Gröbner Basis techniques for certain problems in coding and systems theory

A thesis submitted to the National University of Ireland

for the degree of Ph.D

Henry O'Keeffe

Supervisor: Professor Patrick Fitzpatrick

Department of Mathematics

Faculty of Science

University College Cork

Ireland

October 2003

Abstract

There is much common ground between the areas of coding theory and systems theory. Fitzpatrick has shown that a Gröbner basis approach leads to efficient algorithms in the decoding of Reed-Solomon codes and in scalar interpolation and partial realisation. This thesis simultaneously generalises and simplifies that approach and presents applications to discrete-time modeling, multivariable interpolation and list decoding.

Gröbner basis theory has come into its own in the context of software and algorithm development. By generalising the concept of polynomial degree, term orders are provided for multivariable polynomial rings and free modules over polynomial rings. The orders are not, in general, unique and this adds, in no small way, to the power and flexibility of the technique. As well as being generating sets for ideals or modules, Gröbner bases always contain a element which is minimal with respect to the corresponding term order.

Central to this thesis is a general algorithm, valid for any term order, that produces a Gröbner basis for the solution module (or ideal) of elements satisfying a sequence of generalised congruences. These congruences, based on shifts and homomorphisms, are readily applicable to a wide variety of problems, including key equations and interpolations. At the core of the algorithm is an incremental step. Iterating this step lends a recursive/iterative character to the algorithm. As a consequence, not all of the input to the algorithm need be available from the start and different "paths" can be taken to reach the final solution. The existence of a suitable chain of modules satifying the criteria of the incremental step is a prerequisite for applying the algorithm.

Some problems further constrain the "degrees" of the components in the required solution. In this situation, the solution set is not a module (resp. ideal). Nonetheless, by deriving a specific term order from the degree constraints, the solution set can be described using a Gröbner basis for that order.

For the case where the base polynomial ring for the module is in a single indeterminate, the algorithm takes on a more regular structure and a simpler implementation. The algorithm in its most general form has a worst case complexity of $O(n^3)$. In problems leading to applications in a single indeterminate, this complexity is more typically $O(n^2)$.

The immediate applications are to certain problems for which Fitzpatrick has given algorithms that are special cases of the general one.

- Scalar partial realisation
- Scalar rational interpolation
- Key equations for decoding alternant codes
- Single congruence problem in a polynomial ring with more than one indeterminate.

We extend this list to include

- Systems theory
- Matrix partial realisation and extended M-Padé approximation
- List decoding for Reed-Solomon codes
- List decoding for 1-point Algebraic Geometry codes
- Soft decision list decoding.

A number of other specialised applications are also developed.

The general algorithm has a number of applications in *systems theory*. Homogeneous and inhomogeneous block Hankel and block Toeplitz linear systems of equations can readily be solved from (slightly) more general polynomial congruences. It can be used to derive a model of minimal complexity for autonomous, linear, time-invariant time series. Multivariable (in the systems theory sense) partial realisation is a special case of this behavioural approach. While the algorithm is primarily a computational tool, it can lead to closed form solutions and some models that can be "written down" symbolically without actually executing the algorithm.

The interpolation capabilities of the algorithm, combined with a degree constraint term order, make it applicable to matrix rational interpolation and the extended M-Padé aproximation problem. The latter is a generalisation of that of Padé and Hermite-Padé. While these problems are in a single variable, our algorithm extends to solutions of the multivariable case.

Finally, it is applied to the interpolation steps in *list decoding*. These form a major component of list decoding approaches based on Sudan's algorithm. Because limits can be placed on the search space, efficiencies can be introduced.

For Reed-Solomon codes, one specialisation solves a key equation derived for low rate codes and, for the extended case, another coincides with a well known algorithm. For soft decision decoding, it gives a new algorithm. The hard decision problem is seen to be a special case of the soft decision algorithm.

List decoding of 1-point algebraic geometry codes is approached in a similar way. By associating vector space basis elements from the function field with the standard basis vectors of a free module, the problem is transformed

into a module setting. The hard decision case is similar to an existing algorithm, albeit one based on a vector space approach. The soft decision approach is new and the hard decision case can again be viewed as a special case.

Contents

Chapter 1

Introduction

The significant overlaps between systems theory and coding theory are well known. As early as 1967, Massey and Sain [41] show the close connections between covolutional and cyclic codes on one hand and finite-state machines on the other. In recent times, "behavioural" methods from systems theory are applied in Rosenthal, Schumacher and York [56] to convolutional codes. Kuijper [37] applies behaviours to the decoding of alternant codes.

The classical example of this synergy is the Berlekamp-Massey algorithm. Berlekamp (1968) [8] shows how BCH codes can be decoded by solving a polynomial "key equation"

$$(1 + S(z))\sigma(z) \equiv \omega(z) \bmod z^{2t+1}.$$

That procedure involves only $O(t^2)$ multiplications. Massey [40] reinterprets the algorithm in systems theory terms to yield what became known as the Berlekamp-Massey algorithm. The result is a recursive algorithm which exploits the Hankel nature of the underlying coefficient matrix to improve on the $O(t^3)$ complexity of Gaussian Elimination. This leads on to solutions in scalar partial realisation and classic Padé approximation.

Fitzpatrick (1995) [22] addresses more general polynomial congruences of the form

$$a \equiv bh \bmod x^n.$$

Extending his previous work on Padé approximation and linear recurring sequences, it provides algorithms for decoding alternant codes. A new recursive/iterative technique is derived to handle these congruences by solving a sequence of partial problems

$$a \equiv bh \bmod x^k, 0 \leq k \leq n.$$

The solution (a, b) is the minimal element of a particular Gröbner basis for a submodule of a 2–dimensional module. This approach can be viewed as an analogue of the Berlekamp–Massey algorithm and, with some optimisations [27, 53], can have implemementation advantages over Berlekamp–Massey. By contrast, some Gröbner basis methods for decoding cyclic codes lead to algorithms with exponential complexity [13, 39, 44], although [44] does have polynomial complexity for some primitive BCH codes. Subsequently, the module approach was applied to scalar rational interpolation [23].

The iterative algorithm was extended in Fitzpatrick [26] to accomodate multivariable polynomial congruences

$$a \equiv \sum_{i=1}^{L} b_i h_i \bmod I.$$

It is upon this version that the present work builds. An iterative general algorithm is presented which includes these previous algorithms and applications as special cases. The direct proof of the general algorithm simplifies the theory, which had been built up on a case by case basis. By comparison with the earlier results where the term orders were hand crafted to the particular case, we show how a term order can be automatically derived to identify

degree constrained solutions. The generalisation allows us to include more general congruences and interpolations. The resulting algorithms retain the efficient nature of [22]. As well as subsuming the previous algorithms, for other applications the general approach leads on to procedures which are, in some cases, equivalent to existing "fast" algorithms and, in others, are new in themselves. Thus, as well as unifying the theory, it provides a template for practical algorithms.

1.1 Guide to the Thesis

Chapter 2. Gröbner basis theory. We introduce the Gröbner basis theory of multivariable polynomial rings and modules. We exhibit a new generalised term order which allows us to parameterise the solutions which are further constrained by the degrees of the module components. We rehearse some previously established applications of Gröbner basis theory to coding and systems theory.

Chapter 3. The general algorithm. We present and prove our new general algorithm. This algorithm provides a general setting for, and straightforward justification of, a variety of existing methods and leads on to new applications. We examine criteria for the applicability of the algorithm and consider its complexity. We deduce a simplified algorithm for the special case of polynomials in a single variable.

Chapter 4. Systems theory applications. Using polynomial congruences and a specialisation of our general term order, we show how Hankel and Toeplitz systems of equations can be solved with our algorithm. We apply our algorithm to modelling discrete time-invariant behaviours. Matrix partial realisation is included as a special case. We next apply it to matrix ratio-

nal interpolation and the generalised interpolations of the extended M-Padé problem.

Chapter 5. List decoding applications. Sudan-like algorithms have interpolations at their core. We apply our algorithm to the hard and soft decision list decoding of Reed-Solomon and 1-point Algebraic Geometry codes. The resulting algorithms for hard decision are similar to known good methods. New efficient algorithms are presented in the soft decision case.

Chapter 2

Gröbner basis preliminaries

In this chapter, we present term orders and Gröbner bases from the perspective of free modules. We prepare the notation and definitions for later use.

Many problems impose degree constraints on the solution polynomials. We exhibit a method by which a term order is derived from such constraints and show how the degree-constrained solutions can be described using a Gröbner basis derived from that order.

The chapter ends with a description of the applications which Fitzpatrick addressed using algorithms which are precursors of our general algorithm.

2.1 Gröbner bases and term orders

Gröbner basis theory was introduced by Buchberger in his Ph.D. thesis [12] (1965) and is now well-established. Detailed treatments and comprehensive references to original sources may be found in [1, 7, 14, 15, 28]. The early results deal with ideals of polynomial rings. Möller and Mora [43] develop the theory for polynomial modules. For this thesis, we will emphasise the

module description and will view a polynomial ring as a module over itself.

Let $A = F[x_1, \ldots, x_s]$ be a polynomial ring in s indeterminates over a field F. The set A^L is a free module of rank L over A.

2.1.1 Term orders

The standard basis vector with 1 in position i and 0 elsewhere (and length defined by the context) is denoted $\mathbf{e}_i$. A *term* in A^L is a vector of the type $\mathbf{X} = X\mathbf{e}_i$ where $X = x_1^{t_1} x_2^{t_2} \cdots x_s^{t_s}$ is a *term* in A. Thus a term in A^L is a vector all of whose components are 0 except for one which is a term in A. A *term order* in A^L is a total order $<$ on terms satisfying

(i) $\mathbf{X} < Z\mathbf{X}$ for each term $\mathbf{X}$ in A^L and each term $Z \neq 1$ in A,

(ii) if $\mathbf{X} < \mathbf{Y}$ then $Z\mathbf{X} < Z\mathbf{Y}$ for all terms $\mathbf{X}, \mathbf{Y}$ in A^L and each term Z in A.

The simplest form of a term order and, indeed, a model upon which term orders are based, is when $A^L = F[x]$. In this case, $s = L = 1$ and the order is determined (uniquely) by the exponent of x.

When $A = F[x_1, \ldots, x_s], s > 1$ there is more than one term order. *Lexicographic* order is defined as $x_1^{i_1} \ldots x_s^{i_s} <_{lex} x_1^{j_1} \ldots, x_s^{j_s}$, if there is a $k \in [s]$ such that $i_1 = j_1, \ldots, i_k = j_k$ and $i_{k+1} < j_{k+1}$ with $x_s < \ldots < x_1$. *Degree lexicographic* order is defined as $x_1^{i_1} \ldots, x_s^{i_s} <_{deglex} x_1^{j_1} \ldots, x_s^{j_s}$, if $\sum_{k=1}^s i_k < \sum_{k=1}^s j_k$ or $\sum_{k=1}^s i_k = \sum_{k=1}^s j_k$ and $x_1^{i_1} \ldots x_s^{i_s} <_{lex} x_1^{j_1} \ldots, x_s^{j_s}$. All term orders on A have been classifed by Robbiano [55].

An example of a module term order is obtained as follows. Let $<_{\text{lex}}$ denote lexicographic order in A and define $\mathbf{X} = X\mathbf{e_i} < Y\mathbf{e_j}$ if either $i < j$, or $i = j$ and $X <_{\text{lex}} Y$. This is a *position-over-term* (or POT) order.

Alternatively, define $\mathbf{X} = X\mathbf{e_i} < Y\mathbf{e_j}$ if either $X <_{\text{lex}} Y$, or $X = Y$ and $i < j$. This is a *term-over-position* (or *TOP*) order.

Further term orders can be produced using a weight vector $(w_1, \ldots, w_L)$ where each $w_i, 1 \leq i \leq L$ is a term in A. Based on any term order $<$ on A^L, a *weighted* term order $<_{(w_1,\ldots,w_L)}$ can be constructed by defining $X\mathbf{e_i} <_{(w_1,\ldots,w_L)} Y\mathbf{e_j}$ if $Xw_i < Yw_j$ or if $Xw_i = Yw_j$ and $i = \text{tb}(i, j)$, where tb is a tie breaking function on $\{1, \ldots, L\}$. When $A = F[x]$, we can, without ambiguity, describe the weights by reference to their exponents. Rust and Reid [59] classifies all terms orders on free modules A^L.

For non-zero $\mathbf{f} \in A^L$ we may write

$$\mathbf{f} = a_1\mathbf{X}_1 + \cdots + a_r\mathbf{X}_r \tag{2.1}$$

where the a_i are non-zero constants and the $\mathbf{X}_i$ are terms satisfying $\mathbf{X}_1 > \mathbf{X}_2 > \cdots > \mathbf{X}_r$. The *leading term* $\text{lt}(\mathbf{f})$ of $\mathbf{f}$ is $\mathbf{X}_1$, and the *leading coefficient* $\text{lc}(\mathbf{f})$ is a_1. These definitions are extended to all of A^L by setting $\text{lt}(\mathbf{0}) = \mathbf{0}, \text{lc}(\mathbf{0}) = 0$. For example, with $s = 2, L = 3$, $x_1 <_{\text{lex}} x_2$, and $\mathbf{f} = (x_1^2x_2 + 3x_1^3, x_2 + 1, 2x_1)$, and using the POT order defined above, we find that

$$\mathbf{f} = 2x_1\mathbf{e_3} + x_2\mathbf{e_2} + \mathbf{e_2} + x_1^2x_2\mathbf{e_1} + 3x_1^3\mathbf{e_1}$$

has leading term $x_1\mathbf{e_3}$ and leading coefficient 2. As usual $\text{supp}(\mathbf{f}) = \{\mathbf{X} | \mathbf{X}$ is a term, the coefficient of $\mathbf{X}$ in $\mathbf{f}$ is non-zero$\}$.

If $\mathbf{X} = X\mathbf{e}_i$ and $\mathbf{Y} = Y\mathbf{e}_j$ are terms in A^L we say $\mathbf{X}$ *divides* $\mathbf{Y}$ provided $i = j$ and X divides Y in A, that is, if there is a term Z (the quotient) in A satisfying $Z\mathbf{X} = \mathbf{Y}$.

Division among module elements can be performed once a term order is defined. This is a generalisation of the division algorithm for $F[x]$, based on "long division". The key to this process is *reduction*. If $\text{lt}(\mathbf{g})$ divides $\text{lt}(\mathbf{f})$,

$\mathrm{lt}(\mathbf{f}) = Z\mathrm{lt}(\mathbf{g})$ for some term $Z \in A$ and let $\mathbf{f}' = \mathbf{f} - \frac{\mathrm{lc}(\mathbf{f})}{\mathrm{lc}(\mathbf{g})} Z\mathbf{g}$. Then $\mathbf{f}'$ is said to be $\mathbf{f}$ *reduced* by $\mathbf{g}$.

2.1.2 Gröbner bases

A set of non-zero vectors $\mathcal{G} = \{\mathbf{g}_1, \ldots, \mathbf{g}_r\}$ contained in a submodule M is called a *Gröbner basis* of M if, for all $\mathbf{f} \in M$, there exists $i \in \{1, \ldots, r\}$ such that $\mathrm{lt}(\mathbf{g}_i)$ divides $\mathrm{lt}(\mathbf{f})$. We define $\mathcal{LT}(M)$ to be the the module generated by the leading terms of the elements of M. The leading terms of the elements of $\mathcal{G}$ generate $\mathcal{LT}(M)$. Further, $\mathcal{G}$ is a basis of M. Every submodule of A^L has a Gröbner basis. An extension of the original Buchberger algorithm can be used to derive a Gröbner basis, with respect to a chosen order, from any generating set.

Each $\mathbf{f} \in M$ has a *standard representation*, with respect to a Gröbner basis $\mathcal{G} = \{\mathbf{g}_1, \ldots, \mathbf{g}_m\}$, of the form $\mathbf{f} = \sum_{i=1}^{m} a_i \mathbf{g}_i$ where $a_i \in A$ and $\mathrm{lt}(f_i \mathbf{g}_i) \leq \mathrm{lt}(\mathbf{f})$, $1 \leq i \leq m$. A *minimal* element in a submodule $M \subseteq A^L$ is one whose leading term is least among the elements of M, under the given term order $<$. It is unique up to a constant multiple and must appear in any Gröbner basis relative to $<$.

Any element of A^L can be reduced by a Gröbner basis of M. The (uniquely defined) remainder is called the *normal form* of the element with respect to M and the term order.

We say that a Gröbner basis is *strictly ordered* if there are no duplicates among its leading terms (although some may be multiples of others), and its elements are in increasing order of leading term. Any Gröbner basis can be converted to a strictly ordered Gröbner basis by simply deleting any element whose leading term is the same as that of another element (choosing

arbitrarily in case of ties). In the algorithms presented here we will use a generic function *ord* which returns a strictly ordered Gröbner basis ord($\mathcal{G}$) for any Gröbner basis $\mathcal{G}$ (no difficulty arises as a consequence of any lack of uniqueness).

A Gröbner basis is *minimal* if none of its elements has leading term a multiple of the leading term of another of its elements. Again, any Gröbner basis can be converted into a minimal Gröbner basis simply by deleting appropriate elements. It is often more efficient, although not strictly necessary, to define the function *ord* so that it constructs ordered minimal Gröbner bases (see Example 5.2.2), but in some situations such bases are not appropriate (see Algorithm 5.2.5, Example 5.2.6) and so we do not adopt them universally.

2.1.3 Degree notation

We require several different types of degree. The *multidegree* $\Delta(X^{\underline{\beta}})$ of a term $X^{\underline{\beta}} = x_1^{\beta_1} x_2^{\beta_2} \dots x_s^{\beta_s} \in A$, $\beta_j \in \mathbb{N}_0, 1 \leq j \leq s$, is defined by $\Delta(X^{\underline{\beta}}) = \underline{\beta}$. If $<$ is a term order on A, the multidegree of $f \in A$ is given by $\Delta(f) = \Delta(\mathrm{lt}(f))$. More generally, if $<$ is a term order on A^L and $\mathrm{lt}(\mathbf{f}) = X^{\underline{\beta}}\mathbf{e}_i$ then $\Delta(\mathbf{f}) = \underline{\beta}$. Next, if $\underline{\alpha} = (\underline{\alpha}^{(1)}, \dots, \underline{\alpha}^{(L)}) \in (\mathbb{N}_0^n)^L$, we define $\underline{\lambda} = \underline{\lambda}_{\underline{\alpha}}$ by $X^{\underline{\lambda}} = \mathrm{lcm}\{X^{\underline{\alpha}^{(1)}}, \dots, X^{\underline{\alpha}^{(L)}}\}$. Then the $\underline{\alpha}$*–modified multidegree* of $\mathbf{f}$, where the leading term of $\mathbf{f}$ is $X^{\underline{\beta}}\mathbf{e}_i$, is defined as $\Delta_{\underline{\alpha}}(\mathbf{f}) = \underline{\beta} + \underline{\lambda} - \underline{\alpha}^{(i)}$. It will often be convenient to identify a term order on A with the corresponding total order on $\mathbb{N}_0^n$, that is, $X^{\underline{\beta}} < X^{\underline{\gamma}}$ if and only if $\underline{\beta} < \underline{\gamma}$, and we adopt this convention without further mention.

Example 2.1.1 In Section 2.2, we require a term order, based on the modified multidegree, defined as follows. First select an arbitrary but fixed term

order $<_1$ on A and let tb denote an arbitrary, fixed tie breaking function on $\{1, \ldots, L\}$. Let $\underline{\alpha} = (\underline{\alpha}^{(1)}, \ldots, \underline{\alpha}^{(L)})$ be fixed, and let $U = X^{\underline{\beta}_U}, V = X^{\underline{\beta}_V}$. Define $<$ by

$$\begin{array}{rcl} U\mathbf{e}_i & < & V\mathbf{e}_j \\ \text{when} & & \Delta_{\underline{\alpha}}(U\mathbf{e}_i) <_1 \Delta_{\underline{\alpha}}(V\mathbf{e}_j) \\ \text{or} & & \Delta_{\underline{\alpha}}(U\mathbf{e}_i) = \Delta_{\underline{\alpha}}(V\mathbf{e}_j) \text{ and } \mathrm{tb}(i,j) = i. \end{array} \tag{2.2}$$

The proof that this is a term order is given in Lemma 2.2.3. $\Diamond$

When $A = F[x]$, $\lambda = \max\{\alpha_i\}$ and $\Delta_{\underline{\alpha}}(x^\beta \mathbf{e}_i) = \beta + (\lambda - \alpha_i)$. Therefore, the term order (2.2) reduces a weighted order with $w_i = x^{\lambda - \alpha_i}$.

The *McMillan degree* of a polynomial matrix $M \in F[x]^{p \times (p+m)}$ is defined to be the maximum degree among the $p \times p$ minors of M.

For use in Chapter 5, if $f = \sum_{i,j} f_{ij} x^i y^j \in F[x, y]$ and a, b are non-negative integers we define the (a, b)–*degree* of f by $\Delta_{(a,b)}(f) = \max\{ai + bj | x^i y^j \in \mathrm{supp}(f)\}$, and use the conventional term *total* degree for the $(1, 1)$–degree (no confusion will arise between these uses of the symbol Δ).

Example 2.1.2 The (a, b)–degree can be used to construct term orders in $F[x, y]$ in which the terms are first ordered by (a, b)–degree, and then by some suitable tie breaker such as the degree in y. If $(a, b) = (2, 3)$ this leads to the following order

$$1 < x < y < x^2 < xy < x^3 < y^2 < x^2 y < x^4 < xy^2 < \cdots .$$

$\Diamond$

Finally, we will write ∂g for the degree of a 1–variable polynomial g.

2.2 Term orders for degree constraints

Many problems arising in practice additionally constrain the required solutions by imposing limits on the degrees of their components. For any chosen degree constraints, we show how to select a term order such that the elements of a module $S \subseteq A^L$ which satisfy these contraints can be identified by reference to a Gröbner basis for S with respect to that term order.

Let $<_1$ be a term order on A. A term order $<_L$ on A^L is said to be *compatible* with $<_1$ if $a\mathbf{e}_i <_L b\mathbf{e}_i$ whenever $a <_1 b$. The following lemma will be used freely in the sequel. (In fact, this can be viewed as an alternative definition of a compatible term order—see [17, p. 341].)

Lemma 2.2.1 *If* $\mathrm{lt}_1(h)$ *is the leading term of* $h \in A$ *with respect to* $<_1$ *and* $\mathrm{lt}(\mathbf{g})$ *is the leading term of* $\mathbf{g} \in A^L$ *with respect to a term order* $<_L$*, which is compatible with* $<_1$*, then*

$$\mathrm{lt}(h\mathbf{g}) = \mathrm{lt}_1(h)\mathrm{lt}(\mathbf{g})$$

Proof. Let $\mathrm{lt}_1(h) = u$, $\mathrm{lt}(\mathbf{g}) = w\mathbf{e}_k$. By the definition of term order, every term $v\mathbf{e}_j \in \mathrm{supp}(\mathbf{g})$ satisfies

$$uv\mathbf{e}_j <_L uw\mathbf{e}_k.$$

Every term $t \in \mathrm{supp}(h)$ satisfies $t <_1 u$. We have, by compatibility,

$$t\mathbf{e}_j <_L u\mathbf{e}_j$$

and, hence, by the definition of term order,

$$tv\mathbf{e}_j <_L uv\mathbf{e}_j.$$

Now every term in $h\mathbf{g}$ has the form $t(v\mathbf{e}_j)$ for $t \in \mathrm{supp}(h)$, $v\mathbf{e}_j \in \mathrm{supp}(\mathbf{g})$ and

$$t(v\mathbf{e}_j) = (tv)\mathbf{e}_j <_L (uv)\mathbf{e}_j <_L (uw)\mathbf{e}_k.$$

Thus $(uw)\mathbf{e}_k = \mathrm{lt}(h\mathbf{g})$ as required. □

The next example shows that not all term orders on A^L are compatible with some term order on A.

EXAMPLE 2.2.2 Let $A = F[x, y]$ and $L = 2$. If $<_1$ is any term order with $x < y$ (such as a lexicographic order) and $<_{1'}$ is any term order with $y < x$ (such as a different lexicographic order),we can define a term order $<_2$ on $F[x, y]^2$ by $X_1\mathbf{e}_i <_2 X_2\mathbf{e}_j$ if $(i < j)$ or $(i = j = 1$ and $X_1 <_1 X_2)$ or $(i = j = 2$ and $X_1 <_{1'} X_2)$. Let $h = x + y$ and let $<$ be any term order on A. Since $\mathrm{lt}_2((x+y)\mathbf{e}_1) = y\mathbf{e}_1$ and $\mathrm{lt}_2((x+y)\mathbf{e}_2) = x\mathbf{e}_2$ we see that $<_2$ is not compatible with $<$. ◊

We now consider the term order defined in Example 2.1.1.

