Shiftless Decomposition and Polynomial-time Rational Summation

J. Gerhard
Maplesoft
57 Erb Street West
Waterloo, ON, N2L 6C2, Canada
jgerhard@maplesoft.com

M. Giesbrecht A. Storjohann E.V. Zima
School of Computer Science
University of Waterloo
Waterloo, ON, N2L 3G1, Canada
{mwg,astorjoh,ezima}@scg.uwaterloo.ca

ABSTRACT

New algorithms are presented for computing the dispersion set of two polynomials over $\mathbb{Q}$ and for *shiftless* factorization. Together with a summability criterion by Abramov, these are applied to get a polynomial-time algorithm for indefinite rational summation, using a sparse representation of the output.

Categories and Subject Descriptors

I.1 [**Symbolic and Algebraic Manipulation**]: Algorithms

General Terms

Algorithms

Keywords

rational summation

1. INTRODUCTION

Let K be a field of characteristic zero and consider the first order linear difference equation

$$(E-1)Y = F, \tag{1}$$

where $F \in K(x)$ and E is the shift operator: $Ef(x) = f(x+1)$ for any $f \in K(x)$. The *indefinite rational summation problem* is to assay if (1) has a rational solution for Y, and if it does not, then to extract an additive rational part $R \in K(x)$ from the solution such that the remaining part satisfies a simpler difference equation with denominator of lowest possible degree. This gives an equality

$$\sum_x F = R + \sum_x H \tag{2}$$

where the denominator of H has lowest possible degree. One contribution of this paper is a new algorithm for computing the rational part R and remainder H.

It is well known that the size of R may be exponentially large as a function of the size of F. Consider the case $K = \mathbb{Q}$. The algorithm we present here is distinguished from previous algorithms because it always returns a compact representation of R in time polynomial in the input size. For example, if $F = -1000/x(x+1000)$ our algorithm returns the answer

$$\sum_x \frac{-1000}{x(x+1000)} = \mathrm{LQ}(E^{1000}-1, E-1)\,\frac{1}{x} \tag{3}$$

where LQ denotes the left quotient in the noncommutative ring of linear recurrence operators over $\mathbb{Q}[x]$. In general, the solution may be written as a sum of such unevaluated left quotients. For the example in (3) note that

$$\mathrm{LQ}(E^{1000}-1, E-1) = E^{999} + E^{998} + \cdots + E + 1, \tag{4}$$

so the expanded form of the solution has one thousand terms. More spectacular examples exist, but the representation using a sum of LQ's returned by our algorithm will always be sparse. The key point is that any potential exponential expression swell is avoided until the final (and optional) left divisions by $E-1$. As an immediate corollary of our algorithm we get a polynomial-time test to assay if F is rational summable, i.e., if $H = 0$.

If the expanded form of the output is desired (e.g., as in (4)) this can be computed easily by a left division by $E-1$ in time polynomial in the size of the expanded output. Note that although the result of the left division may be an operator of high order it may happen to be sparse, e.g.,

$$\mathrm{LQ}(E^{1000} - E^{999} + E - 1, E-1) = E^{999} + 1. \tag{5}$$

Other algorithms can suffer from intermediate expression swell even if the final output ends up to be small, since their intermediate results are polynomials of degree about the order of the linear recurrence operator (see below). More examples are given in Section 7.

Outline

As in a number of previous approaches, a key step in our algorithm is to compute the *auto-dispersion set* of a polynomial: $\mathrm{DS}(g,g) = \{h \in \mathbb{Z} \mid \deg(\gcd(E^h g, g)) > 0\}$. One of the contributions here, which is of independent interest, is to present a new, fast probabilistic algorithm for computing the auto-dispersion set when $K = \mathbb{Q}$. See Section 6. Actually, the algorithm solves the more general problem of computing the *dispersion set* of two polynomials: $\mathrm{DS}(f,g) = \{h \in \mathbb{Z} \mid \deg(\gcd(E^h f, g)) > 0\}$. These definitions of auto-dispersion set and dispersion set differ slightly from the corresponding notions in the literature in that they include negative shifts as well.

