventional" case $t \le r/2$. However, a close inspection of the proofs shows that the formulas for $D(u,t)$ are valid even if $t > r/2$, provided the definition of $D(u,t)$ is modified to include multiple erroneous decodings.

$$\leq \overline{L}_0(t) := \frac{1}{(q-1)^r} \sum_{s=0}^{t} (q-1)^s \binom{n}{s} \tag{D-10}$$

Finally, in 1989 Cheung [5] gave an exact closed-from expression for $D(u,t)$ that is suitable for numerical calculations but is too complex to reproduce here. However, numerical experimentation with Cheung's formula indicates that the maximum value of $\overline{L}(u,t)$, for $u \geq r+1-t$, is just a hair's breadth larger than

$$\overline{L}(t) = \frac{(q-1)^r}{q^r}\, \overline{L}_0(t) = \frac{1}{q^r} \sum_{s=0}^{t} \binom{n}{s} (q-1)^s \tag{D-11}$$

which is the average number of codewords in a randomly selected Hamming sphere of radius t.

Example D-1. (Cf. Example 4). Let $(n,k) = (31,15)$ with $q = 32$. Then $t_0 = 8$ and $t_{GS} = 10$. If we take $m = 3$, $t_m = 9$, and use Cheung's formula [5] to find the exact values of $\overline{L}(u)$, and Eqs. (D-8) and (D-9), to get the more easily computed upper bounds $\overline{L}_1(u,t)$ and $\overline{L}_2(u,t)$, we obtain the following table. (By way of comparison, $\overline{L}(9) = 0.000446534$, which is indeed just a hair's breadth smaller than the maximum value of $\overline{L}(u,9)$, viz., $\overline{L}(15,9) = 0.000446693$.)

u	$\overline{L}(u,9)$	$\overline{L}_1(u,9)$	$\overline{L}_2(u,9)$	$\overline{L}_0(9)$
≤ 7	0	0	0	0.000742107
8	0.000029702	0.000029702	0.000732758	0.000742107
9	0.000110353	0.000123059	0.000742007	0.000742107
10	0.000223257	0.000276441	0.000742106	0.000742107
11	0.000328446	0.000447558	0.000742107	0.000742107
12	0.000398966	0.000589304	0.000742107	0.000742107
13	0.000433257	0.000679193	0.000742107	0.000742107
14	0.000444652	0.000722651	0.000742107	0.000742107
15	0.000446693	0.000738051	0.000742107	0.000742107
16	0.000446618	0.000741676	0.000742107	0.000742107
17	0.000446524	0.000742107	0.000742107	0.000742107
	$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
	0.000446534	0.000742107	0.000742107	0.000742107

If we assume the decoder declares "success" if the list contains exactly one codeword, and failure otherwise, then

$$\Pr\{\text{decoder failure}|\leq 7 \text{ errors}\} = 0$$

$$\Pr\{\text{decoder failure}|8 \text{ errors}\} = 0.000029702$$

$$\Pr\{\text{decoder failure}|9 \text{ errors}\} = 0.000110353$$

$$\Pr\{\text{decoder failure}|\geq 10 \text{ errors}\} = 1$$

whereas a (conventional $t = 8$) decoder for this code would have

$$\Pr\{\text{decoder failure}|\leq 7 \text{ errors}\} = 0$$

$$\Pr\{\text{decoder failure}|8 \text{ errors}\} = 0$$

$$\Pr\{\text{decoder failure}|9 \text{ errors}\} = 1$$

$$\Pr\{\text{decoder failure}|\geq 10 \text{ errors}\} = 1$$

In view of these numbers, it is indeed fair to say that the $GS(3)$ decoding algorithm can correct 9 errors.