Lemma 2.2.3 *The relation $<$ defined in* (2.2) *is a term order on A^L which is compatible with $<_1$.*

PROOF. Let $\underline{\lambda} = \underline{\lambda}_{\underline{\alpha}}$. The relation $<$ defines a total order on the terms of A^L, because $<_1$ is total order and, in the event of equality at the same component $\mathbf{e}_k$, we have $\underline{\beta}_U + \underline{\lambda} - \underline{\alpha}_k = \underline{\beta}_V + \underline{\lambda} - \underline{\alpha}_k$, so the terms are equal. Next, let $\mathbf{U} = X^{\underline{\beta}_U}\mathbf{e}_k, Z = X^{\underline{\beta}_Z} \neq 1$. Then $\underline{\beta}_U + \underline{\lambda} - \underline{\alpha}^{(k)} <_1 \underline{\beta}_Z + \underline{\beta}_U + \underline{\lambda} - \underline{\alpha}^{(k)}$ so $\mathbf{U} < Z\mathbf{U}$. Finally, if $\mathbf{U} = X^{\underline{\beta}_U}\mathbf{e}_i < \mathbf{V} = X^{\underline{\beta}_V}\mathbf{e}_j$ and $\underline{\beta}_U + \underline{\lambda} - \underline{\alpha}^{(i)} <_1 \underline{\beta}_V + \underline{\lambda} - \underline{\alpha}^{(j)}$ then adding $\underline{\beta}_Z$ to both sides does not change the order, while if $\underline{\beta}_U + \underline{\lambda} - \underline{\alpha}^{(i)} = \underline{\beta}_V + \underline{\lambda} - \underline{\alpha}^{(j)}$ and $\mathrm{tb}(i, j) = i$, then adding $\underline{\beta}_Z$ to both sides preserves the equality and $\mathrm{tb}(i, j) = i$ is unchanged. In either case $Z\mathbf{U} < Z\mathbf{V}$, as required. □

In many applications solutions $\mathbf{f} = (f^{(1)}, \ldots, f^{(L)})$ are sought, which satisfy a condition

$$\Delta(f^{(i)}) \leq_1 \underline{\alpha}^{(i)} \text{ for } 1 \leq i \leq L \tag{2.3}$$

for some fixed term order $<_1$ and vector $\underline{\alpha}$ of multidegrees. This generic problem of finding solution vectors satisfying degree constraints is the natural generalization of the 1–variable case considered in [22, 23]. Such "required solutions" can be determined from a Gröbner basis of the solution module.

Theorem 2.2.4 *Let $\mathcal{G} = \{\mathbf{g}_1, \ldots, \mathbf{g}_m\}$ be a Gröbner basis, with respect to the term order $<$ defined in* (2.2)*, of an A-module $S \subseteq A^L$. A vector $\mathbf{f} \in S$ satisfies* (2.3) *if and only it can be expressed in the form*

$$\mathbf{f} = \sum_{k \in K} a_k \mathbf{g}_k \tag{2.4}$$

where $K \subseteq \{1, \ldots, m\}$, and $a_k, \mathbf{g}_k$ satisfy $\Delta(a_k) + \Delta_{\underline{\alpha}}(\mathbf{g}_k) \leq_1 \underline{\lambda} = \underline{\lambda}_{\underline{\alpha}}$, for $k \in K$.

Proof. Suppose $\mathbf{f}$ can be expressed as in (2.4). Then $\mathrm{lt}(\mathbf{f}) \leq \mathrm{lt}(a_k \mathbf{g}_k) = \mathrm{lt}_1(a_k)\mathrm{lt}(\mathbf{g}_k)$ for some $k \in K$. If $\mathrm{lt}(\mathbf{f}) = X^{\underline{\beta}_f}\mathbf{e}_i$ and $\mathrm{lt}(\mathbf{g}_k) = X^{\underline{\beta}_g}\mathbf{e}_j$ then $\underline{\beta}_f + \underline{\lambda} - \underline{\alpha}^{(i)} \leq_1 \Delta(a_k) + \underline{\beta}_g + \underline{\lambda} - \underline{\alpha}^{(j)} = \Delta(a_k) + \Delta_{\underline{\alpha}}(\mathbf{g}_k) \leq_1 \underline{\lambda}$. Since $\mathrm{lt}(f^{(r)}\mathbf{e}_r) \leq \mathrm{lt}(\mathbf{f})$ for all $1 \leq r \leq L$, $\Delta(f^{(r)}) + \underline{\lambda} - \underline{\alpha}^{(r)} \leq_1 \underline{\beta}_f + \underline{\lambda} - \underline{\alpha}^{(i)} \leq_1 \underline{\lambda}$ and hence $\Delta(f^{(r)}) \leq_1 \underline{\alpha}^{(r)}$. Therefore $\mathbf{f}$ is a solution of the required type.

Conversely, suppose $\mathbf{f} \in S$ satisfies (2.3). The standard representation is $\mathbf{f} = \sum_{k \in K} a_k \mathbf{g}_k$ for some $K \subseteq \{1, \ldots, m\}$, where $\mathrm{lt}(a_k \mathbf{g}_k) \leq \mathrm{lt}(\mathbf{f})$ for all $k \in K$. Now $\mathrm{lt}(a_k \mathbf{g}_k) = \mathrm{lt}_1(a_k)\mathrm{lt}(\mathbf{g}_k)$, so $\mathrm{lt}_1(a_k)\mathrm{lt}(\mathbf{g}_k) \leq \mathrm{lt}(\mathbf{f})$. With the notation of the previous paragraph, this gives $\Delta(a_k) + \Delta_{\underline{\alpha}}(\mathbf{g}_k) \leq_1 \underline{\beta}_f + \underline{\lambda} - \underline{\alpha}^{(i)}$. Since, by assumption, $\underline{\beta}_f \leq_1 \underline{\alpha}^{(i)}$, the result follows. □

Corollary 2.2.5 *Suppose that $\mathcal{G}$ is a strictly ordered Gröbner basis of S. There exists a solution satisfying (2.3) if and only if $\Delta_{\underline{\alpha}}(\mathbf{g}_1) \leq_1 \underline{\lambda}$, and in that case $\mathbf{g}_1$, the minimal element, is such a solution.*

We can illustrate this result by reference to the solution of the "key equation" in Fitzpatrick [22]. Let $A = F[x]$. Given $g \in A$ with $\partial g \leq n-1$ and non–negative integers ℓ, m a solution $(a, b) \in A^2$ is sought for

$$a \equiv bg \bmod x^n$$

with $\partial a \leq \ell$, $\partial b \leq m$ and $\ell + m < n$. When $\ell \leq m$, $\lambda = \ell$ and the weights are $(0, \ell - m)$. On the other hand, if $m \leq \ell$, $\lambda = m$, the weights are $(m - \ell, 0)$. In either event, by choosing the tie-breaker function $\mathrm{tb}(i, j) = i$ when $i \leq j$, the resulting term order is equivalent to that used in [22].

2.3 Previous coding and systems applications

The generalised algorithm (Chapter 3) can be applied to a wide range of problems. In this section we review previous work by Fitzpatrick on decoding alternant codes, on scalar partial realisation and rational interpolation and on solving multivariable polynomial congruences. These problems lead to congruences of form

$$a \equiv \sum_{i=2}^{L} b_i h_{ik} \bmod I_k, 1 \leq k \leq p$$

where I_k are ideals in $F[x_1, \ldots, x_s]$. The input data are provided by the polynomials $h_{ik} \in A$ and the ideals on the right hand side of these congruences are derived from "order" constraints and interpolation points. The task then is to find $a, b_2, \ldots, b_L \in A$.

The following lemma provides a slightly more general setting for the problems in this section (i.e. with $a = b_1$ and $h_{1k} = -1, 1 \leq k \leq p$).

Lemma 2.3.1 *Consider polynomials $h_{ik} \in A, 1 \leq i \leq L, 1 \leq k \leq p$ and ideals $I_k \subseteq A, 1 \leq k \leq p$. The set $M = \{(b_1, \ldots, b_L) \in A^L | \sum_{i=1}^{L} b_i h_{ik} \equiv 0 \bmod I_k, 1 \leq k \leq p\}$ is a submodule of A^L.*

PROOF. For any $k, 1 \leq k \leq p$, let $M_k = \{(b_1, \ldots, b_L) \in A^L | \sum_{i=1}^{L} b_i h_{ik} \equiv 0 \bmod I_k\}$. Suppose $\mathbf{f} = (f_1, \ldots, f_L)$ and $\mathbf{g} = (g_1, \ldots, g_L) \in A^L$. For any $a, b \in A$ we have

$$\sum_{i=1}^{L} (af_i + bg_i)h_{ik} = a \sum_{i=1}^{L} f_i h_{ik} + b \sum_{i=1}^{L} g_i h_{ik} \equiv 0 \bmod I_k$$

We have $a\mathbf{f} + b\mathbf{g} \in M_k$ and thus M_k is an A-module. Now $M = \cap_{k=1}^{L} M_k$ and so M is also an A-module. □

Thus any Gröbner basis of M can describe the solution set (module). Algorithms for generating the Gröbner bases were presented and proven on a case by case basis. The general algorithm subsumes all of these cases and establishes their validity in a general and more straightforward way.

If, further, the solution has degree constraints, a term order can be chosen so that a Gröbner basis, with respect to that order, can identify the solution set (which will not usually be a module). Of particular interest are situations where there is a unique "required solution" that can be identified as the minimal element of the solution module with respect to a certain term order defined by the problem. Such an element must lie in a Gröbner basis with respect to that term order. In the original works, an individual term order was crafted to address each application under study. Each of these can be derived directly by using the results of Section 2.2. Therefore the following applications are special cases of our general approach.

2.3.1 Decoding alternant codes

The specific context of decoding was addressed in [22, 24]. The errors-only case corresponds to the solution of the single congruence

$$b_1(-1) + b_2 h \equiv 0 \bmod x^n$$

subject to the conditions $\partial b_1 \leq m_1, \partial b_2 \leq m_2, m_1 + m_2 < n$ and b_1, b_2 relatively prime. This is a partial realisation problem in which a required solution is shown to be minimal with respect to a certain term order defined relative to the parameter $r = m_1 - m_2$. The algorithm produces a Gröbner basis of the solution module containing the required solution. The generalisation to errors-and-erasures decoding involves the solution of the congruence subject to the further condition that b_2 be divisible by a fixed polynomial f. In this case initialisation is at the basis $\{(x^{\partial f + r + 1}, 0), (\overline{fh}, f)\}$, where $\overline{fh}$ is the remainder of fh modulo $x^{\partial c + r + 1}$.

2.3.2 Partial realisation, scalar rational interpolation

Both of these problems can be viewed as the parameterisation of the solutions of the system of congruences

$$b_1(-1) + b_2 h_k \equiv 0 \bmod (x - \beta_k)^{N_k}, k = 1, \ldots, p$$

where $h_k = \sum_{t=0}^{N_k - 1} c_{kt}(x - \beta_k)^t$. Thus, $L = 2$ and $h_{1k} = -1, h_{2k} = h_k$ for all k. Various conditions may be imposed on the solutions, such as, $\partial b_1 < \partial b_2, \partial b_1 + \partial b_2 < N = \sum_{k=1}^{p} N_k, b_1, b_2$ relatively prime, $b_2(\beta_k) \neq 0$ for all k, and so on. These are used to define an appropriate term order for the solution module and in certain situations to identify a unique required solution as the minimal element in the module. The partial realisation problem corresponds to the

case $p = 1$ (and $\beta_1 = 0$), while the distinct-abscissae rational interpolation problem corresponds to $N_k = 1$ for all k. See [23] for more details.

2.3.3 Multiple indeterminates

The main problem analysed in [26] is the determination of a Gröbner basis of the solution module of the congruence

$$b_1(-1) + \sum_{j=2}^{L} b_j h_j \equiv 0 \bmod I$$

where $I \subseteq F[x_1, \ldots, x_s]$ is a zero dimensional ideal. Thus $p = 1$ in this case. This extends previous work on Padé approximation in Fitzpatrick and Flynn [21].

A recursive algorithm is given that applies in the special case where a sequence of approximating ideals can be defined with the property that for each l there is a term $\phi_l \notin I_{l+1}$ such that

$$I_l = \langle \phi_l, I_{l+1} \rangle \text{ and } x_i \phi_l \in I_{l+1}, i = 1, \ldots, s$$

Applications to multivariable Padé approximation, Hensel codes, and algebraic geometry codes are given.

Recently, Little *et al.* [38] shows how some of the conditions from [21] can be relaxed. For a specific term order, a Padé approximation can be found among the elements of any Gröbner basis with respect to this order. This contrasts with [21] which has more restricted conditions and where the solution is the minimal element of a Gröbner basis .

Chapter 3

The general algorithm

This chapter presents the results which form the theoretical foundation for the applications in subsequent chapters. We describe a generalised problem and provide a correspondingly general algorithm which produces a Gröbner basis for the solution module of the problem. The general algorithm is valid for any term order. Since a polynomial ring can be viewed as a module over itself, what follows for A-modules also applies to polynomial rings and ideals thereof.

Based on an idea first presented in [22], the solution is approximated by a descending chain of modules. The success of this method is predicated on the ability to generate a suitable Gröbner basis for each module in the chain from that of the previous one. It is this incremental step which is the core of the algorithm. From a known Gröbner basis for an initial module, repeated application of the incremental step leads to the desired solution. Because, in a given case, it affords an amount choice as to which descending chain is chosen, the algorithm has useful flexibility and a recursive/iterative nature.

We examine the applicability and efficiency of the general algorithm. We go on to consider the single indeterminate case from the standpoint of sim-

plification and complexity.

3.1 The incremental step

We first describe the incremental step of our algorithm in the following theorem. It will be applicable to submodules $M_\ell, M_{\ell+1}$ of an A-module M, where $M_\ell \supset M_{\ell+1}$, and there exists an F–homomorphism

$$\theta_\ell : M_\ell \to F, \ker(\theta_\ell) = \mathrm{M}_{\ell+1}. \tag{3.1}$$

This function returns the "discrepancy" between an element of M_ℓ and the submodule $M_{\ell+1}$.

We begin with a lemma.

Lemma 3.1.1 *Let $M_\ell \supset M_{\ell+1}$ and let θ_ℓ satisfy* (3.1)*. Then there exist constants β_i such that*

$$(x_i - \beta_i)M_\ell \subseteq M_{\ell+1}, 1 \leq i \leq s. \tag{3.2}$$

Further β_i is unique and $\beta_i = \theta_\ell(x_i \boldsymbol{f}_\ell)$, where $\boldsymbol{f}_\ell$ is the unique element, modulo $M_{\ell+1}$, with $\theta_\ell(\boldsymbol{f}_\ell) = 1$.

Proof. Let $\mathbf{f} \in M_\ell \backslash M_{\ell+1}$. Then $\theta_\ell(\mathbf{f}) = \delta$, where $\delta \neq 0$. Define $\mathbf{f}_\ell$ to be $\frac{1}{\delta}\mathbf{f}$ so that $\theta_\ell(\mathbf{f}_\ell) = 1$. Suppose $\theta_\ell(\mathbf{g}) = 1$ for some $\mathbf{g} \in M_\ell$. Now $\theta_\ell(\mathbf{f}_\ell - \mathbf{g}) = 0$ and so $\mathbf{f}_\ell - \mathbf{g} \in M_{\ell+1}$. This means that $\mathbf{f}_\ell$ is unique modulo $M_{\ell+1}$.

Let $\beta_i = \theta_\ell(x_i \mathbf{f}_\ell), 1 \leq i \leq s$. For any $\mathbf{h} \in M_\ell$, we claim $(x_i - \beta_i)\mathbf{h} \in M_{\ell+1}$. Since $M_{\ell+1}$ is an A-module, $(x_i - \beta_i)\mathbf{h} \in M_{\ell+1}$ for $\mathbf{h} \in M_{\ell+1}$. Suppose $\mathbf{h} \in M_\ell \backslash M_{\ell+1}$. Thus, $\theta_\ell(\mathbf{h}) = \gamma, \gamma \neq 0$. It follows that $\theta_\ell(\mathbf{h} - \gamma \mathbf{f}_\ell) = 0$ and $\mathbf{h} - \gamma \mathbf{f}_\ell = \mathbf{m}$ for some $\mathbf{m} \in M_{\ell+1}$ and that $(x_i - \beta_i)\mathbf{h} = (x_i - \beta_i)(\mathbf{m} + \gamma \mathbf{f}_\ell) = (x_i - \beta_i)\mathbf{m} + \gamma(x_i - \beta_i)\mathbf{f}_\ell$. It then follows that $\theta_\ell((x_i - \beta_i)\mathbf{h}) = \gamma\theta_\ell((x_i - \beta_i)\mathbf{f}_\ell) = 0$ and $(x_i - \beta_i)\mathbf{h} \in M_{\ell+1}$.

Finally, β_i is unique. Suppose there is a $\beta_i^{'}$ such that for any $\mathbf{h} \in M_\ell \backslash M_{\ell+1}$,$(x_i - \beta_i^{'})\mathbf{h} \in M_{\ell+1}$. Now, $\theta_\ell((x_i - \beta_i)\mathbf{h}) = 0$ and $\theta_\ell((x_i - \beta_i^{'})\mathbf{h}) = 0$. Thus $\theta_\ell(x_i\mathbf{h}) = \beta_i\theta_\ell(\mathbf{h})$ and $\theta_\ell(x_i\mathbf{h}) = \beta_i^{'}\theta_\ell(\mathbf{h})$. Since $\theta_\ell(\mathbf{h}) \neq 0$, we have $\beta_i = \beta_i^{'}$. □

If $\mathcal{W}$ is an ordered set, we will denote its j^{th} element by $\mathcal{W}[j]$.

Theorem 3.1.2 *Let M be an A–module and let $M_\ell \supset M_{\ell+1}$ be submodules of M with θ_ℓ satisfying* (3.1) *and $\beta_i, i \in [s]$ as in* (3.2). *Let $H : A^L \to M$ be an F–linear function such that for each $i, 1 \leq i \leq s$ there exists $\gamma_i \in F$ satisfying*

$$H(x_i\mathbf{b}) = (x_i + \gamma_i)H(\mathbf{b})$$

for all $\mathbf{b} = (b_1, \ldots, b_L) \in A^L$. Let $S \subseteq A^L$ be a submodule satisfying

$$H(\mathbf{b}) \equiv 0 \bmod M_\ell \text{ for all } \mathbf{b} \in S \tag{3.3}$$

and let $S' \subseteq S$ be the set of elements satisfying

$$H(\mathbf{b}) \equiv 0 \bmod M_{\ell+1}. \tag{3.4}$$

Then S' is a submodule of A^L.

If $\mathcal{W}$ is a strictly ordered Gröbner basis of S relative to a term order $<$ then a Gröbner basis $\mathcal{W}'$ of S' relative to $<$ can be constructed by the following algorithm

Algorithm 3.1.3 : the incremental step.

Define $\alpha_j := \theta_\ell(H(\mathcal{W}[j]))$ for $1 \le j \le |\mathcal{W}|$

$\mathcal{W}'$=incremental-step($\mathcal{W}, [x_i], [\alpha_j], [\beta_i], [\gamma_i]$)

Proc incremental-step()

If $\alpha_j = 0$ for all j then

$\mathcal{W}' = \mathcal{W}$

otherwise

$j^* :=$ least j for which $\alpha_j \neq 0$

$\mathcal{W}_1 := \{\mathcal{W}_j : j < j^*\}$

$\mathcal{W}_2 := \{(x_i - (\beta_i + \gamma_i))\mathcal{W}[j^*] : 1 \le i \le s\}$

$\mathcal{W}_3 := \{\mathcal{W}[j] - (\alpha_j/\alpha_{j^*})\mathcal{W}[j^*] : j > j^*\}$

$\mathcal{W}_1 \bigcup \mathcal{W}_2 \bigcup \mathcal{W}_3$

End

PROOF. Let $\mathbf{b} \in S' \subseteq S$. Thus $H(\mathbf{b}) \equiv 0 \bmod M_{\ell+1}$. Since S is a module, $x_i\mathbf{b} \in S, 1 \le i \le s$. But $H(x_i\mathbf{b}) = (x_i + \gamma_i)H(\mathbf{b}) \equiv 0 \bmod M_{\ell+1}$. Thus, $X\mathbf{b} \in S'$ for any term X. By the F–linearity of H it follows straightforwardly that S' is a submodule.

By definition, $H(\mathcal{W}[j]) \in M_\ell$, so if $\alpha_j = 0$, for all j, then $\mathcal{W} \subseteq S'$ so $S' = S$. Thus suppose some $\alpha_j \neq 0$ and let j^* be as defined. If $j < j^*$ then clearly $\mathcal{W}[j] \in S'$. Next, $H(\mathcal{W}[j^*]) \in M_l$, so $H((x_i - \gamma_i - \beta_i)\mathcal{W}[j^*]) = (x_i - \beta_i)H(\mathcal{W}[j^*]) \in M_{\ell+1}$, by (3.2), and hence $(x_i - (\beta_i + \gamma_i))\mathcal{W}[j^*] \in S'$.

Finally, for $j > j^*$,

$$\begin{aligned}\theta_\ell(H(\mathcal{W}[j] - (\alpha_j/\alpha_{j^*})\mathcal{W}[j^*])) &= \theta_\ell(H(\mathcal{W}[j]) - (\alpha_j/\alpha_{j^*})H(\mathcal{W}[j^*])) \\ &= \theta_\ell(H(\mathcal{W}[j])) - (\alpha_j/\alpha_{j^*})\theta_\ell(H(\mathcal{W}[j^*])) \\ &= \alpha_j - (\alpha_j/\alpha_{j^*})\alpha_{j^*} = 0\end{aligned}$$

so $\mathcal{W}[j] - (\alpha_j/\alpha_{j^*})\mathcal{W}[j^*] \in S'$ by (3.1). We have now proved that $\mathcal{W}' \subseteq S'$.

We show that $\mathcal{W}'$ is a Gröbner basis as follows. By assumption, $\mathrm{lt}(\mathcal{W}[i]) \neq \mathrm{lt}(\mathcal{W}[j])$, when $i \neq j$. Now, $\mathrm{lt}((x_i - (\beta_i + \gamma_i))\mathcal{W}[j^*]) = x_i\mathrm{lt}(\mathcal{W}[j^*])$ and $\mathrm{lt}(\mathcal{W}[j] - (\alpha_j/\alpha_{j^*})\mathcal{W}[j^*])) = \mathrm{lt}(\mathcal{W}[j]), j > j^*$. Let $\mathbf{f} \in S' \subseteq S$. Then $\mathrm{lt}(\mathbf{f})$ is divisible by some $\mathrm{lt}(\mathcal{W}[j])$. If $j \neq j^*$ then $\mathrm{lt}(\mathbf{f})$ is divisible by the leading term of an element of $\mathcal{W}'$.

Thus, we may suppose that $\mathrm{lt}(\mathcal{W}[j^*])$ is the *only* leading term of the basis elements $\mathcal{W}[j]$ that divides $\mathrm{lt}(\mathbf{f})$. We show that $x_i\mathrm{lt}(\mathcal{W}[j^*])$ also divides $\mathrm{lt}(\mathbf{f})$ for some i. Consider the standard representation $\mathbf{f} = \sum_{j\in J} f_j\mathcal{W}[j]$, with $f_j \neq 0$, and $J \subseteq \{1, \ldots, |\mathcal{W}|\}$. By definition of this representation, and by the assumption on $\mathrm{lt}(\mathcal{W}[j^*])$, it follows that $j^* \in J$, $\mathrm{lt}(\mathbf{f}) = \mathrm{lt}(f_{j^*}\mathcal{W}[j^*])$, and $\mathrm{lt}(f_j\mathcal{W}[j]) < \mathrm{lt}(\mathbf{f})$ for $j \neq j^*$. Let X_j, $j \in J$, be terms in A such that $\mathrm{lt}(f_j\mathcal{W}[j]) = X_j\mathrm{lt}(\mathcal{W}[j])$. Thus $X_j\mathrm{lt}(\mathcal{W}[j]) < X_{j^*}\mathrm{lt}(\mathcal{W}[j^*])$. Suppose that there is some $j \in J$ with $j > j^*$. If $X_{j^*} = 1$ then

$$X_j\mathrm{lt}(\mathcal{W}[j]) < \mathrm{lt}(\mathcal{W}[j^*]) \leq X_j\mathrm{lt}(\mathcal{W}[j^*])$$

which contradicts the increasing order of $\mathcal{W}$. Hence $X_{j^*} \neq 1$, so $x_i\mathrm{lt}(\mathcal{W}[j^*])$ divides $\mathrm{lt}(\mathbf{f})$ for some i. Otherwise, $J \subseteq \{1, \ldots, j^*\}$ and

$$\mathbf{f} - \sum_{j=1}^{j^*-1} f_j\mathcal{W}[j] = f_{j^*}\mathcal{W}[j^*]$$

lies in S'. Therefore $f_{j^*} \neq 1$ since $\mathcal{W}[j^*] \notin S'$. As a consequence, $X_{j^*} \neq 1$ and again $x_i\mathrm{lt}(\mathcal{W}[j^*])$ divides $\mathrm{lt}(\mathbf{f})$ for some i. This completes the proof. $\square$

Remark 3.1.4 Since an ordered minimal Gröbner basis is a strictly ordered Gröbner basis it is clear Theorem 3.1.2 is also valid when $\mathcal{W}$ is an ordered minimal Gröbner basis .