A central ingredient of our approach is a decomposition for univariate polynomials which we call *shiftless decomposition*. We define this notion in Section 2 and give an algorithm to compute it in Section 3. The algorithm requires as input the auto-dispersion set and is applicable over any field of characteristic zero. The algorithm employs factor refinement and reduces the problem to shift and gcd computations. Given the auto-dispersion set of a $g \in K[x]$, the algorithm computes a shiftless decomposition of g with $O((\deg g)^4)$ field operations from K.

Our rational summation algorithm in Section 5 is based on the criterion for rational summability in [3]. For clarity, the criterion was described there using the full factorization of the denominator of F in $\bar{K}[x]$, where $\bar{K}$ is the algebraic closure of K. In fact, the criterion works also for a full factorization in $K[x]$. In Section 4 we recall the criterion and note that it works with a shiftless decomposition as well.

Previous approaches

The algorithmic treatment of rational summation problems started with the works of Abramov [1, 2]. There were a number of algorithms and improvements developed over the following years, see for example [3, 12, 15, 16, 9]. In particular [16] gives a complete overview of these algorithms and improvements to them.

Let $F = f/g \in K(x)$, with $f, g \in K[x] \setminus \{0\}$ coprime. Using division with remainder to split off the polynomial part, which can be summed using, e.g., conversion to the falling factorial basis (see [6] §23.1), we may assume that $\deg f < \deg g$, so that F is *proper*. Let ρ be the *dispersion* of F, the maximal integer distance between roots of the denominator g. If $\rho = 0$ then we take $R = 0$ and $H = F$ in (2), see [1, 3, 16]. In fact, the condition that the denominator of H has minimal degree is equivalent to the dispersion of H being zero. Now, let $\rho > 0$. All algorithms mentioned above carefully avoid factorization in $K[x]$ and fall into one of the following two categories.

- **Iterative** (Hermite reduction analogous) algorithms will start with $R = 0$ and $H = F$ and decrease the dispersion of H by one at each iteration, reducing the non-rational part H and growing the rational part R. The number of iterations is equal to ρ, see [2].

- **Linear algebra based** (Ostrogradsky analogous) algorithms first build universal denominators u and v such that the denominator of R will divide u and the denominator of H will divide v. Then, the problem is reduced to solving a system of linear equations with size $\deg u + \deg v$, see [15, 3, 16]. Since $\deg u \geq \rho$ the choice of u of the lowest possible degree is obviously crucial here.

In both classes of algorithms if $\rho \gg \deg g$ the cost of rational function summation depends essentially on the value of ρ. Consider the following examples:

$$\sum_x \frac{-2\,x + 999}{(x+1)\,(x-999)\,x\,(x-1000)} = \frac{1}{x\,(x-1000)}, \qquad (6)$$

$$\sum_x \frac{x^3 - 1998\,x^2 + 996999\,x + 999999}{(x+1)\,(x-999)\,x\,(x-1000)} = \frac{1}{x\,(x-1000)} + \sum_x \frac{1}{x}. \qquad (7)$$

The dispersion of the summand in both of these examples is 1001. On the one hand, iterative algorithms will require about 1001 steps of polynomial gcd computations. On the other hand, the universal denominator constructed by the linear algebra based algorithms will have degree about 1001. In general, ρ may be as large as the magnitude of the trailing coefficient of g. Thus, the cost of the iterative and linear algebra based algorithms for computing a decomposition as in (2) may be exponential in the size of the input.

In [9] an algorithm is presented for computing a sharp bound for $\deg u$ in the case when F is rational summable, i.e. $H = 0$. This algorithm uses full factorization of g over $K[x]$.

Another approach would be to apply Gosper's algorithm for hypergeometric summation [8]. In fact, for (6) Gosper's algorithm, together with the improvement from [11] for the case of rational input, is potentially fast but it only works in the case when F is rational summable. That is, the algorithm is not applicable for the example shown in (7) since $H = 1/x \neq 0$.

Summarizing, the key features of our algorithm, which distinguish it from previous approaches, are:

- Using a sparse representation of the output in terms of left quotients, the algorithm works in deterministic polynomial time, with the exception of the dispersion computation stage, which is probabilistic polynomial time.