3.2 Applicability of the incremental step

Since the incremental step is central to the algorithm, the conditions on M_ℓ and $M_{\ell+1}$ are very significant to the applicability of the algorithm. We next investigate the implications of (3.1) and (3.2).

The following results examine properties of sets of the form $F\mathbf{f}_\ell + M_{\ell+1}$.

Lemma 3.2.1 *Let $M_\ell = F\boldsymbol{f}_\ell + M_{\ell+1}$, where $M_{\ell+1}$ a proper submodule of M_0 and $\boldsymbol{f}_\ell \in M_0 \backslash M_{\ell+1}$. For any $\boldsymbol{f} = \delta \boldsymbol{f}_\ell + \boldsymbol{m}$, $\boldsymbol{m} \in M_{\ell+1}$, the coefficient δ is well defined.*

PROOF. Suppose also $\mathbf{f} = \gamma \mathbf{f}_\ell + \mathbf{m}^{'}$, $\mathbf{m}^{'} \in M_{\ell+1}$. Then $\mathbf{m} - \mathbf{m}^{'} = (\gamma - \delta)\mathbf{f}_\ell$. Since $\mathbf{m} - \mathbf{m}^{'} \in M_{\ell+1}$, $(\gamma - \delta)\mathbf{f}_\ell \in M_{\ell+1}$ and $(\gamma - \delta) = 0$. Thus $\gamma = \delta$. $\square$

Note that M_ℓ is not necessarily closed under A–multiplication.

Theorem 3.2.2 *Let $M_\ell, M_{\ell+1}$ be A–submodules of M_0 and $M_0 \supseteq M_\ell \supset M_{\ell+1}$. The following statements are equivalent.*

1. *There is an F-homomorphism θ_ℓ with $\theta_\ell : M_\ell \rightarrow F, \ker(\theta_\ell) = M_{\ell+1}$.*
2. *There is an element $\boldsymbol{f}_\ell \in M_\ell \backslash M_{\ell+1}$ such that $M_\ell = F\boldsymbol{f}_\ell + M_{\ell+1}$.*

PROOF. Suppose there is an F-homomorphism θ_ℓ with $\theta_\ell : M_\ell \rightarrow F, \ker(\theta_\ell) =$ $\mathrm{M}_{\ell+1}$. Choose any element $\mathbf{f} \in M_\ell \backslash M_{\ell+1}$. Let $\theta_\ell(\mathbf{f}) = \delta$. Since $\delta \neq 0$, we can define $\mathbf{f}_\ell = \frac{1}{\delta}\mathbf{f}$ and we have $\theta_\ell(\mathbf{f}_\ell) = 1$. Since M_ℓ is a module, $M_\ell \supseteq F\mathbf{f}_\ell + M_{\ell+1}$.

Let $\mathbf{g} \in M_\ell$. If $\mathbf{g} \in M_{\ell+1}$, $\mathbf{g} = 0\mathbf{f}_\ell + \mathbf{g}$. Otherwise, $\mathbf{g} \in M_\ell \backslash M_{\ell+1}$ and $\theta_\ell(\mathbf{g}) = \gamma \neq 0$. Now $\theta_\ell(\mathbf{g} - \gamma\mathbf{f}_\ell) = 0$ and $\mathbf{g} - \gamma\mathbf{f}_\ell = \mathbf{m} \in M_{\ell+1}$. Therefore $\mathbf{g} = \gamma\mathbf{f}_\ell + \mathbf{m}$ and $M_\ell \subseteq F\mathbf{f}_\ell + M_{\ell+1}$.

Conversely, suppose that there is $\mathbf{f}_\ell \in M_\ell \backslash M_{\ell+1}$ so that $M_\ell = F\mathbf{f}_\ell + M_{\ell+1}$. Let $\mathbf{g} = \delta\mathbf{f}_\ell + \mathbf{m}, \delta \in F, \mathbf{m} \in M_{\ell+1}$ be any element of M_ℓ. Define $\theta_\ell(\mathbf{g}) = \delta$. This function is well defined by Lemma 3.2.1. Clearly, θ_ℓ is an F-homomorphism. If $\theta_\ell(\mathbf{g}) = 0$, then $\mathbf{g} = \mathbf{m} \in M_{\ell+1}$. If $\theta_\ell(\mathbf{g}) = \delta \neq 0$, then $\delta\mathbf{f}_\ell = \mathbf{g} - \mathbf{m}$ and $\mathbf{g} \notin M_{\ell+1}$. Thus $\ker(\theta_\ell) = \mathrm{M}_{\ell+1}$.

□

These two results lead to a useful way to define the function θ_ℓ.

Corollary 3.2.3 *If there is an element $\boldsymbol{f}_\ell \in M_\ell \backslash M_{\ell+1}$ such that $M_\ell = F\boldsymbol{f}_\ell + M_{\ell+1}$ is an A-module, then the function θ_ℓ defined by $\theta_\ell(\delta\boldsymbol{f}_\ell + m) = \delta \in F$ satisfies* (3.1).

When the modules under consideration are contained in a free module we can provide further insight in the context of terms.

Lemma 3.2.4 *Let $M_{\ell+1} \subset M_\ell = F\mathbf{f}_\ell + M_{\ell+1}$ be modules contained in A^L. For an arbitrary term order $<$ on A^L, there is an element $\mathbf{f}_\ell{}^*$ such that $M_\ell = F\mathbf{f}_\ell{}^* + M_{\ell+1}$ and no element of $\mathcal{LT}(M_{\ell+1})$ is in the support of $\mathbf{f}_\ell{}^*$. In particular,* $\mathrm{lt}(\mathbf{f}_\ell{}^*)$ *depends only on $<$.*

Proof. Let $\mathcal{G}$ be a Gröbner basis of $M_{\ell+1}$ with respect to $<$. We can reduce $\mathbf{f}_\ell$ with respect to $\mathcal{G}$ to yield $\mathbf{f}_\ell{}^*$. Thus $\mathbf{f}_\ell{}^*$ does not have any terms which are in $\mathcal{LT}(M_{\ell+1})$.

Suppose there is another reduced element $\mathbf{g}$ such that $M_\ell = F\mathbf{g} + M_\ell$. By definition, $\mathrm{lt}(\mathbf{f}_\ell{}^*) \notin \mathcal{LT}(M_{\ell+1})$ and $\mathrm{lt}(\mathbf{g}) \notin \mathcal{LT}(M_{\ell+1})$, but $\mathbf{g} = \mathbf{f}_\ell{}^* + \mathbf{m}_g$, where $\mathbf{m}_g \in M_{\ell+1}$. Thus $\mathrm{lt}(\mathbf{g}) = \mathrm{lt}(\mathbf{f}_\ell{}^*)$. □

In this case, for $\mathbf{g} \in M_\ell$, $\theta_\ell(\mathbf{g})$ can be defined as returning the coefficient of $\mathrm{lt}(\mathbf{f}_\ell^*)$ in the normal form of $\mathbf{g}$ with respect to M_ℓ for that term order.

3.3 The general problem

For an A-module $T^{(0)} \subseteq A^L$, elements $\mathbf{b} = (b_1, \ldots, b_L) \in T^{(0)}$ are sought which satisfy a sequence of p congruences

$$H^{(k)}(\mathbf{b}) \equiv 0 \bmod M^{(k)}, k = 1, \ldots, p \tag{3.5}$$

where $M^{(k)}$ are A-modules. Each $H^{(k)}$ is an F–linear function such that for each $i, 1 \leq i \leq s$, there exists $\gamma_i^{(k)} \in F$ satisfying

$$H^{(k)}(x_i\mathbf{b}) = (x_i + \gamma_i^{(k)})H^{(k)}(\mathbf{b}) \tag{3.6}$$

for all $\mathbf{b} \in A^L$. Following Theorem 3.1.2, the solution set T is a submodule of A^L.

The most general published form of the algorithm is to be found in [48]. This, in turn, generalised [47]. It is a generalisation of number of problems in coding theory and systems theory [22, 23, 25, 26]. The general algorithm provides a unified approach to all of these problems while at the same time leading to a simpler theoretical justification.

3.3.1 The general algorithm

Our general algorithm is applicable providing that for each for each $M^{(k)}$ we have a (descending) chain of modules

$$M_0^{(k)}, \ldots, M_\ell^{(k)}, \ldots, M_{N_k}^{(k)} = M^{(k)}$$

and $F-$homomorphisms $\theta_\ell^{(k)}$ so that for each ℓ

$$M_\ell^{(k)} \supset M_{\ell+1}^{(k)}$$

$$\theta_\ell^{(k)} : M_\ell^{(k)} \to F, \ker(\theta_\ell^{(\mathrm{k})}) = \mathrm{M}_{\ell+1}^{(\mathrm{k})}. \tag{3.7}$$

From Lemma 3.1.1, we have suitable constants $\beta_i^{(k,\ell)}$ where

$$(x_i - \beta_i^{(k,\ell)})M_\ell^{(k)} \subseteq M_{\ell+1}^{(k)}, 1 \leq i \leq s \tag{3.8}$$

We may approach the solution module by a sequence of approximations, as follows. Suppose that the functions $H^{(k)}$ satisfy the conditions of Theorem 3.1.2. Let $(j_1, \ldots, j_p) \in \{1, \ldots, N_1\} \times \cdots \times \{1, \ldots, N_p\}$. We define $T_{(j_1,\ldots,j_p)}$ to be the submodule satisfying

$$H^{(k)}(\mathbf{b}) \equiv 0 \bmod M_{j_k}^{(k)} \text{ for all } k \in \{1, \ldots, p\}. \tag{3.9}$$

Let $T^{(0)} = T_{(0,\ldots,0)}$, $T = T_{(N_1,\ldots,N_p)}$. We will define a descending chain of modules $T^{(0)} \supseteq \cdots \supseteq T$. Beginning with an initial strictly ordered Gröbner basis for $T^{(0)}$ we will use the incremental step to determine a Gröbner basis for the next module in the sequence. If $j_k \leq j'_k$ for all $k \in \{1, \ldots, p\}$ then $T_{(j_1,\ldots,j_p)} \supseteq T_{(j'_1,\ldots,j'_p)}$.

Now suppose that we have a strictly ordered Gröbner basis for $T^{(i)} = T_{(j_1,\ldots,j_p)}$. Then, providing $j'_k = j_k + 1$ for exactly one $k \in \{1, \ldots, p\}$, and $j'_k = j_k$ otherwise, the incremental step provides a Gröbner basis for $T^{(i+1)} = T_{(j'_1,\ldots,j'_p)}$, where we select $M_\ell = M_j^{(k)}$,$M_{\ell+1} = M_{j+1}^{(k)}$ along with corresponding constants β, γ and the homomorphism θ. The resulting Gröbner basis is then converted into a strictly ordered Gröbner basis by the function *ord*. Thus the solution module T can be approached along different sequences of modules. In Algorithm 3.3.1 the choice of sequence is implemented by the function *nextmod*.

Algorithm 3.3.1

Input

functions $H^{(k)}$

constants $\gamma_i^{(k)}, 1 \leq k \leq p, 1 \leq i \leq s$

modules $M_\ell^{(k)}$ and homomorphisms $\theta_\ell^{(k)}, 1 \leq k \leq p, 0 \leq \ell \leq N_k$

constants $\beta_i^{(k,\ell)}, 1 \leq k \leq p, 1 \leq i \leq s, 0 \leq \ell \leq N_k$

$<$ a term order on A^L

$T^{(0)}$ the initial module

$\mathcal{W}_0$ a strictly ordered Gröbner basis of $T^{(0)}$

Output

$\mathcal{W}$ a strictly ordered Gröbner basis of the submodule T

Main Routine

$\mathcal{W} := \mathcal{W}_0$

For *module* from $T^{(0)}$ to T

(k, θ_ℓ) =nextmod(*module*)

$\alpha_j := \theta_\ell(H^{(k)}(\mathcal{W}[j]))$ for $j \in [|\mathcal{W}|]$

$\mathcal{W}'$=incremental-step($\mathcal{W}, [x_i], [\alpha_j], [\beta_i^{(k,\ell)}], [\gamma_i^{(k)}]$)

$\mathcal{W} := \text{ord}(\mathcal{W}')$

Suppose $H^{(k)}$ is an A-homomorphism, then $H^{(k)}$ is F-linear and $H^{(k)}(x_i\mathbf{b}) = x_i H^{(k)}(\mathbf{b})$. Thus $H^{(k)}$ satifies the requirements of the algorithm with $\gamma_i = 0, 1 \leq i \leq s$.

A fortiori, the algorithm applies to functions described in Lemma 2.3.1. Define $H^{(k)}((b_1, \ldots, b_L)) = \sum_{i=1}^{L} b_i h_{ik}$. Then $H^{(k)} : A^L \to A$ is an A-homomorphism. This was the special case studied in [47].

Because the functions $H^{(k)}$ can be used to shift the origin of a problem (with non zero γ_i), the algorithm is also applicable to interpolation problems.

The Gröbner basis produced gives a basis for the entire solution submod-

ule. However, further refinements are possible when only particular subsets of the solution module are required. We first state an obvious result.

Lemma 3.3.2 *At each iteration of the general algorithm, there is no decrease in the leading term of any element of the Gröbner basis.*

Corollary 3.3.3 *If the solution sought is for those elements whose leading term does not exceed a given term* $\mathbf{t}$ *then at any iteration all Gröbner basis elements whose leading terms exceed* $\mathbf{t}$ *may be discarded.*

PROOF. Suppose $\mathbf{f}$ is in the solution submodule and $\mathrm{lt}(\mathbf{f}) \leq \mathbf{t}$. Let $\mathcal{G} = \{\mathbf{g}_1, \ldots, \mathbf{g}_m\}$ be the final Gröbner basis . Then $\mathbf{f}$ has a standard representation of the form $\mathbf{f} = \sum_{i=1}^{m} a_i \mathbf{g}_i$ where $a_i \in A$ and $\mathrm{lt}(f_i \mathbf{g}_i) \leq \mathrm{lt}(\mathbf{f})$, $1 \leq i \leq m$. Thus any element whose leading term exceeds $\mathbf{t}$ makes no contribution to $\mathbf{f}$. At each iteration, any element whose leading term exceeds $\mathbf{t}$ is either eliminated later by some run of the *ord* function or results in a superflous element of the final basis. □

The only increase in leading term occurs during the calculation of the elements of the set $\mathcal{W}_2$ and those that exceed the predefined limit may discarded at that point. Thus potentially smaller numbers of Gröbner basis elements need be considered during the iterations of the algorithm, leading to performance improvements. Those applications that satisfy Theorem 2.2.4 are immediate candidates for this refinement.

The Gröbner basis elements described in Corollary 3.3.3 can be seen, in these circumstances, as a generalisation of *d-Gröbner bases* described in [7] and defined only in the context of a polynomial ring. While such bases could be generated from pairs of S-polynomials for homogeneous ideals only, our algorithm allows us to calculate them without such a restriction.

The module sequence

Since M_0 can be represented by Theorem 3.2.2 in the form $F\mathbf{f}_0 + (\ldots(\ldots + M)\ldots)$ we immediately have

Corollary 3.3.4 *If M is such that there is a descending sequence satisfying* (3.1) *and* (3.2)*, then M has a finite codimension in M_0.*

Having a finite codimension is not sufficient to guarantee the existence of a suitable descending chain, as can be seen in the following example.

EXAMPLE 3.3.5 Let $A = \mathbb{R}[x]$ be the ring in a single indeterminate over the real numbers. If M is the ideal $\langle x^2+1\rangle$ then A/M is a vector space with two basis elements, 1 and x. The ideal M is maximal and there is no element $\mathbf{f} \in A\backslash M$ such that $A = F\mathbf{f} + M$. $\Diamond$

A module sequence construction for some submodules of a free module

We now present a partial result for submodules of A^L which leads to the construction of a descending chain of modules and the corresponding functions. First, we have a well-known result (attributed in [17] to Macaulay) expressed in our terminology. Fix a term order $<$. Recall that $\mathcal{LT}(M)$ is the submodule generated by the leading terms of M with respect to $<$.

Theorem 3.3.6 [[17] 15.3] *Let M be a submodule of A^L. The set of terms not in $\mathcal{LT}(M)$ forms a (vector space) basis for A^q/M.*

Theorem 3.3.7 *Let M be a submodule of A^L with a finite codimension and with $M = \mathcal{LT}(M)$. Let $\mathbf{t}_0, \ldots, \mathbf{t}_{N-1}$ be the terms of A^L not in $\mathcal{LT}(M)$ and ordered by some term order $<'$ (not necessarily $<$). Define $M_N = M$ and $M_\ell = F\mathbf{t}_\ell + M_{\ell+1}, 0 \le \ell \le N-1$. Then*

1. *The subsets M_ℓ form a descending chain of submodules satisfying* (3.1) *and* (3.2) *with* $M_0 = A^L$.

2. *The function θ_ℓ can be defined using the coefficient of $\mathbf{t}_\ell$ as in Corollary* 3.2.3*, and moreover $\beta_i = 0$ for $1 \leq i \leq s$.*

PROOF.

1. From Theorem 3.2.2, it is sufficient to show that $M_\ell = F\mathbf{t}_\ell + M_{\ell+1}$ is an A–module. It is obviously closed under addition. Since $M = M_N$ is an A–module we assume inductively that $M_{\ell+1}$ is an A–module. Notice that all terms $\mathbf{t} \in A^L$ with $\mathbf{t}_\ell <' \mathbf{t}$ are contained in $M_{\ell+1}$. Let $a = \sum_{j=1}^{c} a_j X_j \in A$ and $\mathbf{f} = \alpha\mathbf{t}_\ell + \mathbf{m}$ where $\alpha, a_1, \ldots, a_c \in F, \mathbf{m} \in M_{\ell+1}$. Thus $a\mathbf{f} = \sum_{j=1}^{c} a_j X_j(\alpha\mathbf{t}_\ell + \mathbf{m}) = \sum_{j=1}^{c}(\alpha a_j)(X_j\mathbf{t}_\ell) + \mathbf{m}', \mathbf{m}' \in M_{\ell+1}$. If $X_j \neq 1$, $\mathbf{t}_\ell <' X_j\mathbf{t}_\ell$ and so $X_j\mathbf{t}_\ell \in M_{\ell+1}$. Therefore, $a\mathbf{f} \in F\mathbf{t}_\ell + M_{\ell+1}$. Finally, from Theorem 3.3.6, $M_0 = F\mathbf{t}_0 + \ldots + F\mathbf{t}_{N-1} + M = A^L$.

2. From the proof of 3.2.2 this is a valid definition for θ_ℓ. By Lemma 3.1.1, $\beta_i = \theta_\ell(x_i\mathbf{t}_\ell)$ and the coefficient of $\mathbf{t}_\ell$ in $x_i\mathbf{t}_\ell$ is 0. □

We immediately have the following result for $F[x]$.

Corollary 3.3.8 *For an ideal $I = \langle x^n \rangle \subset F[x]$, a module sequence of the required form always exists and the corresponding term sequence is $1, x, \ldots, x^{n-1}$.*

An example from [26] illustrates the case of two indeterminates.

EXAMPLE 3.3.9 Consider $A = F[x, y]$ and the ideal $I = \langle x^2, xy, y^2 \rangle \subset F[x, y]$ generated by terms of fixed total degree 2. Choose $<$ to be degree lexicographic order with $x < y$. Now $\mathcal{LT}(I) = I$ and the terms not in $\mathcal{LT}(I)$ are $1, x, y$. Let $I = I_3 = \langle x^2, xy, y^2 \rangle$. Then $I_2 = Fy + \langle x^2, xy, y^2 \rangle = \langle x^2, y \rangle$ and θ_2 returns the coefficient of y. Next, $I_1 = Fx + \langle x^2, y \rangle = \langle x, y \rangle$ and θ_1

returns the coefficient of x. Finally, $I_0 = F1 + \langle x, y\rangle = \langle 1\rangle = F[x, y]$ and θ_0 returns the coefficient of 1. $\Diamond$

Initialisation

A situation, arising frequently in applications, occurs when $M_0^{(k)}$ is the image of $H^{(k)}$ for $k \in \{1, \ldots, p\}$. Then $T^{(0)} = A^L$ and the standard basis vectors of A^L, ordered with respect to $<$, is a suitable initial basis.

In other situations, initial modules arise naturally. For example, in erasures decoding [24], the problem is to find $a, b \in F[x]$ such that for an erasure locator polynomial $c \in F[x]$ and a syndrome polynomial g

$$a \equiv bg \bmod x^n \quad and \quad c|b. \tag{3.10}$$

The submodule of A^2 generated by $\{(1, 0), (0, c)\}$ can be viewed as the pairs of polynomials (a, b) where $c|b$. Using this as the initial module, a Gröbner basis for the solution module with respect to $<$ $(=<_{(1,0)})$ is sought. When $\partial c \geq 1$, $\{(1, 0), (0, c)\}$ can be used as an initial Gröbner basis . The solution can then be found after n iterations in the algorithm.

The erasures case provides us with an example of another phenomenon. Because the behaviour of the algorithm is predictable up to the ∂c^{th} iteration, the ∂c^{th} basis can be written down directly as $\{(x^{\partial c}, 0), (\overline{cg}, c)\}$,where $\overline{cg} \equiv cg \bmod x^{\partial c}$. In this way the algorithm can be shortened to $n - \partial c$ iterations.

3.3.2 Complexity

The general algorithm solves a wide variety of problems. One perspective on our algorithm is that it converts these problems to linear systems of homogeneous equations. Next, it solves the linear system and then presents the

results in the original context.

An A-module can be viewed as a vector space over the field F. Since they are F-linear, the functions

$$H^{(k)} : A^L \rightarrow M_0^{(k)} = F\mathbf{f}_0 + \ldots + F\mathbf{f}_{N_k-1} + M^{(k)}$$

behave as vector space homomorphisms. Because we are seeking solutions $\mathbf{b} \in A^L$ for which $H^{(k)}(\mathbf{b}) \equiv 0 \bmod M^{(k)}$, we can confine our analysis to the vector space homomorphisms

$$H_M^{(k)} : A^L \rightarrow M_0^{(k)}/M^{(k)}$$

where $H_M^{(k)}(\mathbf{b}) = H^{(k)}(\mathbf{b}) \bmod M^{(k)}$. From Corollary 3.3.4, the dimension of the range of $H_M^{(k)}$ is N_k, the codimension of $M^{(k)}$. The number of iterations is bounded by the sum of these codimensions, so the algorithm generates a finite number of terms among the Gröbner basis elements. Let n be the number of terms required to express the solution set required (where this may have been reduced if Corollary 3.3.3 applies) and let these terms be the basis for an n–dimensional vector space over F. Thus, each $H_M^{(k)}$ can be viewed as a vector space homomorphism between finite-dimensional vector spaces and admits a matrix representation $H_F \in F^{N_k \times n}$. Since we are seeking those elements whose images are zero, H_F corresponds to a set of N_k homogeneous equations in n variables. The discrepancy is found by multiplying an n-vector by a chosen row of H_F and so for each basis element the discrepancy computation involves $O(n)$ field operations. Since there are p congruences, we can express the overall problem in at most $N = \sum_{k=1}^{p} N_k$ homogeneous linear equations in n variables. To ensure the existence of a non-trivial solution we can assume, from the standpoint of practicality, that $N < n$.

For a given iteration, each Gröbner basis element calculation requires $O(n)$ operations. Since the function *ord* removes elements with duplicate

leading terms (at least) there are, also, at most n basis elements. Finally, there are at most $N < n$ iterations and the *worst case* complexity of the general algorithm is $O(n^3)$. As we shall see later, complexities of $O(n^2)$ are more typical for specific algorithms derived therefrom.

3.3.3 Simplifications in the single indeterminate case

When $A = F[x]$ several simplifications can be made. The terms in $F[x]^L$ are of the form $x^j\mathbf{e}_i, j \geq 0, 1 \leq i \leq L$.

In this case, $s = 1$ and incremental-step$(\mathcal{W}, [x], [\alpha_j], [\beta], [\gamma])$ can be implemented using the following algorithm.

Algorithm 3.3.10 : incremental-step for $F[x]$.

Proc incremental-step()

If $\alpha_j = 0$ for all j then

$\mathcal{W}' = \mathcal{W}$

otherwise

$j^* :=$ least j for which $\alpha_j \neq 0$

$\mathcal{W}_1 := \{\mathcal{W}_j : j < j^*\}$

$\mathcal{W}_2 := \{(x - (\beta + \gamma))\mathcal{W}[j^*]\}$

$\mathcal{W}_3 := \{\mathcal{W}[j] - (\alpha_j/\alpha_{j^*})\mathcal{W}[j^*] : j > j^*\}$

$\mathcal{W}_1 \bigcup \mathcal{W}_2 \bigcup \mathcal{W}_3$

End

Algorithm 3.3.10 has the following properties..

1. It does not change the number of elements in the Gröbner basis produced.

2. The only element whose leading term changes is that with index j^*.

3. The leading term of $\mathcal{W}[j^*]$ is multiplied by x.

4. The position of the leading term of each Gröbner basis element is unchanged.

The function *ord* simplifies to inserting the new element in the correct location among the ordered elements. It will merely discard it if it duplicates the leading term of an existing element.

Corollary 3.3.11 *Each basis produced during the algorithm has at most the number of elements in the initial basis.*

Lemma 3.3.12 *A Gröbner basis $\mathcal{G}$ for a module $M \subseteq F[x]^L$ is a minimal basis if and only if no two basis elements have leading terms in the same position.*

PROOF. If the leading terms of the basis elements are all in different positions then none of them is a multiple of another. Conversely, if two of the leading terms are in the same position then one of them divides the other. □

Corollary 3.3.13 *A minimal Gröbner basis of a submodule of $F[x]^L$ has at most L elements.*

Corollary 3.3.14 *If the initial basis is a minimal Gröbner basis then the algorithm produces a minimal Gröbner basis at each iteration.*

Corollary 3.3.15 *If the initial basis is the ordered standard basis vectors then the algorithm produces a minimal Gröbner basis with L elements at each iteration.*

Thus, if the initial basis is a minimal Gröbner basis the algorithm takes on a regular structure with a fixed basis size at each iteration. This size is known in advance. The implementation of the *ord* function reduces to inserting the new element into the appropriate place in the order. When Corollary 3.3.3 applies, the actual number of elements may fall below L as the element indexed by j^* might be discarded.

Complexity in the single indeterminate case

The size of the Gröbner basis is bounded above by L and at each iteration there are at $O(Ln)$ field operations. This gives an overall worst case complexity of $O(Ln^2)$. In many of the applications the value of L does not change with n, and so their algorithmic complexities are $O(n^2)$.

Chapter 4

Linear systems theory

In this chapter we show how the theory developed in earlier chapters can be applied to a wide variety of problems in linear systems theory. In particular, all of the algorithms produced are special cases of our general algorithm.

We begin by showing that *block Hankel* and *block Toeplitz* systems of linear equations can be viewed as polynomial congruences with specific degree constraints. Our algorithm, combined with our general term order for degree constraints, can be applied immediately to homogeneous congruences and we demonstrate how this approach can be extended to inhomogeneous congruences. In this way both homogeneous and inhomogenous systems of linear equations are solved. Using other degree constraints, we go on to derive algorithms for performing division-related calculations in a finite field or division ring.

Next, we consider the *exact modelling* of *discrete-time behaviours*. A recursive algorithm, equivalent to a previously known one, is evolved which generates all of the models of the behaviours under study. The *partial realisation* problem is solved by extracting a *controllable* model from the overall solution.

Finally, *rational interpolations* are examined. Our algorithm can lend itself to interpolations from two perspectives. In Chapter 5 we use origin-shifting capabilities of the general algorithm. For the purpose of this chapter, we address the interpolation by the choice of module sequences and solve the *matrix interpolation problem.* This culminates by addressing the *extended M-Padé approximation* problem.

Since matrix calculations are not commutative, we will chose to operate with left division and row reduction. Nevertheless, right sided and column oriented operations are equally appropriate.

4.1 Hankel and Toeplitz systems of linear equations

For $r, s \in \mathbb{N}_0$, and matrices $H_i \in F^{p\times m}, i = 0, \ldots, r+s-1$, we consider the *block* Hankel matrix

$$H = \begin{pmatrix} H_0 & H_1 & H_2 & \ldots & H_{s-1} \\ H_1 & H_2 & H_3 & \ldots & H_s \\ H_2 & H_3 & H_4 & \ldots & H_{s+1} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ H_{r-1} & H_r & H_{r+1} & \ldots & H_{r+s-1} \end{pmatrix}$$

(i.e. where H is constant on block antidiagonals).

The corresponding system of equations has sm unknowns and rp equations. Denote the unknowns by the $F^{sm\times 1}$ vector

$$b = \left(b_{s-1}^{(1)}, b_{s-1}^{(2)}, \ldots, b_{s-1}^{(m)}, b_{s-2}^{(1)}, \ldots, b_0^{(1)}, \ldots, b_0^{(m)}\right)$$

and, finally, let $a \in F^{rp\times 1}$, where

$$a = \left(a_0^{(1)}, a_0^{(2)}, \ldots, a_0^{(p)}, a_1^{(1)}, \ldots, a_{r-1}^{(1)}, \ldots, a_{r-1}^{(p)}.\right)$$

The problem is to solve

$$Hb = a \tag{4.1}$$

for b, given H and a.

A *block* Toeplitz matrix has the form

$$T = \begin{pmatrix} H_0 & H_{-1} & H_{-2} & \dots & H_{-s+1} \\ H_1 & H_0 & H_{-1} & \dots & H_{-s+2} \\ H_2 & H_1 & H_0 & \dots & H_{-s+2} \\ \dots & \dots & \dots & \dots & \dots \\ H_{r-1} & H_r & H_{r+1} & \dots & H_{r-s} \end{pmatrix}$$

with matrices $H_i \in F^{p\times m}, i = -s+1, \dots, r-1$. (i.e. where T is constant on block diagonals).

It is well known that such a matrix can be converted to block Hankel form. Let $I_m \in F^{m\times m}$ be the identity matrix. We define a block matrix J_{sm} as

$$J_{sm} = \begin{pmatrix} 0 & 0 & \dots & \dots & I_m \\ 0 & \dots & \dots & I_m & 0 \\ \dots & \dots & I_m & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ I_m & 0 & \dots & \dots & 0 \end{pmatrix}$$

where I_m denotes the identity matrix.

Now $J_{sm}^2 = I_{sm}$, the identity matrix. Further, TJ_{sm} is in block Hankel form. Now $Tb = TI_{sm}b = (TJ_{sm})(J_{sm}b) = a$. Thus, a Toeplitz system can be addressed by solving the corresponding Hankel problem and reversing the order of the unknowns. (The converse is also true because HJ_{sm} has block Toeplitz form). Therefore, we can confine our analyis to Hankel problems.

The solution of a block Hankel system of linear equations can be recast as solving a sequence of ($F[x]$) polynomial congruences. Let

$$a_k = \sum_{\ell=0}^{r-1} a_\ell^{(k)} x^\ell, 1 \le k \le p$$

$$b_j = \sum_{\ell=0}^{s-1} b_\ell^{(j)} x^\ell, 1 \le j \le m$$

$$h_j^{(k)} = \sum_{\ell=0}^{r+s-1} (H_\ell)_{k,j} x^\ell, 1 \le j \le m, 1 \le k \le p.$$

Lemma 4.1.1 *The solution of the Hankel system* (4.1) *is equivalent to finding* $b_j \in F[x], 1 \le j \le m+p$ *such that*

$$\sum_{j=1}^{m} b_j h_j^{(k)} \equiv b_{m+k} + x^{s-1} a^{(k)} \bmod x^{s+r-1}, 1 \le k \le p \tag{4.2}$$

and subject to $\partial b_j < s$ *for* $1 \le j \le m$ *and* $\partial b_{m+k} < s-1$.

Proof. Suppose that b satisfies equation (4.1). By construction, the corresponding polynomials $b_1, \ldots, b_m$ satisfy (4.2) for some polynomials b_{m+k} with $\partial b_{m+k} < s-1$.

Conversely, suppose that $b_1, \ldots, b_{m+p}$ satisfy (4.2) and the degree constraints. Consider the coefficients of x^n on each side of (4.2) for $s-1 \le n < s+r-1$. Since $\partial b_{m+k} < s-1$, b_{m+k} does not contribute to these coefficients. Let $\ell = n-(s-1)$. We then have rp equations of the form

$$\sum_{j=1}^{m} \sum_{i=0}^{s-1} (H_{\ell+i})_{k,j} b_j^{((s-1)-i)} = a_\ell^{(k)},$$

where $0 \le \ell \le r-1$ and $1 \le k \le p$. □

4.1.1 Homogeneous systems

When the system is homogeneous, that is, when $a = \mathbf{0}$, where $\mathbf{0} \in F^{rp\times 1}$ contains all zeros, the polynomial congruences become

$$\sum_{j=1}^{m} b_j h_j^{(k)} \equiv b_{m+k} \bmod x^{s+r-1}, 1 \leq k \leq p$$

subject to $\partial b_j < s$ for $1 \leq j \leq m$ and $\partial b_{m+k} < s-1$. By defining $h_{m+k}^{(k)} = -1$ and $h_{m+i}^{(k)} = 0$ when $k \neq i$, we see this a special case of a set of "homogeneous" polynomial congruences

$$\sum_{j=1}^{L} b_j h_j^{(k)} \equiv 0 \bmod x^{n^{(k)}}, 1 \leq k \leq p \tag{4.3}$$

where $\partial b_j \leq \alpha_j, 1 \leq j \leq m+p = L$.

We can use our general algorithm to solve the congruences (4.3). Corollary 3.3.8 gives the module sequence and the corresponding functions θ_ℓ return the coefficients of x^ℓ. In this case, $\lambda_{(s-1,\ldots,s-1,s-2,\ldots,s-2)} = \max\{s-1,\ldots,s-1,s-2,\ldots,s-2\} = s-1$ and the corresponding weights $(0,\ldots,0,1,\ldots,1)$ lead to a term order (with the tie-breaker function $\mathrm{tb}(i,j) = i$ when $i \leq j$). We can use the final Gröbner basis and Theorem 2.2.4 to identify all $\mathbf{b} \in F[x]^L$ corresponding to the solutions of (4.3) as

$$\mathbf{b} = \sum_{k\in K} a_k \mathbf{g}_k$$

where $K \subseteq \{1,\ldots,m+1\}$, and $a_k, \mathbf{g}_k$ satisfy $\partial a_k + \Delta_{(s-1,\ldots,s-1,s-2)}(\mathbf{g}_k) \leq s-1$, for $k \in K$.

Thus we can use the following algorithm to generate a Gröbner basis for all elements corresponding to solutions of the general "homogeneous" polynomial congruences, and by selecting the appropriate term order we can solve systems of linear equations.

Algorithm 4.1.2

Input:

polynomials $\{h_i^{(k)} : 1 \leq i \leq n_k, 1 \leq k \leq p\}$

weights $(w_1, \ldots, w_L)$

Output:

Gröbner basis $\mathcal{W}$ of the solution module of generalised homogeneous congruences (4.3).

Main Routine:

$\mathcal{W} := \text{ord}(\{\mathbf{e}_1, \ldots, \mathbf{e}_L\})$

For k from 1 to p

For ℓ from 0 to $n_k - 1$

For j from 1 to $|\mathcal{W}|$

α_j:=coefficient of x^ℓ in $\sum_{i=1}^{L} b_i h_i^{(k)}$

$\mathcal{W}'$=incremental-step$(\mathcal{W}, [x], [\alpha_j], [0], [0])$

$\mathcal{W}$:=ord$(\mathcal{W}')$

EXAMPLE 4.1.3 Consider the single congruence

$$b_1(x^2 + 2x + 1) + b_2(x^3 + x^2) \equiv 0 \bmod x^7. \tag{4.4}$$

If we require $\partial b_1 \leq 2$ and $\partial b_2 \leq 3$, then $\lambda = 3$ and the weights are $(1, 0)$. We have $L = 2, p = 1, n_1 = 7$. The ordered Gröbner basis is

$$(x^2, -x - 1), (0, x^5)$$

Now $\Delta_{(2,3)}((x^2, -x - 1)) = 3$, so the only constrained solutions are of the form

$$\gamma(x^2, -x - 1), \gamma \in F.$$

$\Diamond$

4.1.2 Inhomogeneous systems

When $a \neq \mathbf{0}$, our transformation does not lead to F-linear functions $H^{(k)}$ and our algorithm does not immediately apply. However, if we introduce an extra polynomial b_{m+p+1} and restrict $\partial b_{m+p+1} < 1$ (i.e. $b_{m+p+1} \in F$), then all solutions of the inhomogeneous system are contained among the solutions of the congruences

$$(\sum_{j=1}^{m} b_j h_j^{(k)}) + (-1)b_{m+k} + (-x^{s-1}a^{(k)})b_{m+p+1} \equiv 0 \bmod x^{s+r-1}, 1 \leq k \leq p \tag{4.5}$$

subject to $\partial b_j < s$ for $1 \leq j \leq m$, $\partial b_{m+k} < s-1$ and $\partial b_{m+p+1} < 1$. The latter can be found using the weighted term order $<_{(0,\ldots,0,1,\ldots,1,,s-1)}$ with weights (0,. . . ,0,1,. . . ,1,s-1).

It is clear that that solutions to the inhomogeneous system of linear equations are exactly those of the constrained polynomial congruences for which the component $b_{m+p+1} = 1$. Unfortunately, the algorithm may also produce solutions where the value of the $(m+p+1)$-component is zero. Based on Theorem 2.2.4, we have the following characterisations of the required subset of solutions. We will establish these results in the setting of more general "inhomogeneous" polynomial congruences,

$$\sum_{j=1}^{m+p+1} b_j h_j^{(k)} \equiv 0 \bmod x^{n^{(k)}}, 1 \leq k \leq p \tag{4.6}$$

where $\partial b_j \leq \alpha_j, 1 \leq j \leq m+p$ and $\partial b_{m+p+1} \leq 0, b_{m+p+1} \neq 0$.

Theorem 4.1.4 *Let $\{\mathbf{g}_1, \ldots, \mathbf{g}_{m+p+1}\}$ be a Gröbner basis (with respect to the weighted order) produced by our algorithm and let $K \subseteq \{1, \ldots, m+p+1\}$ be such that $\Delta_{(\alpha_1,\ldots,\alpha_{m+p},0)}(\mathbf{g}_k) \leq \lambda$ for all $k \in K$.*

1. *A solution to the inhomogeneous system* (4.6) *exists if and only there is some $j \in K$ such that $\mathbf{g}_j$ is 1 at component $m+p+1$.*

2. *If any solution exists, it can be expressed as*

$$\mathbf{f} = \mathbf{g_j} + \sum_{k \in K, k \neq j} a_k \mathbf{g}_k$$

where $K \subseteq \{1, \ldots, m+p+1\}$, *and* $a_k, \mathbf{g}_k$ *satisfy* $\partial a_k + \Delta_{(\alpha_1,\ldots,\alpha_m,\alpha_{m+p},0)}(\mathbf{g}_k) \leq \lambda$, *for* $k \in K$.

PROOF.

1. Suppose there exists such an element $\mathbf{g}_j$. The leading term of $\mathbf{g}_j$ is $1\mathbf{e}_{m+p+1}$, because, otherwise, the leading term $x^i\mathbf{e}_r$ would be at another component $r < m+p+1$ and, taking into account tb, $i + \lambda - \alpha_r > \lambda$ would mean $i > \alpha_r$. Since $1\mathbf{e}_{m+p+1}$ is the leading term of $\mathbf{g}_j$, it is a solution for the inhomogeneous problem.

Conversely, note that, since the algorithm ensures that the leading terms are at different components for each basis element, any element with leading term $1\mathbf{e}_{m+p+1}$ is unique. Thus no other basis element $\mathbf{g}_k, k \in K$ has a non-zero value in location $(m+p+1)$. Therefore, there must be an element $\mathbf{g}_j$ with leading term $1\mathbf{e}_{m+p+1}$ in order for a solution to exist. Finally, the initial basis element $1\mathbf{e}_{m+p+1}$ has 1 in component $(m+p+1)$. No other basis element produced by iterations of the algorithm has a non-zero $(m+p+1)$-component unless its leading term is greater than $1\mathbf{e}_{m+p+1}$ and thus that component is unchanged.

2. From part 1. we see that, if a solution exists, then $\mathbf{g}_j$ is a solution. Further, it is the only $\mathbf{g}_k$ to contribute to the value in the $(m+p+1)$ -component. □

We have seen that when a solution exists, the elements $\mathbf{g}_k, k \in K, k \neq j$ have zeros in location $(m+p+1)$. When viewed as elements of $F[x]^{m+p}$ they are the solutions of the corresponding homogeneous problem.

Corollary 4.1.5 *When a solution exists, the following statements are equivalent.*

1. *There is a unique solution.*
2. *There is precisely one element in K and this is the index of the minimal element of the Gröbner basis.*
3. *There is no solution to the homogeneous problem.*

EXAMPLE 4.1.6 We examine an inhomogeneous congruence related to the previous example.

$$b_1(x^2+2x+1)+b_2(x^3+x^2) \equiv x^5+x^3+2x^2 \bmod x^7.$$

This is transformed into

$$b_1(x^2+2x+1)+b_2(x^3+x^2)+b_3(-x^5-x^3-2x^2) \equiv 0 \bmod x^7.$$

If we require $\partial b_1 \leq 2$,$\partial b_2 \leq 3$, $\partial b_3 \leq 0$ then $\lambda = 3$ and the weights are $(1,0,3)$. The ordered Gröbner basis is

$$(x^2,-x-1,0),(0,x^2-x+2,1),(0,x^5,0).$$

Now $\Delta_{(2,3,0)}((0,x^2-x+2,1)) = 3$, so the constrained solutions are of the form

$$(0,x^2-x+2,1)+\gamma(x^2,-x-1,0), \gamma \in F.$$

$\Diamond$

Applications

Two recent applications illustrate the situation where the solution is unique.

The first is from Popovici and Fitzpatrick [53]. It presents an efficient hardware architecture for calculations in a finite field. By regarding elements of a finite field of size q^ℓ as polynomials in $A = F_q[x]$ with degree less than ℓ, an algorithm is presented which performs finite field calculations of the form ab/c. In the polynomial ring, the operations are performed modulo an irreducible polynomial f of degree ℓ. If $u = ab/c$ then in the polynomial context

$$uc \equiv ab \bmod f.$$

Thus polynomials u, v, w satisfying

$$uc + vf + wab \equiv 0 \bmod x^{2\ell-1}.$$

are sought such that $\partial u \leq \ell - 1, \partial v \leq \ell - 2$ and $\partial w \leq 0$. Now $\lambda_{(\ell-1,\ell-2,0)} = \ell - 1$. This leads to a weighted term order with weights $(0, 1, \ell - 1)$. Since f is irreducible, there are no u, v where $uc + vf \equiv 0 \bmod x^{2\ell-1}$ with $\partial u \leq \ell - 1, \partial v \leq \ell - 2$. Therefore, the minimal (first) element of the Gröbner basis is the unique solution.

A second application is from a problem discussed in Wolf and Fitzpatrick [70]. They present a method for directly performing division in a factor ring $F[x]/\langle f \rangle$, as compared with inversion followed by multiplication. When the polynomial f is irreducible, $F[x]/\langle f \rangle$ is a field and it can be viewed as a special case of the previous example when $a = 1$. While their algorithm uses a different method, the more general form of their problem where f may be reducible is also amenable to our approach. They seek a polynomial u, such that

$$uc \equiv b \bmod f$$

$\partial b, \partial c, \partial u < \partial f$ and $gcd(c, f) = 1$. We can represent this as

$$uc + vf + wb \equiv 0 \bmod x^{2\partial f - 1}$$

with $\partial u \leq \partial f - 1$,$\partial v \leq \partial f - 2$ and $\partial w \leq 0$. Now $\lambda_{(\partial f-1, \partial f-2, 0)} = \partial f - 1$. This leads to a weighted term order with weights $(0, 1, \partial f - 1)$. Since $gcd(c, f) = 1$, there are no u, v where $uc + vf \equiv 0 \bmod x^{2\partial f-1}$ with $\partial u \leq \partial f - 1, \partial v \leq \partial f - 2$. Therefore, the first element of the Gröbner basis is the unique solution.

4.1.3 Complexity

The complexity of these algorithms is $O((m+p)((m+p)s)^2)$. In the classical Hankel case $m = p = 1$ and the algorithm has complexity $O(s^2)$. Similarly, for constant m, p the complexity is $O(s^2)$.

4.2 Modelling discrete-time behaviours

In recent decades, the use of linear algebraic structures in control theory has been less emphasised for many real-world situations. Nonetheless, algebraic problems continue to be of more than just academic interest [11]. This is particularly true in the context of finite fields, where there are applications to coding theory and digital signal processing.

The concept of *behaviours* was introduced to linear systems theory by Willems [66, 67, 68, 69]. The behavioural approach provides an abstract, unified setting for linear systems and control theory. Traditionally, these theories take a "black box" view, where the inputs and outputs are modelled from an processor-centric perspective. Inputs and outputs are often dependent of the choice of models and/or the equations involved. By contrast, the behavioural approach treats all the signals of a system on the same basis

and focuses on the solution space rather than the defining equations. Some important characteristics such as controllability are seen to be independent of the state-space representation. A *phenomenon* produces elements of a set $\mathbb{U}$, the *universum*. The elements of $\mathbb{U}$ are called *outcomes*. A model $M = \{\mathcal{B}, \mathbb{U}\}$ identifies a subset $\mathcal{B}$ of $\mathbb{U}$ of possible outcomes. This subset is called the *behaviour* of the model.

A *dynamical system* is a triple $\Sigma = (\mathbb{T}, \mathbb{W}, \mathcal{B})$, with $\mathbb{T} \subseteq \mathbb{R}$ the time axis, $\mathbb{W}$ the signal space, and the behaviour $\mathcal{B} \subseteq \mathbb{W}^{\mathbb{T}} (= \mathbb{U})$. In this case a behaviour can be viewed as a set of trajectories through time $\mathbb{T}$ taking values in $\mathbb{W}$ at each time point. A system is *linear* if $\mathbb{W}$ is a vector space (over some field F) and $\mathcal{B}$ is a linear subspace of $\mathbb{W}^{\mathbb{T}}$. It is *discrete* if $\mathbb{T} \subseteq \mathbb{Z}$. A discrete system is *time invariant* if $\sigma^t \mathcal{B} \subseteq \mathcal{B}$, where $\sigma^t(f(t')) = f(t + t')$ is the *backward t-shift*.

The modelling of systems based on observed data arises naturally in the context of behaviours. Suppose we have data $\mathcal{D} \subseteq \mathbb{U}$ and a family of models $\mathcal{M}$. The family $\mathcal{M}$ is called a *model class*. A model $\{\mathcal{B}, \mathbb{U}\}$ is said to be *unfalsified* by $\mathcal{D}$ if $\mathcal{D} \subseteq \mathcal{B}$. If $\mathcal{B}_1 \subseteq \mathcal{B}_2$ the model $\{\mathcal{B}_1, \mathbb{U}\}$ is said to be *more powerful* than $\{\mathcal{B}_2, \mathbb{U}\}$. A model is called the *most powerful unfalsified model* (MPUM) in a model class $\mathcal{M}$ if it is in $\mathcal{M}$, is unfalsified, and is more powerful than any other unfalsified model in $\mathcal{M}$.

Based on an exact modelling framework presented by Antoulas and Willems [6], Antoulas [3] uses a behavioural approach to model discrete-time time series. The data under consideration are time series

$$\mathbf{w}^{(k)} : \mathbb{Z} \to \mathbb{R}^L, L \geq 1, k = 1, \ldots, m$$

where $\mathbf{w}^{(k)}$ is zero until a finite time $-N_k$ in the past and current time is at the origin.

The model class is identified with matrices Θ with L columns whose elements are polynomials in the *forward shift* σ^{-1} (which lead to linear and time-invariant systems). The behavior $B(\Theta)$ corresponding to Θ is $\ker(\Theta) = \{\mathbf{w} \in (F^L)^{\mathbb{Z}_-} : \Theta(\sigma^{-1})\mathbf{w} = 0\}$ and so the model corresponding to Θ is unfalsified if and only if $\mathbf{w}^{(k)} \in \ker(\Theta), k = 1, \ldots, m$. We can refer to a model as Θ without ambiguity.