- Since the size of the expanded output is exponential in the input size in general, no polynomial-time algorithm giving the expanded output exists. In cases where the size of the expanded output is polynomial in the input size, however, our method obtains the expanded output in polynomial time as well.

- A trivial modification of our algorithm yields a test for rational summability which executes in a polynomial number of operations. This test does not rely on any special type of data representation.

- Over the field of rational numbers, our algorithm does not require full factorization, but only factorization modulo a prime and gcd computations; see Sections 3 and 6.

2. SHIFTLESS DECOMPOSITION

Let K be a field of characteristic zero. Polynomials $f, g \in K[x]$ are *coprime* if $\gcd(f, g) = 1$ (notation: $f \perp g$) and *shift coprime* if $\gcd(f, E^h g) = 1$ for all $h \in \mathbb{Z}$ (notation: $f \perp_{\mathrm{S}} g$). Let $g \in K[x]$ be nonzero. Suppose

$$g = c \prod_{i=1}^{v} \prod_{j=1}^{n_i} E^{h_{ij}} g_i^{e_{ij}} \qquad (8)$$

for some choices $c \in K$, $g_i \in K[x]$, $h_{ij} \in \mathbb{Z}$ and $e_{ij} \in \mathbb{Z}_{\geq 1}$.

Definition 1. *A decomposition as in (8) is a* shiftless decomposition *over* $K[x]$ *if*

- *g_i is monic and non-constant;*
- *$g_i \perp_S g_j$ for $i \neq j$;*
- *g_i is squarefree and the auto-dispersion set of g_i is $\{0\}$;*
- *$0 = h_{i1} < h_{i2} < \ldots < h_{in_i}$;*

for $1 \leq i \leq v$.

Example 2. *Let $g = x(x^2-1) \in \mathbb{Q}[x]$. Then*

$$g = \overbrace{(x)}^{g_1}\overbrace{(x^2-1)}^{g_2}$$

is not a shiftless decomposition since g_2 has auto-dispersion set $\{-2, 0, 2\}$.

Example 3. *Let $g = (x^2+x+1)(x+1)(x+3) \in \mathbb{Q}[x]$. Then*

$$g = \overbrace{(x^2+x+1)}^{g_1}\overbrace{(x+1)}^{g_2}\overbrace{(x+3)}^{g_3} \qquad (9)$$

in which case $c = 1$, $v = 3$ and $n_1 = n_2 = n_3 = 1$. As this example shows, we can always trivially choose $n_i = 1$ for all i . A different decomposition of g based on the same factorization is given by

$$g = \overbrace{(x^2+x+1)}^{g_1}\overbrace{(x+1)}^{g_2}\overbrace{(x+3)}^{E^2 g_2} \qquad (10)$$

in which case $c = 1$, $v = 2$, $n_1 = 1$ and $n_2 = 2$.

The decomposition shown in (10) is shiftless *while that shown in (9) is not.*

On the one hand, a given factorization of a polynomial does not necessarily induce a shiftless decomposition, cf. Example 2. On the other hand, if a particular factorization does induce a shiftless decomposition, then this is unique and is determined by partitioning the factors into a minimal number of *shift classes* — two factors belonging to the same shift class precisely when they are integral shifts of each other, cf. Example 3.

A given polynomial may have non-trivially different shiftless decompositions based on different factorizations. Some of the decompositions may be coarser (v is smaller) or more refined (v is larger). The most refined shiftless decomposition of a given polynomial is based on the full factorization in $K[x]$. These ideas are explained by the following examples.