Equivalently, each $\mathbf{w}$ can be viewed as a (negative) power series, $\sum_{t \in \mathbb{Z}_-} \mathbf{w}_t x^t$, where $\mathbf{w}_t \in F^L$. The forward shift is equivalent to multiplying by x and truncating at the term of degree zero. A time series is in $B(\Theta)$ if and only if the polynomial L-vector $\mathbf{h}^{(k)} = x^{N_k}\mathbf{w}^{(k)}$ satisfies

$$\Theta(x)\mathbf{h}^{(k)} \equiv 0 \bmod x^{N_k+1}, k = 1, \ldots, m.$$

If we view each of the L columns individually and consider a row $(b_1, \ldots, b_L)$ of the matrix Θ, then $\mathbf{w}^{(k)} \in B(\Theta)$ if and only if each row of Θ satisfies

$$\sum_{i=1}^{L} b_i h_i^{(k)} \equiv 0 \bmod x^{N_k+1}, k = 1, \ldots, m. \tag{4.7}$$

This makes the problem amenable to solution by our techniques.

Viewing $\mathbf{b} = (b_1, \ldots, b_L)$ as an element of $F[x]^L$, let M be the module whose elements $\mathbf{b}$ satisfy (4.7). For each k, a sequence of ideals can be chosen as

$$I_{l_k}^{(k)} = \langle x^{l_k} \rangle, l_k = 0, \ldots N_k$$

and the corresponding discrepancy function returns the coefficient of x^{l_k} in $\sum_{i=1}^{L} b_i h_i^{(k)}$ for an element $\mathbf{b} = (b_1, \ldots, b_L)$.

The rows of any unfalsified model Θ can be generated from a Gröbner basis for M with respect to any term order. The term-over-position (TOP) term order in $F[x]^L$ is defined by $x^r\mathbf{e}_i < x^t\mathbf{e}_j$ if $r < t$, or if $r = t$ and $i < j$. Under TOP, the degree of the leading term of a basis element is the *row degree*

of the corresponding matrix row. The initial module is $F[x]^L$ with Gröbner basis $\{\mathbf{e}_1, \ldots, \mathbf{e}_L\}$.The procedure *ord* need only maintain a table giving the degree and position of the leading term for each basis element. These pairs are unchanged by the incremental step except for the degree of the element $\mathcal{W}[j^*]$, and so re-ordering takes the form of inserting the multiple $x\mathcal{W}[j^*]$ into its correct new position. The final basis produced corresponds to a matrix with ordered row degrees, which, by Lemma 3.3.12, is *row reduced*. This result is a MPUM and is a generating system equivalent to that produced by the algorithm in [3]. Every unfalsified model in the model class can be be generated by taking linear combinations of the rows of a generating system.

Up to this point, and in the spirit of the behavioural approach, no distinction has been made between inputs and outputs of a system. Let $\Theta \in \mathbb{R}^{r\times(p+m)}[x]$ describe an *input-output* model with r rows, m inputs and p outputs. Let $\Theta = (N \quad -D)$ with $N \in \mathbb{R}^{r\times m}[x]$ and $D \in \mathbb{R}^{r\times p}[x]$. A time series $\mathbf{w}$ is correspondingly partitioned into inputs $\mathbf{u} \in (\mathbb{R}^m)^{\mathbb{Z}_-}$ and outputs $\mathbf{y} \in (\mathbb{R}^p)^{\mathbb{Z}_-}$. If $\mathbf{w} \in \ker(\Theta)$ then

$$D\mathbf{y} = N\mathbf{u}.$$

A generating system Θ^* has L rows and $\det(\Theta^*) \neq 0$. Since the future trajectories in such a system are completely determined by its history it is called *autonomous*. A system is *controllable* if, regardless of its previous history, it can be steered to any desired future trajectory. Consider $\Theta = (N \quad -D) \in \mathbb{R}^{p\times(p+m)}$ where $N \in \mathbb{R}^{p\times m}$ and $D \in \mathbb{R}^{p\times p}$. In this context, the input-output system Θ is controllable if and only if N and D are (left) coprime. In [6] it was shown that a model Θ is a *controllable minimal complexity unfalsified model* if and only if p is the number of invariant factors of Θ^* which are unity and the sum of its row degrees is minimal. For the

special case of [3], the invariant factors of Θ^* are all shown to be powers of x and, thus, $p = rank(\Theta^*(0))$. A minimal complexity controllable model Θ can be extracted from Θ^* as the first subset of the ordered row reduced basis for which $rank(\Theta(0))$ equals the number of invariant factors of Θ which are unity.

4.2.1 Partial realisation

The outputs of a system at any point can be determined from the inputs and the current state. The *transfer function* maps the inputs of a system to the outputs. A *realisation* is a system chosen from a class of systems such that its transfer function coincides with that under study. Thus a transfer function may be "realised" in a number of ways (e.g. Kailath [34]).

Consider a discrete-time linear system with m inputs u and p outputs y.

$$s_{t+1} = Ax_t + Bu_t$$

$$y_t = Cs_t$$

where $A \in F^{n\times n}, B \in F^{n\times m}$ and $C \in F^{p\times n}$. The system has n *states*. The transfer function of the system in the rational matrix $T = C(xI - A)^{-1}B$. A *minimal realisation* of T is such a triple $\{A, B, C\}$ which has the smallest value of n. An *observable* realisation is one where the states, which are initially zero, can be deduced from a knowledge of the inputs and the outputs. A realisation is *controllable* if the system can be put into a chosen state by adjusting the inputs.

A (left) *matrix fraction description* of a rational transfer function T is a pair of polynomial matrices N, D with $DT = N$. It is *irreducible* if D and N are (left) coprime. A minimal realisation of a transfer function can be derived

from an irreducible matrix fraction description ([34]). A minimal realisation is also observable and controllable. Rosenthal *et al.* [56] and York [71] examine the relationships between behaviours, the notions of controllability and observability, and error correcting codes.

Realisations may be made from the *impulse response* values (Markov parameters) of a discrete-time finite dimensional system. If only the first M Markov parameters are available, the *partial realisation* problem seeks a minimal realisation whose first M Markov parameters coincide with these. For a single-input, single-output system the Markov parameters are field elements $h_0, \ldots, h_{M-1}, \ldots$, and the *scalar partial realisation* problem is to find relatively prime polynomials a, b such that

$$a/b \equiv h \bmod x^M$$

where $h(x) = h_0 + h_1 x + h_2 x^2 + \ldots + h_{M-1} x^{M-1}$. This is treated as a special case of rational interpolation by Antoulas and Anderson [4] and it is also included in Fitzpatrick [23].

Multivariable (multi-input, multi-output) partial realisation is addressed by Dickinson, Morph and Kailath [16] and Antoulas [2]. It can be performed from the perspective of the modelling problem in the previous section and has been so viewed by Kuijper [36].

We illustrate the application of our algorithm in this context using an example of Antoulas ([3, Example B]).

EXAMPLE 4.2.1 We want to determine a generating system and a controllable model of minimum complexity from the following Markov parameters $(A_i, i = 0, \ldots, 4)$ of a three-input, two-output system.

$$A_0 = 0 \qquad A_1 = \begin{pmatrix} 1 & 1 & 1 \\ 2 & 2 & 0 \end{pmatrix} \qquad A_2 = \begin{pmatrix} 2 & 2 & 1 \\ 4 & 4 & 0 \end{pmatrix}$$

$$A_3 = \begin{pmatrix} 4 & 6 & 3 \\ 8 & 10 & 2 \end{pmatrix} \qquad A_4 = \begin{pmatrix} 8 & 11 & 6 \\ 16 & 21 & 7 \end{pmatrix} ..$$

The corresponding matrix of polynomials $h_i^{(k)}$ is

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ x + 2x^2 + 4x^3 + 8x^4 & x + 2x^2 + 6x^3 + 11x^4 & x + x^2 + 3x^3 + 6x^4 \\ 2x + 4x^2 + 8x^3 + 16x^4 & 2x + 4x^2 + 10x^3 + 21x^4 & 2x^3 + 7x^4 \end{pmatrix}$$

and we denote the columns of this matrix $\mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \mathbf{h}^{(3)}$.

Here we can illustrate again the possibility of deriving a closed form solution. A formal implementation of the algorithm allows us to "write down" a Gröbner basis with respect to POT order, defined in this 1-variable situation by $x^r \mathbf{e}_i < x^t \mathbf{e}_j$ if $i < j$ or if $i = j$ and $r < t$. In this setting the operation of the algorithm is predictable in advance and the final result can be derived directly from the input data without any computation. This gives the basis

$$\begin{pmatrix} x^5 & 0 & 0 & 0 & 0 \\ 0 & x^5 & 0 & 0 & 0 \\ 0 & 0 & x^5 & 0 & 0 \\ -x - 2x^2 - 4x^3 - 8x^4 & -x - 2x^2 - 6x^3 - 11x^4 & -x - x^2 - 3x^3 - 6x^4 & 1 & 0 \\ -2x - 4x^2 - 8x^3 - 16x^4 & -2x - 4x^2 - 10x^3 - 21x^4 & -2x^3 - 7x^4 & 0 & 1 \end{pmatrix}.$$

This model is of full rank with two invariant factors equal to 1. The controllable model consists of rows 4 and 5 and the sum of its row indices is 8. The output of our algorithm with respect to TOP order is a full rank (row reduced) model of the same behaviour and will therefore have two invariant factors with value 1.

We apply our algorithm using the TOP term order. After the Markov parameters A_1, A_2, A_3 have been processed we have the basis consisting of the rows of the matrix

$$\begin{pmatrix} \frac{2}{3}x & \frac{2}{3}x & -\frac{2}{3}x & \frac{2}{3}-\frac{2}{3}x & -\frac{2}{3}+x \\ \frac{1}{2}x^2 & \frac{1}{2}x^2 & x^2 & -x+x^2 & \frac{1}{4}x \\ -\frac{1}{3}x+x^3 & -\frac{1}{3}x & -\frac{2}{3}x & \frac{2}{3}-\frac{2}{3}x-x^2 & -\frac{1}{6} \\ x^3 & x^3 & 0 & 0 & -\frac{1}{2}x^2 \\ 0 & 0 & x^3 & -x^2 & \frac{1}{2}x^2 \end{pmatrix}$$

and the controllable model is derived from rows 1 and 3.

The next step is to consider the component $\mathbf{h}_1$ modulo x^5. The α_j are $0, 0, -2, -2, 0$ and this gives $j^* = 3$ and the next basis

$$\begin{pmatrix} \frac{2}{3}x & \frac{2}{3}x & -\frac{2}{3}x & \frac{2}{3}-\frac{2}{3}x & -\frac{2}{3}+x \\ \frac{1}{2}x^2 & \frac{1}{2}x^2 & x^2 & -x+x^2 & \frac{1}{4}x \\ \frac{1}{3}x & \frac{1}{3}x+x^3 & \frac{2}{3}x & -\frac{2}{3}+\frac{2}{3}x+x^2 & \frac{1}{6}-\frac{1}{2}x^2 \\ 0 & 0 & x^3 & -x^2 & \frac{1}{2}x^2 \\ -\frac{1}{3}x^2+x^4 & -\frac{1}{3}x^2 & -\frac{2}{3}x^2 & \frac{2}{3}x-\frac{2}{3}x^2-x^3 & -\frac{1}{6}x \end{pmatrix}.$$

The new controllable model is given by row 1 and row 3 (which was row 4 in the previous basis). When the two remaining series $\mathbf{h}_2, \mathbf{h}_3$ are processed, the final basis is

$$\begin{pmatrix} -\frac{3}{2}x+\frac{1}{2}x^2 & -\frac{3}{2}x+\frac{1}{2}x^2 & \frac{3}{2}x+x^2 & -\frac{3}{2}+\frac{1}{2}x+x^2 & \frac{3}{2}-2x \\ \frac{2}{3}x^2 & \frac{2}{3}x^2 & -\frac{2}{3}x^2 & \frac{2}{3}x-\frac{2}{3}x^2 & -\frac{2}{3}x+x^2 \\ \frac{1}{2}x & \frac{1}{2}x+x^3 & \frac{1}{2}x & -\frac{1}{2}+\frac{1}{2}x+x^2 & \frac{1}{4}x-\frac{1}{2}x^2 \\ -\frac{1}{3}x^2+x^4 & -\frac{1}{3}x^2 & -\frac{2}{3}x^2 & \frac{2}{3}x-\frac{2}{3}x^2-x^3 & -\frac{1}{6}x \\ 0 & 0 & x^4 & -x^3 & \frac{1}{2}x^3 \end{pmatrix}$$

and the controllable model consists of rows 1 and 3 (with respective row degrees 2 and 3). The sum of the row degrees of the controllable model is 5

and, since this is a row reduced matrix, the McMillan degree is also 5 . We can choose the controllable model Θ_{contr} as

$$\begin{pmatrix} -\frac{3}{2}x + \frac{1}{2}x^2 & -\frac{3}{2}x + \frac{1}{2}x^2 & \frac{3}{2}x + x^2 & -\frac{3}{2} + \frac{1}{2}x + x^2 & \frac{3}{2} - 2x \\ \frac{1}{2}x & \frac{1}{2}x + x^3 & \frac{1}{2}x & -\frac{1}{2} + \frac{1}{2}x + x^2 & \frac{1}{4}x - \frac{1}{2}x^2 \end{pmatrix}.$$

Now, adding a polynomial multiple of row 2 of the final basis to any row of Θ_{contr} will not change $\text{rank}(\Theta_{contr}(0))$. As long as that action does not change the row degree, the McMillan degree is unchanged. All minimal controllable models can be parameterised (as in [3]) in the following way,

$$\Theta_{contr}(x, \alpha, \beta, \gamma) = \begin{pmatrix} 1 & \alpha & 0 & 0 & 0 \\ 0 & \beta x + \gamma & 1 & 0 & 0 \end{pmatrix} \Theta^*$$

$\Diamond$

4.3 Rational interpolation

Given n distinct "knots" $\beta_1, \ldots, \beta_n \in F$ and pairs $(\beta_k, \sigma_{k,0}), \ldots, (\beta_k, \sigma_{k,n_k-1}) \in F^2$ for $k = 1, \ldots, n$ and defining the polynomials

$$h_k = \sigma_{k,0} + \sigma_{k,1}(x - \beta_k) + \ldots + \sigma_{k,n_k-1}(x - \beta_k)^{n_k-1} \tag{4.8}$$

a pair of polynomials $a, b \in F[x]$ is sought such that

$$\frac{a}{b} \equiv h_k \bmod (x - \beta_k)^{n_k}, k = 1, \ldots, n$$

where $b(\beta_k) \neq 0, k = 1, \ldots, n$. The corresponding "weak interpolation" problem is to find $a, b \in F[x]$ such that

$$a \equiv bh_k \bmod (x - \beta_k)^{n_k}, k = 1, \ldots, n$$

Fitzpatrick [23] applies Gröbner basis techniques to solve the weak interpolation problem for the scalar case. We now address the matrix version of the problem.

Given n distinct "abscissae" $\beta_k \in F$, and the pairs $(\beta_k, \mathcal{S}_{k,0}), \ldots, (\beta_k, \mathcal{S}_{k,n_k-1})$ where $\mathcal{S}_{k,j} \in F^{p\times m}, j \in [n_k - 1], k \in [n]$, consider the polynomial matrices

$$S^{(k)} = \mathcal{S}_{k,0} + \mathcal{S}_{k,1}(x-\beta_k) + \ldots + \mathcal{S}_{k,n_k-1}(x-\beta_k)^{n_k-1}, k = 1, \ldots, n. \quad (4.9)$$

We seek (left) matrix fraction descriptions $N \in F^{p\times m}[x], D \in F^{p\times p}[x]$ with minimal McMillan degree such that

$$N \equiv DS^{(k)} \bmod (x-\beta_k)^{n_k}, k = 1, \ldots, n.$$

By reorganising, we have the following (matrix) congruence

$$NI_m - DS^{(k)} = \begin{pmatrix} N & -D \end{pmatrix} \begin{pmatrix} I_m \\ S^{(k)} \end{pmatrix} \equiv 0_{p\times m} \bmod (x-\beta_k)^{n_k}, k = 1, \ldots, n.$$

Again, the rows of $(N \quad -D)$ can be viewed as vectors $\mathbf{b} \in A^{p+m}$. A basis for the rowspace of $(N \quad -D)$ can be found from those module elements $\mathbf{b} \in A^{p+m}$ which satisfy

$$\mathbf{b}H^{(k)} \equiv \mathbf{0}_{1\times m} \bmod (x-\beta_k)^{n_k}, k = 1, \ldots, n$$

where $H^{(k)} = \begin{pmatrix} I_m \\ S^{(k)} \end{pmatrix}$. This yields nm congruences of the form

$$\sum_{i=1}^{p+m} b_i h_{ij}^{(k)} \equiv 0 \bmod (x-\beta_k)^{n_k}, j = 1, \ldots, m, k = 1, \ldots, n \quad (4.10)$$

and a basis for the rows of any such $(N \quad -D)$ can be found using our algorithm. We then have a "module" sequence $\langle 1 \rangle \subset \langle x-\beta_k \rangle \subset \ldots \subset \langle (x-\beta_k)^{n_k} \rangle$ and we can define θ_ℓ to return the coefficient of $(x-\beta_k)^\ell$.

With $n = 1$ and $\beta_1 = 0$, the partial realisation problem is a special case. Using TOP as in the previous section, the resulting algorithm is equivalent to one to be found in Antoulas and Willems [6].

Algorithm 4.3.1

Input:

For each $k, 1 \leq k \leq n$

field constant β_k and nonnegative integer n_k

polynomials $h_{ij}^{(k)}, 1 \leq i \leq p, 1 \leq j \leq m$

Output:

Gröbner basis $\mathcal{W}$ of the solution module of the rational interpolation problem (4.10).

Main Routine:

$\mathcal{W} := \{\mathbf{e}_1, \ldots, \mathbf{e}_L\}$

For k from 1 to n

For ℓ from 0 to $n_k - 1$

For i from 1 to m

For j from 1 to L

α_j:=coefficient of $(x - \beta_k)^\ell$ in $\sum_{r=1}^{L} b_r h_{ri}^{(k)}$

$\mathcal{W}'$=incremental-step($\mathcal{W}, [x], [\alpha_j], [\beta_k], [0]$)

$\mathcal{W}$:=ord($\mathcal{W}'$)

We will illustrate this using an example from Antoulas, Ball, Kang and Willems [5].

EXAMPLE 4.3.2 We have $n = 3, p = 2, m = 2, \beta_1 = 0, \beta_2 = 1, \beta_3 = 2, n_1 = n_2 = n_3 = 1, S_{1,0} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, S_{2,0} = S_{3,0} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.

Then $H^{(1)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}, H^{(2)} = H^{(3)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}$.

The initial basis is

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

and after 5 iterations we have the following ordered basis

$$\begin{pmatrix} -1 & -1 & 1 & 1 \\ 0 & -x & x & 0 \\ (x-2)(x-1) & x & x-2 & 0 \\ -(x-1)(x-2) & x(x-2) & -(x-2) & 0 \end{pmatrix}.$$

The discrepancies at the final iteration are

$$\begin{pmatrix} 0 \\ 0 \\ 2 \\ 0 \end{pmatrix}$$

and the final Gröbner basis is

$$\begin{pmatrix} -1 & -1 & 1 & 1 \\ 0 & -x & x & 0 \\ -(x-1)(x-2) & x(x-2) & -(x-2) & 0 \\ (x-2)^2(x-1) & x(x-2) & (x-2)^2 & 0 \end{pmatrix}.$$

A mimimal matrix fraction is contained in rows 1 and 3

$$\begin{pmatrix} -1 & -1 & 1 & 1 \\ -(x-1)(x-2) & x(x-2) & -(x-2) & 0 \end{pmatrix}.$$

All minimal matrix fractions can be found by adding suitable multiples of row 2 to row 1 and row 3. $\Diamond$

4.3.1 The extended M–Padé problem

Beckermann and Labahn [9] introduced the extended M–Padé approximation problem, as a generalization of the Padé and Hermite–Padé problems, to address a wide variety of interpolation and approximation problems in a single variable. They used module theoretic methods to parametrise the set of all solutions of the general problem. Further, they developed a method of recursively generating solution bases that enabled them to break a given problem down into smaller problems of the same type along arbitrary paths. In this section we indicate how our algorithm can be applied to this problem. It is clear that our approach extends beyond the single variable case.

Let $F_0 = \{\beta_1, \ldots, \beta_t\} \subseteq F$ be a set of "knots". Let $F^{p\times L}[[x]]_{F_0}$ be the set of formal Newton series with coefficients in $F^{p\times L}$, that is, $G \in F^{p\times L}[[x]]_{F_0}$ if for all $\beta_j \in F_0$ and all $r \in \mathbb{N}_0$, the r^{th} derivative of G at β_j is known and is an element of $F^{p\times L}$. The set of polynomials with coefficients in $F^{p\times L}$ is a subset of $F^{p\times L}[[x]]_{F_0}$. A vector $\mathbf{u} = (u_1, \ldots, u_L)$ of monic polynomials all of whose zeros are elements of F_0, is called an *order vector* (with respect to F_0). Let $G \in F^{L\times L}[[x]]_{F_0}$. We seek a basis of the set of polynomial vectors $P \in F^{L\times 1}[x]$ satisfying

$$GP = \text{diag}(u_1, \ldots, u_L)W$$

where the *residual* W lies in $F^{L\times 1}[[x]]_{F_0}$. (This can be extended to elements

of $F^{p\times L}[[x]]_{F_0}$, by appending L-p rows which contain arbitrary elements of F and setting u_i to 1 for $L-P+1 \leq i \leq l$).

Setting $G=(g_{ij})$, $P=(p_j)$ and $W=(w_i)$, this becomes

$$\sum_{i=1}^{L} g_{ki}p_i = u_k w_k, k=1,\ldots,p.$$

Since u_k is a product of factors $(x-\beta_j)^{n_j}$ this is equivalent to

$$\sum_{i=1}^{L} g_{ki}p_i \equiv 0 \bmod \langle \prod_{j\in J}(x-\beta_j)^{n_j}\rangle \tag{4.11}$$

for some subset $J \subseteq \{1,\ldots,t\}$. Here g_{k_i}, w_k are Newton series rather than polynomials. Therefore, while (4.11) is suggestive of (4.10), the congruence holds in a ring of Newton series instead of a polynomial ring. It does, however, conform to the requirements of (3.5). The ideal $M^{(k)}=\langle u_k\rangle$ can be regarded as a module over the polynomial ring. The sum of products of Newton series and polynomials on the left hand side can be viewed as a function from A^L into this module.

EXAMPLE 4.3.3 In Example 2.9 from [9] we have $F_0=\{0\}, p=L=3$ and

$$G=\begin{pmatrix} \frac{1}{2}+x^2-x^4 & 1+\sin^4(x^2) & \frac{1}{\sqrt{1+x^2}} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

and $u=(x^8,1,1)$. Row 1 of G leads to the congruence

$$\left(\frac{1}{2}+x^2-x^4\right)p_1+\left(1+\sin^4(x^2)\right)p_2+\left(\frac{1}{\sqrt{1+x^2}}\right)p_3 \equiv 0 \bmod \langle x^8\rangle. \tag{4.12}$$

Since the other two conditions are modulo $\langle 1\rangle$, they are automatically satisfied. $\Diamond$

To apply Algorithm 3.3.1, we observe that in each of the congruences, a typical pair of successive modules in the descending sequence has the form $M_\ell = \langle f \rangle \supseteq \langle (x - \beta_s) f \rangle = M_{\ell+1}$. Thus (3.2) is satisfied. If $g \in M_\ell$ then $g = fw$ for some $w \in F[[x]]_{F_0}$, and expanding w around β_s, we can define θ_ℓ by taking $\theta_\ell(g)$ to be the constant term in the expansion. Thus the problem may be solved using our techniques.

We can now address Example 4.3.3. Let $\beta_s = 0$. As already indicated, rows 2 and 3 of G do not influence the solution and may be ignored for the purposes of calculating a basis. By expanding the elements of row 1 of G as power series, the congruence (4.12) can be expressed as

$$(\frac{1}{2} + x^2 - x^4)p_1 + (1 + x^8 + O(x^{12}))p_2 +$$
$$(1 - \frac{1}{2}x^2 + \frac{3}{8}x^4 - \frac{5}{16}x^6 + \frac{35}{128}x^8 + O(x^{10}))p_3 \equiv 0 \bmod \langle x^8 \rangle.$$

Choosing TOP order, our algorithm produces a Gröbner basis for the solution module which we present as the rows of the matrix

$$\begin{pmatrix} \frac{-10}{19}x^2 + \frac{2}{19} & x^2 - \frac{33}{19} & \frac{32}{19} \\ \frac{9}{19}x^2 - \frac{11}{76} & -\frac{5}{4}x^2 + \frac{59}{152} & x^2 - \frac{6}{19} \\ x^4 + \frac{11}{2}x^2 & -\frac{59}{4}x^2 & 12x^2 \end{pmatrix}.$$

Moreover, we can not only subdivide the problem into subproblems of the same type [9], but also solve any or all them using our algorithm. The ultimate solution can be arrived at exclusively by that algorithm or in combination with other methods. As in [47], the incremental step is much simpler for single variable problems than in the general case, because the position of the leading term of each basis element is not changed, so that the function *ord* has only to move one basis element at each iteration.