Example 4. *Suppose $g \in K[x]$ admits the shiftless decomposition*

$$g = \Big(\prod_{h\in\{0,5\}} E^h g_1\Big)\Big(\prod_{h\in\{0,5\}} E^h g_2\Big)\Big(\prod_{h\in\{0,2,7\}} E^h g_3\Big), \qquad (11)$$

where g_1, g_2 are irreducible but, suppose, the full factorization in $K[x]$ of g_3 is $g_3 = g_{31}g_{32}$. The decomposition shown in (11) has three shift classes. On the one hand, we can combine the shift classes corresponding to g_1 and g_2 to get the less refined shiftless decomposition

$$g = \Big(\prod_{h\in\{0,5\}} E^h(g_1g_2)\Big)\Big(\prod_{h\in\{0,2,7\}} E^h g_3\Big) \qquad (12)$$

which has only two shift classes. On the other hand, we can split the shift class corresponding to g_3 to get a more refined shiftless decomposition

$$\begin{aligned} g \;=\; & \Big(\prod_{h\in\{0,5\}} E^h g_1\Big)\Big(\prod_{h\in\{0,5\}} E^h g_2\Big) \\ & \Big(\prod_{h\in\{0,2,7\}} E^h g_{31}\Big)\Big(\prod_{h\in\{0,2,7\}} E^h g_{32}\Big) \end{aligned} \qquad (13)$$

which has four shift classes. The decomposition in (12) is the coarsest while that in (13) is the most refined shiftless decomposition of g.

The previous examples dealt with decompositions of squarefree polynomials.

Example 5. *Let*

$$f = (x+1)^2(x+3)^2(x^2+1)^2(x^2+4x+5)^2.$$

Then both

$$f = \overbrace{(x+1)^2(x^2+1)^2}^{g_1^2}\overbrace{(x+3)^2(x^2+4x+5)^2}^{E^2 g_1^2}$$

and

$$f = \overbrace{(x+1)^2}^{g_1^2}\overbrace{(x+3)^2}^{E^2 g_1^2}\overbrace{(x^2+1)^2}^{g_2^2}\overbrace{(x^2+4x+5)^2}^{E^2 g_2^2}$$

are shiftless decompositions of f.

Example 6. *Let $f = (x+1)^2(x+3)^2(x^2+1)^2(x^2+4x+5)$. Then the coarsest shiftless decomposition of f has $v = 2$, i.e.*

$$f = \overbrace{(x+1)^2}^{g_1^2}\overbrace{(x+3)^2}^{E^2 g_1^2}\overbrace{(x^2+1)^2}^{g_2^2}\overbrace{(x^2+4x+5)}^{E^2 g_2}.$$

Although $(h_{11}, h_{12}) = (h_{21}, h_{22}) = (0, 2)$, however, the shift classes corresponding to g_1 and g_2 cannot be combined because $(e_{11}, e_{12}) = (2, 2)$ while $(e_{21}, e_{22}) = (2, 1)$.

3. SHIFTLESS FACTORIZATION

Let K be a field of characteristic zero. Our algorithm for shiftless factorization employs factor refinement, and in particular the construction of a gcd-free basis. Let $A = \{a_1, a_2, \ldots, a_m\}$ be a nonempty set of m polynomials from $K[x]$. Let $\mathcal{B} = \{b_1, b_2, \ldots, b_n\}$ be a set of monic and non-constant polynomials. Following [5], the set $\mathcal{B}$ is a *gcd-free basis* for A if

(a) $b_i \perp b_j$ for all $i \neq j$; and

(b) there exist mn non-negative integers e_{ij} such that $a_i = \prod_{1\leq j\leq n} b_j^{e_{ij}}$ for all $i, 1 \leq i \leq m$.

Let the sum of the degrees of the entries in A be equal to d. In [5] an algorithm `GcdFreeBasis` is described that computes a gcd-free basis with $O(d^2)$ field operations from K. The algorithm uses only gcd and exact division in $K[x]$. Starting with the set $\{a_1, a_2, \ldots, a_m\}$, the algorithm repeatedly extracts two elements $\{a, b\}$, and replaces them with the elements of $\{a/g, g, b/g)\} \setminus \{1\}$, where $g = \gcd(a, b)$. This is repeated until the set satisfies property (a). Although a given set A may have more than one gcd-free basis, the basis obtained using only gcds and exact divisions is unique. A characterization of this uniqueness is given in [4, Theorem 3]. The observation below follows as a corollary.

LEMMA 7. *Suppose $\mathcal{B}$ and $\overline{\mathcal{B}}$ are gcd-free bases of the same set of polynomials A. If $\mathcal{B}$ is the basis obtained from A using only gcds and exact divisions, and $\mathcal{B}$ is also a gcd-free basis of $\overline{\mathcal{B}}$, then $\overline{\mathcal{B}} = \mathcal{B}$.*

We present our algorithm first for a squarefree $g \in K[x]$. The extension to a possibly non-squarefree polynomial will be discussed below. Suppose g has a shiftless decomposition

$$g = c\prod_{i=1}^{v}\prod_{j=1}^{n_i} E^{h_{ij}} g_i.$$

By combining shift classes and coarsening the factorization if necessary we may assume without loss of generality that

$$(h_{i1}, \ldots, h_{in_i}) \neq (h_{j1}, \ldots, h_{jn_j}) \tag{14}$$

holds for all $1 \leq i < j \leq v$. This condition gives the coarsest possible shiftless decomposition of g (v is minimal). Let $\mathcal{A}$ be the auto-dispersion set of g.

THEOREM 8. *Let $\mathcal{B}$ be the gcd-free basis of*

$$\{\gcd(g, E^h g)\}_{h\in\mathcal{A}}$$

obtained using only gcd and exact division. Then

$$\mathcal{B} = \{E^{h_{ij}} g_i\}_{1\leq i\leq v, 1\leq j\leq n_i}.$$

PROOF. Apply Lemma 7 with $\overline{\mathcal{B}} = \{E^{h_{ij}} g_i\}_{1\leq i\leq v, 1\leq j\leq n_i}$. It will be sufficient to show that for any two distinct factors $f_1, f_2 \in \{E^{h_{ij}} g_i\}_{1\leq i\leq v, 1\leq j\leq n_i}$ there exists an $h \in \mathcal{A}$ such that exactly one of the f_i divides $\gcd(g, E^h g)$ and the other is relatively prime to $\gcd(g, E^h g)$. Suppose f_1 and f_2 belong to the same shift class: without loss of generality say that $f_1 = E^{s_1} g_1$, $f_2 = E^{s_2} g_1$ and $s_1 > s_2$. We can then choose $h = s_1$, in which case $f_1 | \gcd(g, E^h g)$ and $f_2 \perp \gcd(g, E^h g)$. Now suppose that f_1 and f_2 belong to different shift classes: without loss of generality say $f_i \in E^{s_i} g_i$, $i = 1, 2$. Let $\mathcal{H}_i = \{s_i - h_{ij}\}_{1\leq j\leq n_i}$, $i = 1, 2$. Then $\mathcal{H}_i$ is the set of all shifts s such that $f_i | \gcd(g, E^s g)$, $i = 1, 2$. Because of assumption (14), we have $\mathcal{H}_1 \neq \mathcal{H}_2$ so we can choose an $h \in \mathcal{H}_1$ such that $h \notin \mathcal{H}_2$ or vice versa. □

We now extend the ideas above to work for a possibly non-squarefree $f \in K[x]$. Suppose f has a shiftless decomposition

$$f = c\prod_{i=1}^{v}\prod_{j=1}^{n_i} E^{h_{ij}} g_i^{e_{ij}}.$$

By combining shift classes if necessary, we may assume without loss of generality that at least one of

$$\begin{aligned}&(h_{i1}, \ldots, h_{in_i}) \neq (h_{j1}, \ldots, h_{jn_j}), \quad \text{or}\\ &(e_{i1}, \ldots, e_{in_i}) \neq (e_{j1}, \ldots, e_{jn_j})\end{aligned} \tag{15}$$

holds for all $1 \leq i < j \leq v$. This condition gives the coarsest possible shiftless decomposition of f (v minimal). Let $f_1 = \texttt{SquareFreePart}(f)$, and factor f as $f = f_1 f_2 \cdots f_k$ where $f_i = \texttt{SquareFreePart}(f/(f_1 \cdots f_{i-1}))$, $2 \leq i \leq k$. Let $\mathcal{A}$ be the auto-dispersion set of f. In our algorithm we will use the fact that $\mathcal{A}$ is also equal to the auto-dispersion set of f_1. The proof of the next theorem is similar to that of Theorem 8.