However, the types of degree constraint allowed by the methods of [9] are more general than those that we can apply. Essentially, they permit different

constraints in different congruences, whereas our algorithm depends on the constraint being the same throughout. For this special type of constraint we can parametrise all solutions (Theorem 2.2.4).

We conclude this section by indicating some of the theoretical results from Chapter 2 of [9] that are straightforward consequences of our approach. We follow that paper in assuming, without loss of generality, that $p = L$ and view the basis as a matrix whose rows are the basis elements.

We have seen that the solutions which satisfy the order constraints form a submodule of $F[x]^L$. Since there are L elements in the initial (standard basis vectors) basis, there are L elements in each basis (Corollary 3.3.15). This is the core of [9, Theorem 2.6].

We have I_L as the initial solution. If there is a non-zero discrepancy, the incremental step acts on the previous solution by premultiplying it by the matrix of the form

$$\begin{pmatrix} 1 & \dots & 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & x-\beta_k & 0 & \dots & 0 \\ 0 & \dots & \frac{-\alpha_{j^*+1}}{\alpha_{j^*}} & 1 & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \frac{-\alpha_L}{\alpha_{j^*}} & 0 & \dots & 1 \end{pmatrix}$$

and then premultiplying by elementary matrices to effect the reordering. Otherwise, there is no change. Thus the determinant of the basis will be unchanged or will equal that of the previous basis multiplied by $x - \beta_k$. Inductively, we see that the determinant of the final basis divides $\det(\mathrm{diag}(u))$. This yields [9, Theorem 2.7(b)].

4.4 Finite precision effects

Questions of numerical stability can arise when calculations from a field with characteristic 0 are performed in a finite precision enviroment. Polynomial operations are particularly susceptible because the leading term of a module element may be poorly defined if its leading coefficient is smaller than the computational precision. Nonetheless, Fitzpatrick [25] shows that early versions of our algorithm performed well in this regard.

The computational aspects of our algorithm are contained within the incremental step (Algorithm 3.1.3). It has two distinct phases. The first computes the discrepancies of the elements of the incoming Gröbner basis. Recalling the complexity discussion (Section 3.3.2), we note that the discrepancy is the scalar product of an n-vector and a row of an $N_k \times n$ matrix of field elements. The relevant module terms function merely as placeholders for this purpose and no particular location is *a priori* more significant than any other in the calculation. Therefore for this phase the polynomial approach suffers no disadvantage as against a matrix method.

The second phase derives a new Gröbner basis from the discrepancies. The leading terms of the incoming basis elements are significant in this case. For each basis element other than that indexed by j^* the leading term and the its coefficient are unchanged by the incremental step. The element indexed by j^* is replaced by a set of s elements each of which retain the leading coefficient of the former. Thus the incremental step does not change the value of the leading coefficients. For example, if the initial basis is an ordered set of standard basis vectors then the leading coefficient of each Gröbner basis element at every iteration will be 1. Therefore an error in a leading term can only arise from an erroneous value for j^*. This in turn can only be caused by

discrepancy calculations which return values smaller that the computational precision.

Thus any errors in the coefficients of the solutions have their origin in matrix-style calculations. It would be an interesting topic of further research to examine the sensitivity of the algorithm to erroneous values of j^*.

Chapter 5

List decoding applications

This chapter sees the application of our general techniques to *list decoding* of error correcting codes. Sudan's algorithm and its variants use a two stage approach to this problem. The first and most time-consuming step performs an interpolation. For the *hard decision* case, we derive interpolation algorithms for Reed-Solomon and 1–point Algebraic Geometry codes, which are similar to known methods. New algorithms are presented for list decoding of Reed-Solomon and 1–point Algebraic Geometry codes using *soft* information. These have complexity better than standard methods.

5.1 List decoding

The project of coding theory has been to discover practical methods of encoding information which achieve the theoretical reliability and performance promised by Shannon's pioneering work [63]. Information is encoded for transmission and the encoded information, possibly corrupted en route, is decoded on reception. For "real-time" applications such as data storage/retrieval, multimedia or interactive communications, an efficient means

of decoding is crucial. Unrestricted *nearest neighbour* decoding of *linear block* codes takes exponential time. Assuming that the number of errors in a received word does not exceed half the *minimum distance* of the code, a unique closest codeword can be recovered in polynomial time. When the number of errors in a received word exceeds half the minimum distance of the code there may be more than one codeword consistent with the received word. As an alternative to trying to find a particular codeword, the decoder may attempt to generate a *list* of the consistent codewords. Ideally, such a list would be short and have only one element on most occasions. For a list with more than one element a choice can be made among these according to some criterion, perhaps based on supplementary information from the channel, or at the expense of a (hopefully rare) decoding error. This is known as *list decoding* and was introduced by Elias (see [18] and the references given there).

5.1.1 Sudan's Algorithm

A polynomial time technique for list decoding applied to Reed-Solomon codes was invented by Sudan [60]. The essential idea is that a received word is used to create a set of points and a polynomial in two variables interpolating these points is sought. The list of candidate codewords can be found among the factors of the polynomial.

The specific problem addressed in Sudan [60] is as follows. Let F be a field and let $\{(x_i, y_i) \in F^2, i = 1, \ldots, n\}$ be a set of distinct points. Given integers d and t where $t \geq d\lceil\sqrt{2(n+1)/d}\rceil - \lfloor d/2 \rfloor$, Sudan's algorithm determines the set of (univariate) polynomials $f(x)$ of degree at most d with $|\{i|f(x_i) = y_i\}| \geq t$. The first step is to derive a pair of integers ℓ, m which satisfy $(\ell+1)(m+1) + d\binom{m+1}{2} > n$ and $m + \ell d < t$. Then a non-zero polynomial

$Q(x, y)$ is derived, whose $(1, d)$–degree is at most $m+\ell d$, such that $Q(x_i, y_i) = 0$ for all $i = 1, \ldots, n$. This polynomial is guaranteed to exist by the choice of ℓ and m and can be found using standard methods with cubic complexity. The univariate polynomials being sought are among the factors $y - f(x)$ of Q.

5.2 Reed-Solomon codes

The basic theory of Reed-Solomon codes can be found in a number of references such as [42, 51, 52]. Let F_q be the finite field with q elements and let γ be a primitive element of F_q. We can define the Reed-Solomon code of length $N = q - 1$ and dimension $K, 0 \leq K \leq N$ as the subspace

$$\mathcal{C}_q(N, K) = \{(f(1), f(\gamma), \ldots, f(\gamma^{q-2}))|f \in F_q[x], \partial f < K\}.$$

The minimum distance of $\mathcal{C}_q(N, K)$ is $N - K + 1$.

When the number of errors $\tau \leq \lfloor \frac{N-K}{2} \rfloor$ a unique closest codeword can be found [8, 52, 22]. Otherwise, uniqueness cannot be guaranteed. By setting $t = N - \tau$ and $d = K - 1$, Sudan's algorithm can be used to produce a list of consistent codewords when the Reed-Solomon code has rate $\frac{K}{N} < \frac{1}{3}$.

The applicability of Sudan's approach has been enhanced by requiring the polynomial to have zeros of varying multiplicity at the interpolation points [30, 35]. The most general version of the problem (as considered by Kötter and Vardy [35]) has the following form. A polynomial $Q(x, y)$ can be expanded around (x_i, y_i) to give $Q(x, y) = \sum_{j_1} \sum_{j_2} q_{j_1 j_2 i}(x - x_i)^{j_1}(y - y_i)^{j_2}$. It follows that $Q(x + x_i, y + y_i) = \sum_{j_1} \sum_{j_2} q_{j_1 j_2 i} x^{j_1} y^{j_2}$ for certain coefficients $q_{j_1 j_2 i}$. Given a set of points $(x_i, y_i), i = 1, \ldots, n$, and multiplicities m_i, we require the polynomial $Q(x, y)$, minimal with respect to some (a, b)–degree,

such that

$$q_{j_1 j_2 i} = 0 \text{ for } 1 \leq i \leq n \text{ and } j_1 + j_2 < m_i.$$

A number of algorithms to solve this problem can be derived from our general result. For our treatment, the problem translates as finding $Q \in F[x, y]$ so that

$$Q(x + x_i, y + y_i) \equiv 0 \bmod \langle \{x^{j_1} y^{j_2} | j_1 + j_2 = m_i\} \rangle \text{ for } i = 1, \ldots, n. \quad (5.1)$$

To transform this into a problem amenable to solution by our methods, we take (in the notation of Chapter 3) $A = F[x, y], L = 1, H^{(k)} : A \to A, H^{(i)}(Q(x, y)) = Q(x + x_i, y + y_i)$, so that $H^{(i)}(xQ(x, y)) = (x + x_i)Q(x + x_i, y + y_i) = (x + x_i)H^{(i)}(Q(x, y))$, giving $\gamma_1 = \gamma_x = x_i$, and similarly $\gamma_2 = \gamma_y = y_i$. We define $I^{(i)} = \langle \{x^{j_1} y^{j_2} | j_1 + j_2 = m_i\} \rangle$ and so Theorem 3.3.7 applies. Recalling [26], we will choose to work with terms in A in degree lexicographic order and define a descending sequence of $\frac{m_i(m_i+1)}{2}$ ideals $I_r^{(i)}$ between $A = \langle 1 \rangle$ and $I^{(i)}$ corresponding to terms $\varphi_r^{(i)}$, where $I_r^{(i)} = \langle \varphi_r^{(i)}, I_{r+1}^{(i)} \rangle$ and $xI_r^{(i)} \subseteq I_{r+1}^{(i)}, yI_r^{(i)} \subseteq I_{r+1}^{(i)}$ (giving condition (3.2) with $\beta_1 = \beta_x = 0, \beta_2 = \beta_y = 0$). The homomorphism $\theta_r^{(i)}$ returns the coefficient of $\varphi_r^{(i)}$ in the unique representative of an element of $I_r^{(i)}$ modulo $I_{r+1}^{(i)}$, thus satisfying (3.7). For the solution module, we use the term order defined by (a, b)–degree with ties broken by the degree of y (*cf.* Example 2.1.2). The specific polynomial sought in [35] is the minimal element of the solution module with respect to this term order, and, as such, can be identified as the first element in a strictly ordered Gröbner basis of the solution module. This polynomial requires the minimum amount of further processing in order to find the corresponding list of codewords.

More formally, the algorithm iterates, for each point (x_i, y_i), through the terms in degree-lexicographic order (where we choose $x < y$) as far as the

last term with total degree less than m_i.

Algorithm 5.2.1

Input:

points $\{(x_i, y_i) : 1 \leq i \leq n\}$, with multiplicities m_i

Output:

Gröbner basis $\mathcal{W}$ of the solution module of (5.1)

the required Q as the first element of $\mathcal{W}$

Main Routine:

$\mathcal{W} := \{1\}$ (†)

For i from 1 to n

For j_1 from 0 to $m_i - 1$

For j_2 from 0 to j_1

For j from 1 to $|\mathcal{W}|$

α_j:=coefficient of $x^{j_1-j_2}y^{j_2}$ in $\mathcal{W}[j](x + x_i, y + y_i)$

$\mathcal{W}'$=incremental-step($\mathcal{W}, [x, y], [\alpha_j], [0], [x_i, y_i]$)

$\mathcal{W}$:=ord($\mathcal{W}'$)

EXAMPLE 5.2.2 In this example we allow the function *ord* to construct ordered minimal Gröbner bases. With the $(2, 3)$–degree term order described in Example 2.1.2, let $n = 2, (x_1, y_1) = (1, 0), m_1 = 1, (x_2, y_2) = (0, 1), m_2 = 2$. Here we shall write $[f]_{x^u y^v}$ for the coefficient of $x^u y^v$ in f, and indicate only the α_j and $\mathcal{W}$ at each step. The iterations are as follows:

$\underline{(i, j_1, j_2) = (1, 0, 0)}$

$[1]_1 = 1 \rightarrow \{x - 1, y\}$

$\underline{(i, j_1, j_2) = (2, 0, 0)}$

$[x-1]_1 = -1, [y+1]_1 = 1 \rightarrow \{y+x-1, x(x-1)\}$ (here $\mathcal{W}_3 = \{y+x-1\}$ and $(y - 1)(x - 1)$ has been eliminated in the formation of a minimal Gröbner basis)

$\underline{(i, j_1, j_2) = (2, 1, 0)}$

$[y + x]_x = 1, [x(x-1)]_x = -1 \rightarrow \{x^2 + y - 1, x(y + x - 1), (y - 1)(y + x - 1)\}$

$\underline{(i, j_1, j_2) = (2, 1, 1)}$

$[x^2 + y]_y = 1, [x(y + x)]_y = 0, [y(y + x)]_y = 0 \rightarrow \{x(y + x - 1), x(x^2 + y - 1), (y - 1)(y + x - 1)\}$ $\Diamond$

This algorithm begins with a single element in the basis (step (†)), and the number of elements can change with the incremental step. In theory there can be rapid growth in the size of the basis as the computation proceeds. However, the interpolation problems treated in this section all impose upper bounds on the $(0, 1)$-degree of the interpolating polynomials, and on the degrees of terms required to have zero coefficients. We may incorporate these into a sufficiently large global bound, and hence derive algorithms in which the number of basis elements at each iteration is fixed. These will generally be more efficient for implementation purposes. Such degree bounds can also be used, as in 3.3.3, to discard basis elements whose leading terms get too large.

Suppose that each interpolating polynomial satisfying the weighted degree constraint and all relevant terms are contained in $B = \{f \in F[x, y] | \Delta_{(0,1)}(f) < L\}$ for some $L \in \mathbb{N}$. For convenience, we can identify B, as $F[x]$-module, with $F[x]^L$ using,

$$\psi : B \rightarrow F[x]^L,$$

$$\psi(x^{j_1} y^{j_2}) = x^{j_1} \mathbf{e}_{j_2+1}, 0 \leq j_2 \leq L - 1,$$

and extending by linearity. The set $V = \{\mathbf{f} \in F[x]^L | \mathbf{f} = \psi(Q), Q \in B, Q \text{ an interpolating polynomial}\}$ is a submodule of $F[x]^L$. The (a, b)–degree term order defined above can be used in an obvious fashion in $F[x]^L$, where the

degree of $x^{j_1}\mathbf{e}_{j_2}$ is $aj_1 + b(j_2 - 1)$ and ties are broken by $\mathbf{e}_1 < \mathbf{e}_2 < \ldots < \mathbf{e}_L$. We can now use our algorithm to solve the 1–variable minimization problem, by finding a Gröbner basis of V.

Let $A = F[x]$, and define $H^{(i)} : A^L \rightarrow A^L$ by $H^{(i)}(\mathbf{b}) = \psi[\psi^{-1}(\mathbf{b})(x + x_i, y+y_i)]$. Now $\psi^{-1}(x\mathbf{b}) = x\psi^{-1}(\mathbf{b})$ and so $H^{(i)}(x\mathbf{b}) = \psi[(x+x_i)\psi^{-1}(\mathbf{b})(x+x_i, y + y_i)] = (x + x_i)\psi[\psi^{-1}(\mathbf{b})(x + x_i, y + y_i)] = (x + x_i)H^{(i)}(\mathbf{b})$, and $\gamma_1 = \gamma_x = x_i$. We define $M^{(i)} = \langle\{x^{j_1}\mathbf{e}_{j_2+1} | j_1 + j_2 = m_i\}\rangle$. Let $I_r^{(i)}$ denote the sequence of ideals defined in the preamble to Algorithm 5.2.1. Then $M_r^{(i)} = \psi(I_r^{(i)} \bigcap B)$ defines a corresponding descending sequence of $\frac{m_i(m_i+1)}{2}$ A–modules between A^L and $M^{(i)}$ and the terms $\psi(\varphi_r^{(i)})$ satisfy $M_r^{(i)} = \langle\psi(\varphi_r^{(i)}), M_{r+1}^{(i)}\rangle$, and $xM_r^{(i)} \subseteq M_{r+1}^{(i)}$ (giving condition (3.2) with $\beta_1 = \beta_x = 0$). The homomorphism $\theta_r^{(i)}$ returns the coefficient of $\psi(\varphi_r^{(i)})$. (Alternatively, theorem 3.3.7 can be applied directly using a sequence of terms from A^L and a weighted term order with weights $(0, 1, \ldots, L - 1)$). This results in a 1–variable algorithm with a Gröbner basis of size L at each step, in which for each component position there is precisely one element with leading term in that position. As noted previously, in this situation the function *ord* takes a particularly simple form. The positions of the leading terms of the basis elements are unchanged by the incremental step, and thus re-ordering entails moving at most one element at each iteration.

Algorithm 5.2.3

Input:

points $\{(x_i, y_i) : 1 \leq i \leq n\}$, with multiplicities m_i

Output:

Gröbner basis $\mathcal{W}$ of the solution module

$\psi(Q)$ as the first element of $\mathcal{W}$

Main Routine:

$\mathcal{W} :=$ ord(standard basis vectors of A^L) (†)

For i from 1 to n

For j_1 from 0 to $m_i - 1$

For j_2 from 0 to j_1

For j from 1 to L (⋆)

α_j:=coefficient of $x^{j_1-j_2}\mathbf{e}_{j_2+1}$ in $\psi[\psi^{-1}(\mathcal{W}[j])(x + x_i, y + y_i)]$

$\mathcal{W}'$=incremental-step$(\mathcal{W}, [x], [\alpha_j], [0], [x_i])$ (‡)

$\mathcal{W}$:=ord$(\mathcal{W}')$

We illustrate this algorithm by reworking Example 5.2.2.

EXAMPLE 5.2.4 We shall take $L = 3$. The initial basis is $\{(1, 0, 0), (0, 1, 0), (0, 0, 1)\}$. Again, the notation $[\,\cdot\,]_X$ will indicate the coefficient of the monomial X in the relevant module element. The iterations are as follows:

$\underline{(i, j_1, j_2) = (1, 0, 0)}$

$[\varphi(\varphi^{-1}(1, 0, 0)(x + 1, y)]_{1\mathbf{e}_1} = 1,$

$[\varphi(\varphi^{-1}(0, 1, 0)(x + 1, y)]_{1\mathbf{e}_1} = 0,$

$[\varphi(\varphi^{-1}(0, 0, 1)(x + 1, y)]_{1\mathbf{e}_1} = 0,$

$\rightarrow \{(x - 1, 0, 0), (0, 1, 0), (0, 0, 1)\}.$

Note that the leading terms have $(2, 3)$-degrees 2,3,6, respectively.

$\underline{(i, j_1, j_2) = (2, 0, 0)}$

$[\varphi(\varphi^{-1}(x - 1, 0, 0)(x, y + 1)]_{1\mathbf{e}_1} = -1,$

$[\varphi(\varphi^{-1}(0,1,0)(x,y+1)]_{1\mathbf{e}_1} = 1,$

$[\varphi(\varphi^{-1}(0,0,1)(x,y+1)]_{1\mathbf{e}_1} = 1,$

$\rightarrow \{(x-1,1,0),(x(x-1),0,0),(x-1,0,1)\}.$

$\underline{(i,j_1,j_2) = (2,1,0)}$

$[\varphi(\varphi^{-1}(x-1,1,0)(x,y+1)]_{x\mathbf{e}_1} = 1,$

$[\varphi(\varphi^{-1}(x(x-1),0,0)(x,y+1)]_{x\mathbf{e}_1} = -1,$

$[\varphi(\varphi^{-1}(x-1,0,1)(x,y+1)]_{x\mathbf{e}_1} = 1,$

$\rightarrow \{(x^2-1,1,0),(x(x-1),x,0),(0,-1,1)\}.$

$\underline{(i,j_1,j_2) = (2,1,1)}$

$[\varphi(\varphi^{-1}(x^2-1,1,0)(x,y+1)]_{1\mathbf{e}_2} = 1,$

$[\varphi(\varphi^{-1}(x(x-1),x,0)(x,y+1)]_{1\mathbf{e}_2} = 0,$

$[\varphi(\varphi^{-1}(0,-1,1)(x,y+1)]_{1\mathbf{e}_2} = 1,$

$\rightarrow \{(x(x-1),x,0),(x(x^2-1),x,0),(-x^2+1,-2,1)\}.$

Note that the final basis corresponds to $\{x(y+x-1), x(x^2+y-1), y^2-x^2-2y+1\}$. The first two elements are the same as those in the basis derived in 5.2.2, while the third is obtained from that basis as $(y-1)(y+x-1)-x(y+x-1)$.

$\Diamond$

We end with a variation of Algorithm 5.2.3 related to [45, Algorithm 15]. For any $\rho \in \mathbb{N}$, the set $\mathcal{W}_0 = \{x^{j_1}\mathbf{e}_{j_2} | 0 \leq j_1 \leq \rho - 1, 1 \leq j_2 \leq L\}$ is a Gröbner basis of A^L with $|\mathcal{W}_0| = \rho L$ elements. With this as the initial basis, we obtain an algorithm with ρL elements in the basis at each iteration.

Algorithm 5.2.5

Algorithm 5.2.3 with the following changes

$\mathcal{W} := \text{ord}(\mathcal{W}_0) = \text{ord}(\{x^{j_1}\mathbf{e}_{j_2} | 0 \leq j_1 \leq \rho - 1, 1 \leq j_2 \leq L\})$ (†)

for j from 1 to ρL ($\star$)

$\mathcal{W}'$=ρ-incremental-step($\mathcal{W}, [x], [\alpha_j], [0], [x_i], \rho$) (‡)

Proc ρ-incremental-step()

: modified incremental step - instruction ($**$)

If $\alpha_j = 0$ for all j then

$\mathcal{W}' = \mathcal{W}$

otherwise

$j^* :=$ least j for which $\alpha_j \neq 0$

$\mathcal{W}_1 := \{\mathcal{W}_j : j < j^*\}$

$\mathcal{W}_2 := \{(x - (\beta + \gamma))^\rho \mathcal{W}[j^*]\}$ ($**$)

$\mathcal{W}_3 := \{\mathcal{W}[j] - (\alpha_j/\alpha_{j^*})\mathcal{W}[j^*] : j > j^*\}$

$\mathcal{W}_1 \bigcup \mathcal{W}_2 \bigcup \mathcal{W}_3$

End

PROOF. We prove by induction that each iteration of the inner loop (the incremental step) produces a Gröbner basis $\{\mathcal{W}[1], \ldots, \mathcal{W}[\rho L]\}$ for next solution submodule in the sequence and that the sets $P_j = \{x^{m\rho}\text{lt}(\mathcal{W}[j]) | m \in \mathbb{N}_0\}, j = 1, \ldots, \rho L$ partition the leading terms of that submodule. The assertion is obviously true for the initial basis $\mathcal{W}_0$. Let us now fix i, r and drop the superscripts (i). As in Theorem 3.1.2, let $S \subseteq A^L$ be a submodule such that $H(\mathbf{b}) \equiv 0 \bmod M_r$ for all $\mathbf{b} \in S$, and $S' = \{\mathbf{b} \in A^L | H(\mathbf{b}) \equiv 0 \bmod M_{r+1}\}$.

We assume that $\mathcal{W}$ is a Gröbner basis for S and that the sets

$$P_j = \{x^{m\rho}\text{lt}(\mathcal{W}[j]) | m \in \mathbb{N}_0\}, j = 1, \ldots, \rho L$$

partition the leading terms of S. Since $(x - x_i)\mathcal{W}[j^*] \in S'$ it follows that

$(x-x_i)^\rho \mathcal{W}[j^*]$ is in S'. Also, $\mathrm{lt}((x-x_i)^\rho \mathcal{W}[j^*]) = x^\rho \mathrm{lt}(\mathcal{W}[j^*])$. Following the proof of Theorem 3.1.2, we may assume $S' \subseteq S$, and observe that $\mathrm{lt}(\mathcal{W}[j^*])$ is the only leading term that changes at this iteration. We may therefore restrict our attention to those $\mathbf{f} \in S'$ such that $\mathrm{lt}(\mathcal{W}[j^*])$ divides $\mathrm{lt}(\mathbf{f})$.

By definition, $\mathrm{lt}(\mathbf{f})$ is in one and only one subset P_j of the partition. If $j \neq j^*$ then $\mathrm{lt}(\mathbf{f})$ is divisible by the leading term of a basis element other than $\mathcal{W}[j^*]$. We can therefore assume $\mathrm{lt}(\mathbf{f}) \in P_{j^*}$ and thus $\mathrm{lt}(\mathbf{f}) = x^{m_f \rho} \mathrm{lt}(\mathcal{W}[j^*])$ for some $m_f \in \mathbb{N}_0$. If $\mathrm{lt}(\mathbf{f}) \neq \mathrm{lt}(\mathcal{W}[j^*])$ then $\mathrm{lt}(\mathbf{f})$ is a multiple of $x^\rho \mathrm{lt}(\mathcal{W}[j^*])$, which is what we want. Otherwise, $\mathrm{lt}(\mathbf{f}) = \mathrm{lt}(\mathcal{W}[j^*])$. Let $\mathbf{f}^* = \mathbf{f} - (\mathrm{lc}(\mathbf{f})/\mathrm{lc}(\mathcal{W}[j^*]))\mathcal{W}[j^*]$ and note that $\mathbf{f}^* \in S$. Now, $\mathrm{lt}(\mathbf{f}^*) < \mathrm{lt}(\mathcal{W}[j^*])$ and

$$\begin{aligned} \theta_\ell(H(\mathbf{f}^*)) &= \theta_\ell(H(\mathbf{f})) - (\mathrm{lc}(\mathbf{f})/\mathrm{lc}(\mathcal{W}[j^*]))\theta_\ell(H(\mathcal{W}[j^*])) \\ &= -(\mathrm{lc}(\mathbf{f})/\mathrm{lc}(\mathcal{W}[j^*]))\theta_\ell(H(\mathcal{W}[j^*])) \\ &\neq 0 \end{aligned}$$

so $\mathbf{f}^*$ is not an element of S'. However, the standard representation of $\mathbf{f}^*$ contains only basis elements whose leading terms are less than $\mathrm{lt}(\mathcal{W}[j^*])$ and which are therefore elements of S'. This is a contradiction, and we conclude that the new basis is a Gröbner basis of S'.

Next, let $P'_j = P_j$ for $j \neq j^*$ and $P'_{j^*} = \{x^{m\rho}(x^\rho \mathrm{lt}(\mathcal{W}[j^*]))|m \in \mathbb{N}_0\}$. Since $P'_{j^*} \subset P_{j^*}$, the sets $P'_j, j = 1, \ldots, \rho L$ are disjoint. Thus the new basis has no duplicate leading terms and the function *ord* does not remove any basis elements. Suppose that $\mathbf{f} \in S'$ and $\mathrm{lt}(\mathbf{f}) \notin \bigcup_{j \neq j^*} P'_i (= \bigcup_{j \neq j^*} P_j)$. From the previous paragraph, $\mathrm{lt}(\mathbf{f}) = x^{m_f \rho} \mathrm{lt}(\mathcal{W}[j^*])$ for some $m_f \in \mathbb{N}_0, m_f \neq 0$. It now follows that $\bigcup_{j=1}^{\rho L} P'_j$ partitions the set of leading terms of S'. $\square$

EXAMPLE 5.2.6 Once again, we reformulate Example 5.2.2. It is straightforward to verify that the final basis is a Gröbner basis of the same submodule as that obtained in 5.2.2 and 5.2.4. Here $\rho = 3$ and the sequence of bases is as follows:

Initially $\{(1,0,0),(x,0,0),(0,1,0),(x^2,0,0),(0,x,0),(0,0,1),$
$(0,x^2,0),(0,0,x),(0,0,x^2)\}$

$(i,j_1,j_2) = (1,0,0)$

$\{(x-1,0,0),(0,1,0),(x^2-1,0,0),(0,x,0),((x-1)^3,0,0),$
$(0,0,1),(0,x^2,0),(0,0,x),(0,0,x^2)\}$

$(i,j_1,j_2) = (2,0,0)$

$\{(x-1,1,0),(x(x-1),0,0),(0,x,0),((x^3-3x^2+2x,0,0),$
$(x-1,0,1),(0,x^2,0),(0,0,x),(x^3(x-1),0,0),(0,0,x^2)\}$

$(i,j_1,j_2) = (2,1,0)$

$\{(x^2-1,1,0),(-(x-1),x-1,0),(x^3-3x^2-2,-2,0),(0,-1,1),(0,x^2,0),$
$(x^3(x-1),0,0),(-(x-1),1,,x),(x^3(x-1),x^3,0),(0,0,x^2)\}$

$(i,j_1,j_2) = (2,1,1)$

$\{(x(x-1),x,0),(x^2(x-1),0,0),(-x^2+1,-2,1),(0,x^2,0),(x^3(x-1),0,0),$
$(x^2-x,0,x),(x^3(x-1),x^3,0),(x^3(x^2-1),x^3,0),(0,0,x^2)\}$ $\Diamond$

5.2.1 Hard decision list decoding

Low rate codes

Sudan's algorithm [60] applies provided the rate of the code $\frac{K}{N} < \frac{1}{3}$. The interpolation step has complexity $O(n^3)$ when Gaussian elimination is used. Roth and Ruckenstein [57, 58] improved the efficiency of Sudan's original algorithm by deriving and solving an *extended key equation*. The main effect is to accelerate the original interpolation step. Suppose that the list of code-

words sought is to contain at most ℓ elements, then Sudan's algorithm can correct up to $\tau = \tau(\ell)$ errors. The received word is used to generate a set of polynomials $S^{(i)}$ and a solution $(\Lambda^{(1)}, \ldots, \Lambda^{(s)}, \Omega)$ is sought to the congruence

$$\sum_{i=1}^{\ell} \Lambda^{(i)} x^{(\ell-i)(N-K)} S^{(i)} \equiv \Omega(x) \bmod x^{\ell(N-K)} ([57]) \tag{5.2}$$

subject to the constraints $\partial\Lambda^{(i)} < m + 1 + (\ell - i)(K - 1), 1 \leq i \leq \ell$ and $\partial\Omega(x) < \ell(N - K) - \tau$. This was solved in [57] using a modified form of the "Fundamental Iterative Algorithm" of Feng and Tseng [20].

In [58], the key equation takes on the more simplified form

$$\sum_{i=1}^{\ell} \Lambda^{(i)} x^{(i-1)(K-1)} S^{(i)} \equiv \Omega(x) \bmod x^{N-K} \tag{5.3}$$

subject to the constraints $\partial\Lambda^{(i)} < N_i = N - \tau - i(K - 1), 1 \leq i \leq \ell$ and $\partial\Omega(x) < N - K - \tau$. Both (5.2) and (5.3) are of a form that can solved by algorithm 4.1.2.

By setting $h_{\ell+1} = -1$ and $h_i = x^{(i-1)(K-1)} S^{(i)}, 1 \leq i \leq \ell$, congruence (5.3) can be recast as

$$\sum_{i=1}^{\ell+1} b_i h_i \equiv 0 \bmod x^{N-K} \tag{5.4}$$

subject to constraints of the form $\partial b_i < N_i, 1 \leq i \leq \ell$ and $\partial b_{\ell+1} < N - K - \tau$. We can apply 4.1.2 using the sequence of ideals $I_r = \langle x^r \rangle, 0 \leq r \leq N - K$ and the function θ_r which returns the coefficient of x^r in the expansion of the left hand side of (5.4). The term order arising is another instance of that described in Section 2.2.

EXAMPLE 5.2.7 [58, Example 7.1]. Consider a Reed-Solomon code of length 18 over F_{19} with dimension $K = 2$. For $\ell = 4, m = 1$, we have $\tau = 12$. The degree constraints are then $4, 3, 2, 1, 3$ and the weights for the term order are $0, 1, 2, 3, 1$.

For the received word

$$(5, 5, 1, 10, 10, 7, 2, 18, 6, 6, 1, 15, 13, 5, 14, 3, 1, 0), \tag{5.5}$$

the four syndromes lead to solutions of (5.3) which take the form

$$\alpha(-8x^4+2x^3+6x^2+6x-3, 4x^3-8x^2-3x-4, -4x^2+8x-2, x-1, 2x^2+x-1)$$

$$+\beta(-x^4+4x^3+9x^2+x+7, 5x^3-8x^2-4x-2, -7x^2+5x+6, -4x+9, x^3+8x-4),$$

where $\alpha, \beta \in F_{19}$. The solutions produced by [58] correspond to $\alpha = 4, \beta = 11$ and $\alpha = 9, \beta = -1$. $\Diamond$

On the other hand, we can find Q directly using the algorithms from the previous section. In this case, $L = \ell + 1$ and $m_i = 1, 1 \leq i \leq n$ (or $\rho = 1$). Since a suitable polynomial with $(1, K-1)$–degree is known to exist in this module, the minimal element of the Gröbner basis with respect to $(1, K-1)$ order is a fortiori a solution. The minimal element produced by Algorithm 5.2.3 when applied to Example 5.2.7 is

$$(-6x^5+8x^4-8x^3+5x^2-7x+4, 8x^4-3x^3+9x^2-5x-5, x^2-6x-5, x^2-8x+2, -2).$$

This gives

$$Q = (-6x^5 + 8x^4 - 8x^3 + 5x^2 - 7x + 4) + (8x^4 - 3x^3 + 9x^2 - 5x - 5)y +$$

$$(x^2 - 6x - 5)y^2 + (x^2 - 8x + 2)y^3 + (-2)y^4.$$

Extension to codes of all rates

The conditions on Sudan's algorithm confine its applicability to low rate codes. Guruswami and Sudan [30] extended the algorithm to codes of all rates by requiring, in addition, that the polynomial $Q(x, y)$ have certain

derivatives equal to zero, equivalently, have certain multiple zeros, at the interpolating points.

We require a polynomial $Q(x, y)$, with a constrained $(1, K-1)$–degree, such that

$$q_{j_1 j_2 i} = 0 \text{ for } 1 \leq i \leq N \text{ and } j_1 + j_2 < m.$$

Again, the minimal element of the Gröbner basis with respect to $(1, K-1)$ order is a solution. We can apply Algorithm 5.2.1 directly or use Algorithm 5.2.3 on the constrained problem. We will illustrate the latter with an example from [45].

EXAMPLE 5.2.8 [45, Section 8]. Consider a Reed-Solomon code of length $N = 15$ and dimension $K = 7$ over F_{16} (with primitive element α satisfying $\alpha^4 + \alpha + 1 = 0$). Suppose the received word is

$$(1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0).$$

We can apply Algorithm 5.2.3 to find the interpolation polynomial. We have $m_i = 4, 1 \leq i \leq 15$. There exists a suitable polynomial f with $\Delta_{(1,6)}(f) \leq 40$. Since this forces $\Delta_{(0,1)}(f) < 7$, we can choose $L = 7$. The final basis has minimal element

$$(0, 0, x^{20} + x^{10} + 1, 0, x^{10}, 0, 1)$$

and the corresponding polynomial is

$$Q = (x^{20} + x^{10} + 1)y^2 + (x^{10})y^4 + (1)y^6.$$

$\Diamond$

For the Guruswami and Sudan formulation, Algorithm 5.2.5 coincides with the technique given by Nielsen and Høholdt [45, Algorithm 15]. For a multiplicity ρ the initial basis is $\{x^{j_1} y^{j_2} | 0 \leq j_1 \leq \rho - 1, 0 \leq j_2 \leq L\}$, where L

is the maximum $(0,1)$–degree required, which corresponds to the basis used above.

5.2.2 List decoding with soft information

Kötter and Vardy [35] employ a Sudan-like technique in the context of algebraic soft decision decoding of Reed-Solomon codes. Suboptimal soft decision list decoding is presented in the context of concatenated codes, with the outer code being a Reed-Solomon code. After a word is received, soft information is presented to the outer decoder in the form of posterior probabilities $\{\Pi_{ij}\}$, where Π_{ij} is the probability that field element γ_i was sent given the y_j was at location j in the received word. From this $q \times N$ reliability matrix, they derive a $q \times N$ multiplicity matrix $M = \{m_{ij}\}$ using a greedy algorithm. For multiplicity m_{ij} the corresponding interpolation point is (γ_i, y_j). The multiplicity is not, in general, the same at each of the qN points.

The hard decision situation corresponds to the case where the reliability matrix has a single 1 in each column and is 0 elsewhere. Only these n entries lead to non-zero multiplicities. Thus, we can see that both Sudan's original algorithm and the extension by Guruswami and Sudan are special cases of the method of Kötter and Vardy. In addition, Kötter and Vardy impose a condition of *minimality* of the required solution $Q(x,y)$ with respect to an (a,b)–degree for $a = 1, b = K-1$. We see that our techniques solve the Kötter-Vardy problem in a very natural way.

5.3 Algebraic Geometry codes

The construction of Algebraic Geometry (or Geometric Goppa) codes is analogous to that of Reed-Solomon codes. It is based on evaluating rational

functions at points on algebraic curves. A comprehensive description of AG codes can be found in [10, 51, 64]. Practical decoding of AG codes was introduced in Justesen *et al.* [32]. Based on an algorithm from Sakata [61], faster decoding was presented in Justesen *et al.* [33].

We will only consider the special case of 1-point (AG) codes. Let F_q be the finite field with q elements and $F_q[x]$ the ring of polynomials in one indeterminate over F_q. A Reed-Solomon code of dimension K and length $N = q - 1$ can viewed as the evaluation of polynomials in $F_q[x]$ with degree less than K at the N non-zero elements of F_q. The size of the base field is a limit on the length of such a code. Drawing on Algebraic Geometry, longer codes can be created by analogy with the description of Reed-Solomon codes above. Rational functions on a curve are evaluated at F_q-rational points of that curve, where the pole order of these functions at a single point takes the part which is played by the polynomial degree in the case of Reed-Solomon codes.

Let χ be an absolutely irreducible curve of genus g over F_q. Denote $n+1$ F_q-rational points on χ by $P_1, \ldots, P_n, P_\infty$. Define $\mathcal{L}(\ell P_\infty))$ to be the set of rational functions in χ at P_∞ such that the pole order of these functions at P_∞ is at most ℓ. For $2g-1 \leq k < n$, a 1-point code $\mathcal{C}_\chi(k, P_\infty)$ can be defined as the vector space (over F_q)

$$\{(f(P_1), \ldots, f(P_n)) | f \in \mathcal{L}(kP_\infty)\}. \tag{5.6}$$

For any $\ell \geq 2g - 1$ there are functions $\phi_{1,\infty}, \ldots, \phi_{\ell-g+1,\infty}$, with increasing pole orders, that form a (vector space) basis of $\mathcal{L}(\ell P_\infty)$. Thus a code defined by (5.6) has length $N = n$ and dimension $K = k - g + 1$. (For $\ell > 0$, the basis of $\mathcal{L}(\ell P_\infty)$ can be extended to one with $\ell + 1$ elements by assigning the zero function to elements corresponding to the g "gaps" in the pole orders.

It is this form that is used in [35].)

Both Shokrollahi and Wasserman [65] and Guruswami and Sudan [30] applied Sudan's technique to 1–point codes. An improved interpolation step was presented by Høholdt and Nielsen [31]. Sakata [62] also provided an interpolation improvement by using a Gröbner basis approach and the Berlekamp-Massey-Sakata algorithm. Kötter and Vardy [35] used an approach similar to [30] for soft decision decoding of Reed-Solomon and 1–point codes.

We address the interpolation problems for 1–point codes arising in [30] and [35] by applying our general algorithm. We present a common algorithm for both the hard and soft decision cases.

Let $R = \bigcup_{\ell=0}^{\infty} \mathcal{L}(\ell P_\infty)$ and let z be transcendental over $F_q(\chi)$. Consider the polynomial $Q \in R[z]$ where

$$Q(z) = \sum_{j_1=0}^{b} \sum_{j_2=1}^{a} q_{j_1,j_2} z^{j_1} \phi_{j_2,\infty}.$$

It is well known ([30],[31],[35]) that, for any of the points $P_i \neq P_\infty$, there is a basis for $\mathcal{L}(\ell P_\infty)$ of functions $\phi_{1,i}, \ldots, \phi_{\ell-g+1,i}$ with increasing zero order at P_i. There is a set of basis conversion constants $\{\delta_{i,j_2,j_3} \in F_q | i \in [n], j_2, j_3 \in [\ell - g + 1]\}$ such that for any i, j_2

$$\phi_{j_2,\infty} = \sum_{j_3} \delta_{i,j_2,j_3} \phi_{j_3,i} \tag{5.7}$$

(Again, for $\ell > 0$, these bases can be extended to ones with $\ell + 1$ elements by assigning the zero function to elements corresponding to the "gaps" in the pole orders.) These observations lead to the extension of Sudan's algorithm to 1–point codes. In that case, the shifting of the first indeterminate by x_i is replaced by expanding with respect to the zero basis at P_i.

By associating $\phi_{j_2,\infty}$ with $\mathbf{e}_{j_2}$, we can view Q as an element $\mathbf{Q}_M$ of the free $F_q[z]$-module $M = F_q[z]^a$, where each component has degree $\leq b$.

Let $Q^{(i,\gamma)}(z) = Q(z+\gamma)$. We can expand $Q^{(i,\gamma)}(x)$ around the basis elements $\phi_{1,i}, \ldots, \phi_{\ell,i}$ at P_i. By associating $\phi_{j_2,i}$ with $\mathbf{e}_{j_2}$ in this expansion, $Q^{(i,\gamma)}(z)$ can be viewed as an element $\mathbf{Q}_M^{(i,\gamma)}$of M.

The function which maps Q to $Q^{(i,\gamma)}$ depends only on γ and $\{\delta_{i,j_2,j_3} \in F_q | i \in [n], j_2, j_3 \in [a]\}$. We define as its counterpart the function $H^{(i,\gamma)} : M \rightarrow M$ that maps $\mathbf{Q}_M$ to $\mathbf{Q}_M^{(i,\gamma)}$. We see that $H^{(i,\gamma)}$ is F-linear and $H^{(i,\gamma)}(z\mathbf{b}) = (z+\gamma)H^{(i,\gamma)}(\mathbf{b})$. Thus $H^{(i,\gamma)}$ satisfies (3.6).

Q is said to have a *zero of multiplicity at least* m at (P_i, γ) if the coefficients of the terms $\phi_{j_2,i} z^{j_1}$ of $Q^{(i,\gamma)}(z)$ are zero when $j_1 + (j_2 - 1) < m$. Equivalently, $\mathbf{Q}_M$ has a zero of multiplicity at least m at (P_i, γ) if the coefficients of the terms $z^{j_1}\mathbf{e}_{j_2}$ of $\mathbf{Q}_M^{(i,\gamma)}$ are zero when $j_1 + (j_2 - 1) < m$. Thus a module sequence satisfying (3.7) can be constructed as in Theorem 3.3.7.

The following problems have interpolations at their core. The parameters for these interpolations are chosen so as to guarantee the existence of interpolating polynomials whose factors provide the list of valid codewords. These parameters also curtail the search space for polynomials so that efficiencies can be introduced. In particular, our module description leads to practical algorithms.

5.3.1 Hard decision list decoding

The interpolation step in Guruswami and Sudan [30] has parameters ℓ and m. These can be calculated in advance from the code parameters n,k and the number of errors we wish to correct.. Let $(z_1, \ldots, z_n)$ be the received word. Define $s = \lfloor \frac{\ell - g}{k+g-1} \rfloor$. A polynomial of the form

$$Q(z) = \sum_{j_1=0}^{s} \sum_{j_2=1}^{\ell-g+1-(k+g-1)j_1} q_{j_1 j_2} \phi_{j_2,\infty} z^{j_1}$$

is sought which has a zero of multiplicity at least m at each point (P_i, z_i), $1 \leq i \leq n$.

Using the module description, there exists a solution $Q_M \in F[z]^{l-g+1}$ whose terms satisfy $(k+g-1)j_1 + (j_2 - 1) < \ell - g + 1$. All solutions whose terms have this property are contained in $F[z]^{\ell-g+1}$. A fortiori, there is a minimal solution with respect to $<_{k+g-1,(0,1,2,\ldots,\ell-g)}$ in $F[z]^{\ell-g+1}$. This element will be the first of an ordered Gröbner basis, with respect to this order, of the solution module

$$\{\mathbf{b} \in F_q[z]^{\ell-g+1} | H^{(i,y_i)}(\mathbf{b}) \equiv 0 \bmod M_m, i = 1, \ldots, n\}$$

where $M_m = \{\mathbf{f} \in A^u |$ the coefficients of terms $\mathbf{t}$ of $\mathbf{f}$ are zero for all $\mathbf{t} = z^{j_1}\mathbf{e}_{j_2}$ with $j_1 + (j_2 - 1) < m\}$.

Thus, all of the requirements of our general algorithm are satisfied and the miminal element produced, by applying it as follows, is a solution to the interpolation problem. We order the terms of M_m using POT.

Algorithm 5.3.1

Input

Genus g of the curve.

Parameters n, k, ℓ, m.

Basis conversion constants

$\{\delta_{i,j_2,j_3} \in F_q | i \in [n], j_2, j_3 \in [\ell - g + 1]\}$.

The received word $(z_1, \ldots, z_n)$.

Output

The first element of $\mathcal{W}$ a minimal Gröbner basis with respect to $<_{k+g-1,(0,1,\ldots,l-g)}$.

Main Routine

$\mathcal{W}$:=the ordered standard basis vectors of $F[x]^{\ell-g+1}$.

Choose term order $<_{k+g-1,(0,1,\ldots,\ell-g)}$

For i from 1 to n

For j_2 from 1 to $\min\{m, \ell - g + 1\}$

For j_1 from 0 to $m - j_2$

$\alpha_j := \text{coeff}(x^{j_1}\mathbf{e}_{j_2}, H^{(i,y_i)}(\mathcal{W}[j]))$

for $j \in [\ell - g + 1]$

$\mathcal{W}'$=incremental-step$(\mathcal{W}, [z], [\alpha_j], [0], [z_i])$

$\mathcal{W} := \text{ord}(\mathcal{W}')$

This version of the algorithm is similar in form to Høholdt and Nielsen [31] Algorithm 3. By contrast, they took the perspective of a vector space over the function field.

5.3.2 List decoding with soft information

For the purposes of this section we use the extended version of the bases of $\mathcal{L}(aP_\infty)$. Such bases $\phi_{0,i}, \ldots, \phi_{a,i}, i \in [n] \bigcup \{\infty\}$ have $a+1$ elements. Consider

the polynomial

$$Q(z) = \sum_{j_2=0}^{a} \sum_{j_1=0}^{b} q_{j_1,j_2} \phi_{j_2,\infty} z^{j_1}.$$

Reflecting the changes in the bases, we will say Q has a *zero of multiplicity at least* m at (P_i, γ) if the coefficients of the terms $\phi_{j_2,i} z^{j_1}$ of $Q^{(i,\gamma)}(z)$ are zero when $j_1 + j_2 < m$.

Kötter and Vardy [35] investigated soft decision list decoding of concatenated codes with the outer code being a 1–point code.

Let $\gamma_1, \ldots, \gamma_q$ be the elements of F_q. The task is to find a polynomial $Q(z)$, with minimal leading term with respect to $kj_1 + j_2$, which has a multiplicity at least m_{ij} for each point (P_j, γ_i) with $j \in [n]$ and $\gamma_i \in F_q$, when $m_{ij} \neq 0$.

Again, the existence of a polynomial is guaranteed by the parameters of the problem. The cost $C(M)$ of the multiplicity matrix M is defined as $\frac{1}{2} \sum_{i=1}^{q} \sum_{j=1}^{n} m_{ij}(m_{ij}+1)$. Choose ν to be minimum value such the dimension of the vector space of terms where $kj_1 + j_2 \leq \nu$ is greater than $C(M)$. This guarantees the existence of a solution with $kj_1 + j_2 \leq \nu$ and we can confine our search to $F_q[z]^L$ where $L = \nu + 1$.

We associate $\phi_{j,i}$ with $\mathbf{e}_{j+1}$. Similarly, $\mathbf{Q}_M$ has a zero of multiplicity at least m at (P_i, γ) if the coefficients of the terms $z^{j_1}\mathbf{e}_{j_2}$ of $\mathbf{Q}_M^{(i,\gamma)}$ are zero when $j_1 + (j_2 - 1) < m$. Again, Theorem 3.3.7 can be used to generate a descending module sequence.

A minimal element of the solution submodule with respect to $<_{k,w}$ where $w = (0, 1, 2, \ldots, L-1)$ corresponds to the required solution and thus, again, we can apply our algorithm.

5.3.3 The common list decoding algorithm

While an algorithm for the hard decision interpolation problem is an immediate consequence of our general algorithm, this problem can also be considered as a special case of the soft decision interpolation and a common algorithm [49] can be used to solve them both.

We can create a "multiplicity" matrix from the received word $(z_1, \ldots, z_n)$. Let $m_{ij} = m$ when $z_i = \gamma_j$ and $m_{ij} = 0$ otherwise. Set $L = \ell - g + 1$ and $K = k + g - 1$.

Both problems can now be solved using the following algorithm.

Algorithm 5.3.2

Input

M the $q \times n$ multiplicity matrix

L the module dimension

Function field basis functions $\phi_{j,i}$,$j \in [L], i \in [n] \bigcup \{\infty\}$.

Basis conversion constants $\{\delta_{i,j_2,j_3} \in F_q | i \in [n], j_2, j_3 \in [L]\}$.

K a weight for $<_{K,w}$ a term order with $w = (0, 1, \ldots, L-1)$.

Output

The first element of $\mathcal{W}$ a minimal Gröbner basis with respect to $<_{K,w}$

Main Routine

$\mathcal{W}$:=the ordered standard basis vectors of $F[z]^L$.

Choose term order $<_{K,w}$,$w = (0, 1, \ldots, L-1)$

For j from 1 to n

For i from 1 to q

If $m_{ij} \neq 0$

For j_2 from 1 to $\min\{m_{ij}, L\}$

For j_1 from 0 to $m_{ij} - j_2$

$\alpha_j := \text{coeff}(z^{j_1}\mathbf{e}_{j_2}, H^{(i,y_i)}(\mathcal{W}[j]))$

for $j \in [L]$

$\mathcal{W}'$=incremental-step($\mathcal{W}, [z], [\alpha_j], [0], [\gamma_i]$)

$\mathcal{W} := \text{ord}(\mathcal{W}')$

We illustrate Algorithm 5.3.2 with an example given by Høholdt and Nielsen.

EXAMPLE 5.3.3 [31, Section 8]. Consider the Hermitian curve χ defined by $Y^2 + Y - X^3 = 0$ over F_4. Let α be a primitive element for F_4 satisfying $\alpha^2 + \alpha + 1 = 0$. This curve has genus 1. With $P_\infty = (0 : 1 : 0)$ on the

corresponding projective curve and
$\{P_i\}_{i=1}^{8} = \{(0,0),(0,1),(1,\alpha),(1,\alpha^2),(\alpha,\alpha),(\alpha,\alpha^2),(\alpha^2,\alpha),(\alpha^2,\alpha^2)\}$,
a 1-point code $\mathcal{C}_\chi(4, P_\infty)$ can be defined as the vector space

$$\{(f(P_1),\ldots,f(P_8))|f \in \mathcal{L}(4P_\infty)\}$$

over F_4. For $m > 0$, $\{x^i y^j | 2i + 3j \leq m, i \geq 0, 0 \leq j \leq 1\}$ is a basis for $\mathcal{L}(mP_\infty)$. This code has length 8, dimension 4 and designed distance 4. Now $f = \alpha^2 + \alpha^2 x + \alpha^2 x^2 \in \mathcal{L}(4P_\infty)$ and the corresponding codeword is $(\alpha^2, \alpha^2, \alpha^2, \alpha^2, 0, 0, 0, 0)$. This codeword has weight 4 and, so, this code has a minimum distance of 4.

Suppose that code word $(0, 0, 0, 0, 0, 0, 0, 0)$ was sent and the received word was $(\alpha^2, 0, 0, \alpha^2, 0, 0, 0, 0)$. This means that two errors have occurred and the usual error correcting capability of the code is exceeded.

Using list decoding, [31] shows that there are two codewords within a Hamming distance 2 of that received word. These codewords, $(0, 0, 0, 0, 0, 0, 0, 0)$ and $(\alpha^2, \alpha^2, \alpha^2, \alpha^2, 0, 0, 0, 0)$, correspond the zero function and $f_1 = \alpha^2 + \alpha^2 x + \alpha^2 x^2$, respectively. Based on an approach by Feng and Blahut [19], a modified interpolation step is presented in [31]. For the case of Hermitian codes, the calculation of zero bases can be simplified and alternative methods can be found in [31] and [46, Section 5.5].

By setting $n = 8, L = 71, K = 4$ and $m_{ij} = 12$, this example can be solved using Algorithm 5.3.2. Expressing the minimal element of the resulting Gröbner basis as a polynomial in $R[z]$ and grouping in powers of z, a solution to the Guruswami and Sudan interpolation problem is
$Q(z) = (\alpha^2 x^{27} + \alpha^2 x^{25} y + \alpha^2 x^{25} + \alpha x^{24} y + \alpha^2 x^{24} + \alpha^2 x^{23} y + \alpha^2 x^{23} + x^{22} y +$
$x^{21} y + x^{21} + \alpha x^{20} y + \alpha x^{19} y + \alpha x^{19} + x^{17} y + x^{17} + x^{16} y + \alpha^2 x^{15} y + \alpha^2 x^{15} +$
$x^{14} y + x^{13} y + x^{13} + \alpha x^{12} y + \alpha x^{11} y + \alpha x^{11} + x^9 y + x^9 + x^8 y + \alpha^2 x^7 y + \alpha^2 x^7 +$

$$x^6y + x^5y + x^5 + \alpha x^4y + \alpha x^3y + x^3 + \alpha xy + \alpha x + \alpha^2y + \alpha^2)z^3 +$$
$$(x^{25} + x^{24} + x^{23}y + x^{23} + \alpha x^{22}y + x^{22} + \alpha x^{21}y + x^{21} + \alpha x^{20}y + \alpha x^{19}y + \alpha^2x^{19} +$$
$$\alpha^2x^{18}y + \alpha^2x^{18} + \alpha x^{17}y + \alpha^2x^{17} + x^{16}y + x^{15}y + x^{15} + \alpha x^{14}y + x^{14} + \alpha x^{13}y + x^{13} +$$
$$\alpha x^{12}y + \alpha x^{11}y + \alpha^2x^{11} + \alpha^2x^{10}y + \alpha^2x^{10} + \alpha x^9y + \alpha^2x^9 + x^8y + x^7y + x^7 + \alpha x^6y +$$
$$x^6 + \alpha x^5y + x^5 + \alpha x^4y + \alpha x^3y + \alpha^2x^3 + \alpha^2x^2y + \alpha^2x^2 + \alpha xy + \alpha x + y + 1)z^5 +$$
$$(x^{23} + \alpha x^{22}y + \alpha^2x^{22} + \alpha x^{21}y + x^{21} + \alpha x^{20}y + \alpha x^{19}y + \alpha x^{19} + x^{18} + \alpha x^{17}y +$$
$$\alpha^2x^{17} + \alpha^2x^{16}y + \alpha^2x^{16} + \alpha x^{15}y + \alpha x^{15} + \alpha^2x^{14}y + \alpha^2x^{14} + x^{13}y + \alpha x^{13} +$$
$$\alpha^2x^{12} + \alpha^2x^{11}y + \alpha x^{11} + \alpha^2x^{10}y + \alpha x^{10} + \alpha x^8y + \alpha^2x^8 + x^7y + x^7 + x^6y +$$
$$\alpha x^5 + \alpha^2x^4y + x^3y + x^3 + \alpha x^2y + \alpha x^2 + \alpha^2xy + \alpha^2x)z^6 +$$
$$(\alpha x^{21} + \alpha^2x^{20}y + \alpha^2x^{20} + \alpha^2x^{19} + \alpha^2x^{17}y + \alpha^2x^{16}y + \alpha x^{16} + \alpha^2x^{15}y + \alpha^2x^{15} + x^{14}y +$$
$$x^{13}y + x^{12}y + x^{12} + \alpha x^{11}y + \alpha x^{11} + \alpha^2x^{10}y + \alpha x^{10} + \alpha^2x^9 + \alpha x^8y + \alpha x^8 + \alpha x^7y +$$
$$x^7 + \alpha^2x^6y + \alpha x^6 + \alpha^2x^5y + x^5 + \alpha x^4y + \alpha^2x^4 + x^3y + x^3 + \alpha x^2y + \alpha x^2 + xy + x)z^7 +$$
$$(x^{20} + \alpha x^{19} + x^{18}y + x^{17}y + x^{17} + x^{16}y + \alpha^2x^{16} + x^{15} + \alpha x^{14} + \alpha x^{13} + \alpha^2x^{12} +$$
$$x^{10}y + \alpha^2x^{10} + x^9y + \alpha^2x^9 + x^8y + \alpha^2x^8 + \alpha x^7 + \alpha^2x^6 + \alpha x^5 + x^4 + x^3 + x^2y +$$
$$x^2 + xy + x + y + 1)z^8 +$$
$$(\alpha x^{18} + x^{17} + \alpha x^{16}y + \alpha^2x^{16} + \alpha x^{14} + x^{12} + \alpha x^{11} + \alpha x^{10} + \alpha x^8y + \alpha^2x^8 + \alpha x^7 +$$
$$x^5 + \alpha x^3 + \alpha y + \alpha)z^9 +$$
$$(\alpha^2x^{16} + \alpha x^{14} + \alpha^2x^{13}y + \alpha^2x^{13} + x^{12}y + \alpha x^{11} + \alpha^2x^{10}y + \alpha^2x^{10} + x^9y + \alpha x^8 +$$
$$\alpha^2x^7y + \alpha^2x^7 + x^6y + \alpha x^5 + \alpha^2x^4y + x^3y)z^{10} +$$
$$(x^{14} + x^{13} + \alpha^2x^{12} + x^{11}y + \alpha^2x^{11} + \alpha^2x^{10}y + \alpha x^9y + x^8 + x^7 + \alpha^2x^6 + x^5y +$$
$$\alpha^2x^5 + \alpha^2x^4y + \alpha x^3y)z^{11} +$$
$$(\alpha x^{12} + \alpha^2x^{11} + \alpha x^{10}y + \alpha x^{10} + \alpha x^9y + \alpha x^9 + \alpha x^8y + \alpha^2x^7 + \alpha x^6y + \alpha x^6 +$$
$$\alpha x^5y + \alpha x^4y + \alpha^2x^4 + \alpha^2x^3 + \alpha x^2y + \alpha xy + x + \alpha y + \alpha)z^{12} +$$
$$(\alpha^2x^{10} + \alpha x^9 + \alpha^2x^8y + \alpha x^8 + \alpha^2x^7 + x^6 + \alpha^2x^5 + \alpha^2x^4y + x^4 + \alpha x^3 + \alpha x^2 +$$
$$x + \alpha^2y + \alpha^2)z^{13} +$$
$$(x^8 + \alpha x^7 + x^6y + x^6 + x^5y + \alpha^2x^5 + \alpha^2x^4 + x^3y + \alpha x^3 + x^2 + xy + \alpha x + y)z^{14} +$$
$$(\alpha x^6 + x^5 + \alpha x^4y + x^4 + x^3 + \alpha x^2y + x^2 + \alpha^2x + \alpha y)z^{15} +$$

$(\alpha^2x^2 + \alpha^2x + \alpha^2)z^{16} +$

z^{17}.

The polynomials z and $z - f_1$ are factors of this polynomial.

The version of the interpolation problem in [31] can be solved by Algorithm 5.3.2 with the parameters $n = 8, L = 34, K = 4$ and $m_{ij} = 6$. The resulting polynomial

$Q(z) = (\alpha^2x^{12}+\alpha x^{11}y+\alpha^2x^{11}+x^{10}y+x^{10}+x^9y+x^9+x^8y+\alpha x^8+\alpha x^7y+\alpha^2x^7+$
$x^6y+x^6+x^5y+x^5+x^4y+\alpha x^4+\alpha x^3y+\alpha^2x^3+x^2y+x^2+xy+x+y+1)z^2+$
$(\alpha^2x^{11}+x^{10}+\alpha^2x^9+\alpha^2x^8y+x^8+\alpha x^7+\alpha^2x^6y+x^6+\alpha^2x^5y+\alpha x^5+\alpha^2x^4y+$
$\alpha^2x^3y+\alpha x^3+\alpha^2xy+\alpha^2x)z^3+$
$(x^9+x^7y+x^7+\alpha x^6y+\alpha^2x^6+\alpha^2x^5y+\alpha^2x^5+x^4y+\alpha^2x^3y+\alpha x^3+x^2y+$
$\alpha xy+\alpha x+\alpha^2y+\alpha^2)z^4+$
$(\alpha^2x^3+\alpha x^2+\alpha x+\alpha)z^6+$
$(\alpha x^3+x^2y+xy+x+y)z^7+$
$(\alpha^2x+\alpha y+1)z^8$
has both the z and $z - f_1$ among its factors.

$\Diamond$

Chapter 6

Further research

While the theory developed in this thesis applies to modules over polynomial rings in many indeterminates, the algorithms exhibited have, in the main, been for $A = F[x]$. Many of our applications are naturally of this form, but we also have been able to adapt problems in multiple indeterminates to avail of the regular form and the good complexity of the algorithms based on a single indeterminate. Nonetheless, it should be noted that the description of complexity in the most general case is very conservative. In particular, the construction of a minimal Gröbner basis at each iteration would significantly reduce the maximum size of the Gröbner basis .

Thus it may be fruitful to pursue a behavioural approach to nD systems from the perspective of our general algorithm. An allied avenue of research might be to examine the finite-precision effects of the algorithm. While the sensitivity of Gröbner basis calculations to inexact arithmetic is being studied in general, the robustness of our approach may be easier to establish or quantify because much of its underlying calculation is matrix multiplication.

With regard to coding theory, the list decoding of generalised Reed-Müller codes may be amenable to our methods. Recent work by Pellikaan and Wu

[50] addresses this using order domain, Reed-Solomon code and 1-point AG code approaches.

Bibliography

[1] W.W. Adams, P. Loustaunau, *An Introduction to Gröbner bases*, Springer-Verlag, New York, Berlin, 1994.

[2] A.C. Antoulas, On recursiveness and related topics in linear systems, *IEEE Trans. on Automatic Control*, Vol 31 (1986).

[3] A.C. Antoulas, Recursive modeling of discrete-time time series, in P. Van Doren and B. Wyman (eds.) *Linear Algebra for Control Theory, IMA*, Vol 62, 1776–1802, Springer, Berlin, 1994.

[4] A.C. Antoulas, B.D.Q. Anderson, On the scalar rational interpolation problem, *IMA Journal of Mathematical Control and Information* 3, (1986), 61-88.

[5] A.C. Antoulas, J.A. Ball, J. Kang, J.C. Willems, On the solution of the minimal rational interpolation problems, *Linear Algebra and its Applications* 137-138, (1990), 511-573.

[6] A.C. Antoulas, J.C. Willems, A behavioral approach to linear exact modeling, *IEEE Trans. on Automatic Control*, Vol 38(1993).

[7] T. Becker, V. Weispfenning, *Gröbner Bases: A Computational Approach to Commutative Algebra*, Springer Verlag, 1993.

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

[9] B. Beckermann, G. Labahn, Recursiveness in matrix rational interpolation problems, *J. of Computational and Applied Mathematics*, 77 (1997), 5–34.

[10] I. Blake, C. Heegard, T. Høholdt, V. Wei, Algebraic-Geometry Codes, *IEEE Trans. on Inform. Theory*, IT–44 (1998), 2596–2618.

[11] V.D. Blondel, D. Hinrichsen, J. Rosenthal and P. Van Dooren, Preface, *Fourth special issue on linear systems and control, V.D. Blondel, D. Hinrichsen, J. Rosenthal and P. Van Dooren(Eds.), Linear Algebra and its Applications* 351-352, (June 2002), 533-551.

[12] B. Buchberger, Ein algorithmisches Kriterium für Auffunden def Basislemente des Restklassenringes nach einem nulldimensionalen Polynomideal,*Ph.D thesis*, Univ. Innsbruck, Austria (1965).

[13] X. Chen, I.S. Reed, T. Helleseth, K. Truong, Use of Gröbner bases to decode binary cyclic codes up to the true minimum distance, *IEEE Trans. on Inform. Theory*, IT–40 (1994), 1654–1661.

[14] D. Cox, J. Little, D. O'Shea, *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*, Springer-Verlag, New York, Berlin, 1992.

[15] D. Cox, J. Little, D. O'Shea, *Using Algebraic Geometry*, Springer-Verlag, New York, Berlin, 1998.

[16] B.W. Dickinson, M. Morph, and T. Kailath, A minimal realization algorithm for matrix sequences, *IEEE Trans. on Automatic Control*, AC–19 (1974), 31–38.

[17] D. Eisenbud, *Commutative Algebra with a View towards Algebraic Geometry*, Graduate Texts in Mathematics, Springer-Verlag, 1995.

[18] P. Elias, Error-correcting codes for list decoding, *IEEE Trans. on Inform. Theory*, IT–37 (1991), 5–12.

[19] W. Feng, R.E. Blahut, Some results on the Sudan algorithm, In Proc. ISIT 1998 , Cambridge, MA, USA, August 1998.

[20] G.L. Feng, 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*, 37 (1991), 1274–1287.

[21] P. Fitzpatrick, J. Flynn, A Gröbner basis technique for Padé approximation, *J. Symb. Comp.*, 13 (1992), 133–138

[22] P. Fitzpatrick. On the key equation, *IEEE Trans. on Inform. Theory*, IT–41 (1995), 1290–1302.

[23] P. Fitzpatrick. On the scalar rational interpolation problem, *Mathematics of Control, Signals, and Systems*, 9 (1996), 352–369.

[24] P. Fitzpatrick. Errors and erasures decoding of BCH codes, *IEE Proc.-Commun.*, 146, No. 2 (1999), 79–81.

[25] P. Fitzpatrick. Rational interpolation using Gröbner bases: some numerical results, in *Mathematics in Signal Processing* IV, J.G. McWhirter, ed., Clarendon Press, Oxford, (1997), 35–45.

[26] P. Fitzpatrick. Solving a multivariable congruence by change of term order, *J. Symb. Comp.*, 11 (1997), 505–510.

[27] P. Fitzpatrick and S.M. Jennings, Comparison of two algorithms for decoding alternant codes, Appl. Alg. in Eng. Comm., and Comp., 9 (1998), 211-220

[28] R. Fröberg, *An Introduction to Gröbner bases*, John Wiley and Sons, 1997.

[29] G.-M. Greuel, G. Pfister, and H. Schönemann. SINGULAR 2.0. A Computer Algebra System for Polynomial Computations. *Centre for Computer Algebra, University of Kaiserslautern* (2001). `http://www.singular.uni-kl.de`.

[30] V. Guruswami, M. Sudan, Improved decoding of Reed-Solomon and algebraic-geometry codes, *IEEE Trans. on Inform. Theory*, IT–45 (1999), 1757–1767.

[31] T. Høholdt, R. Refslund Nielsen. Decoding Hermitian code with Sudan's algorithm, Springer Lecture Notes in Computer Science, Vol 1719, 260–270 (2000).

[32] J. Justesen, K.J. Larsen, H.E. Jensen, A. Havemose, T. Høholdt, Construction and Decoding of a Class of Algebraic Geometry Codes, *IEEE Trans. on Inform. Theory*, IT–35 (1989), 811–821.

[33] J. Justesen, K.J. Larsen, H.E. Jensen, T. Høholdt, Fast decoding of codes from algebraic plane curves, *IEEE Trans. on Inform. Theory*, IT–38 (1992), 111–119.

[34] T. Kailath, Linear Systems,Prentice-Hall,1980.

[35] R. Kötter, A. Vardy, Algebraic soft-decision decoding of Reed-Solomon codes, presented at *ISIT* 2000, Sorrento, June 2000.

[36] M. Kuijper. An algorithm for constructing a minimal partial realization in the multivariable case, *Systems and Control Letters*, 31 (1997), 225–233.

[37] M. Kuijper. Algorithms for decoding and interpolation, *Codes, Systems and Graphical models, Vol 123 IMA Volumes in Mathematics and its Applications*, (2000), 265–282.

[38] J. Little, D. Ortiz, R. Ortiz-Rosado, R. Pablo, and K. Rios-Soto, Some Remarks on Fitzpatrick and Flynn's Gröbner Basis Technique for Padé Approximation,*J. Symb. Comp.*, 35 (2003), 451–461.

[39] P. Loustannau, E. York, On the decoding of Cyclic Codes using Gröbner bases, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, Proc. of AAECC-8*, (1997), 469–483.

[40] J.L. Massey, Shift-register synthesis and BCH decoding, *IEEE Trans. on IT*, Vol 15, Jan (1969), 122-127.

[41] J.L. Massey, M.K. Sain, Codes, automata and continuous systems: Explicit interconnections, *IEEE Trans. on Automatic Control*, Vol 12, No 6 December (1967), 644-650.

[42] F.J. McWilliams, N.J.A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam, 1977.

[43] H.M. Möller, T. Mora, New constructive methods in classical ideal theory, *Journal of Algebra*, Vol 100 (1986), 138–178.

[44] T. Mora, M. Sala, On Gröbner bases of some symmetric ideals and their application to Coding Theory, *J. Symb. Comp.*, 35 (2003) 177-194.

[45] R.R. Nielsen, T. Høholdt, Decoding Reed-Solomon codes beyond half the minimum distance, *Proceedings of an International Conference on Coding Theory, Cryptography and Related Areas*, Guanajuato, April 1998, Springer-Verlag, 2000.

[46] R.R. Nielsen, List decoding of linear block codes, *Phd thesis*, Technical University of Denmark,(September 2001)

[47] H. O'Keeffe, P. Fitzpatrick, Recursive construction of Gröbner bases for the solution of polynomial congruences, *Codes, Systems and Graphical models*, Vol 123 *IMA Volumes in Mathematics and its Applications*, (2000), 299–311.

[48] H. O'Keeffe, P. Fitzpatrick, Gröbner Basis solutions of constrained interpolation problems, *Fourth special issue on linear systems and control, V.D. Blondel, D. Hinrichsen, J. Rosenthal and P. Van Dooren(Eds.), Linear Algebra and its Applications* 351-352, (June 2002), 533-551.

[49] H. O'Keeffe, P. Fitzpatrick, Hard and soft-decision list decoding of 1-point codes as solutions of constrained interpolation problems In Proc. ISIT 2002, Lausanne, Switzerland, June-July 2002.

[50] R. Pellikaan, X.-W. Wu, List decoding of q-ary Reed-Muller codes, Submitted to: IEEE Trans. Inform. Theory, 2003

[51] V.S. Pless, W.C. Huffman (eds), *Handbook of Coding Theory*, Elsevier, Amsterdam, 1998.

[52] W.W. Peterson, E.J. Weldon, Jr., *Error-Correcting Codes*, 2nd ed., MIT, Cambridge, MA, 1972.

[53] E. Popovici, P. Fitzpatrick, Algorithm and Architecture for a Galois Field Multiplicative Arithmetic Processor, *IEEE Trans. on Inform. Theory*, to appear.

[54] O. Pretzel, *Codes and Algebraic Curves*, Oxford Lecture Series in Mathematics and its Applications 8, 1998.

[55] L. Robbiano, Term orderings on the polynomial ring, *Proc. Eurocal '85, Lecture Notes in Computer Science*, No. 204: *Springer*, (1985), 513–517.

[56] J. Rosenthal, J.M. Schumacher, E.V. York, On behaviors and convolutional codes, *IEEE Trans. on Inform. Theory*, IT–42 6 (1996), 1881–1891.

[57] R. Roth, G. Ruckenstein, Efficient decoding of Reed-Solomon codes beyond half the minumum distance, *Manuscript* (1998)

[58] R. Roth, G. Ruckenstein, Efficient decoding of Reed-Solomon codes beyond half the minumum distance, *IEEE Trans. on Inform. Theory* IT–46 (2000), 246–257.

[59] C.J. Rust, G.J. Reid, *Proc. ISACC '97, Maui, W. Küchlin (Ed.) ACM Press*, (1997), 9–16.

[60] M. Sudan, Decoding of Reed-Solomon codes beyond the error correction bound, *J. of Complexity*, 13 (1997), 180–193.

[61] S. Sakata, Finding a Minimal Set of Linear Recurrence Relations Capabale of Generating a Given Finite Two–dimensional Array,*J. of Symbolic Computation*, 5, (1988), 321–337.

[62] S. Sakata, On fast interpolation method for Guruswami-Sudan list decoding of one-point algrebraic-geometry codes, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes, Proc. of AAECC-14,S. Boztas and I. Shparlinski (Eds.), Springer Lecture Notes in Computer Science*, No. 2227: *Springer Verlag*, (2001), 172–181.

[63] C. Shannon, A mathematical theory of communication,*Bell System Tech. J*,27 (1948), 379–423 and 623–656.

[64] Special issue on Algebraic Geometry Codes, *IEEE Trans. on Inform. Theory* IT–41, No 6, (1995)

[65] M.A. Shokrollahi, H. Wasserman, List decoding of alegbraic-geometric codes, *IEEE Trans. on Inform. Theory* IT–45 (1999), 432–437

[66] J.C. Willems, From time series to linear system, Part I, Finite dimensional linear time invariant systems, *Automatica* Vol 22, No 5 (1986), 561–580

[67] J.C. Willems, From time series to linear system, Part II, Exact modelling,*Automatica* Vol 22, No 6 (1986), 675–694

[68] J.C. Willems, From time series to linear system, Part III, Approximate modelling, *Automatica* Vol 23, No 1 (1987), 87–115

[69] J.C. Willems, Paradigms and Puzzles in the Theory of Dynamical Systems, *IEEE Trans. on Automatic Control*, Vol 36, No 3 March (1991), 259–294

[70] C. Wolf, P. Fitzpatrick, Direct division in factor rings, *Electronics Letters*, Vol 38, no 21 (2002).

[71] E.V. York, Algebraic description and construction of error correction codes: A linear systems point of view, *Ph.D thesis*, Univ. of Notre Dame, Indiana (1997)