THEOREM 9. *Let $\mathcal{B}$ be the gcd-free basis of*

$$\{\gcd(f_1, E^h f_i\}_{h\in\mathcal{A}, 1\leq i\leq k}$$

computed using only gcd and exact division. Then

$$\mathcal{B} = \{E^{h_{ij}} g_i\}_{1\leq i\leq v, 1\leq j\leq n_i}.$$

The correctness of Algorithm `ShiftlessFactorization` follows from Theorem 9. In Phase 3 of the algorithm we compute the gcd-free basis of $\{\gcd(f_1, E^h f_i\}_{h\in\mathcal{A}, 1\leq i\leq k}$ iteratively. We use the fact that, for $A, B \subset K[x]$,

$$\begin{aligned}&\texttt{GcdFreeBasis}(\texttt{GcdFreeBasis}(A) \cup \texttt{GcdFreeBasis}(B))\\ &\qquad = \texttt{GcdFreeBasis}(A \cup B).\end{aligned}$$

```
Algorithm: ShiftlessFactorization
Input:    ▸ $f \in K[x]$, $\deg f = n$.
Output:   ▸ a shiftless decomposition of $f$.
(1) $g := f$;
    for $i$ while $g \neq 1$ do
        $f_i :=$ SquareFreePart($g$);
        $g := g/f_i$
    od;
(2) $\mathcal{A} :=$ AutoDispersion($f_1$);
(3) $\mathcal{B} := \{\}$;
    for $h \in \mathcal{A}$ do
        $\mathcal{B} := \mathcal{B} \cup \{\gcd(f_1, E^h f_i)\}_{1\leq i\leq k}$;
        $\mathcal{B} :=$ GcdFreeBasis($\mathcal{B}$)
    od;
(4) Partition $\mathcal{B}$ into shift equivalence classes to obtain $f_1 = \prod_{i=1}^{v}\prod_{j=1}^{n_i} E^{h_{ij}} g_i$.
(5) Output $f = \prod_{i=1}^{v}\prod_{j=1}^{n_i} E^{h_{ij}} g_i^{e_{ij}}$ where $e_{ij}$ is chosen
    maximal such that $E^{h_{ij}} g_i^{e_{ij}}$ is a divisor of $f_{e_{ij}}$, for $1 \leq$
    $i \leq v$, $1 \leq j \leq n_i$.
```

THEOREM 10. *The algorithm `ShiftlessFactorization` works as stated. If $\deg f = n$ then the algorithm uses $O(n^4)$ field operations, plus the cost of computing the auto-dispersion set of f.*

PROOF. Step 3 will dominate the cost. The product of all entries in $\mathcal{B}$ after each iteration of the loop will be an associate of f_1. The sum of degrees of entries in $\mathcal{B}$ for each basis refinement will be $\leq 2\deg f$, giving a total cost for step 3 of $O(n^2|\mathcal{A}|)$. Since $|\mathcal{A}| \leq n^2$ the result follows. □

4. ABRAMOV'S CRITERION

Let K be a field of characteristic zero. As in the introduction, and following [3], we consider the difference equation

$$(E - 1)Y = F, \tag{16}$$

where F is a non-zero proper rational function in x. A solution to the rational summation problem consists of proper $R, H \in K(x)$ such that

$$\sum_x F = R + \sum_x H, \tag{17}$$

where H is a rational function whose denominator has dispersion zero. We temporarily replace the coefficient field K by its algebraic closure $\overline{K}$. The full partial fraction decomposition of F has the form

$$F = \sum_{i=1}^{m}\sum_{k=1}^{t_i} \frac{\beta_{ik}}{(x - \alpha_i)^k}, \tag{18}$$

with $\alpha_i, \beta_{ik} \in \overline{K}$. Write $\alpha_i \sim \alpha_j$ if $\alpha_i - \alpha_j$ is an integer. Obviously, $\sim$ is an equivalence relation in the set $\{\alpha_1, \ldots, \alpha_m\}$. Each of the corresponding equivalence classes has a largest element in the sense that the other elements of the class are obtained by subtracting positive integers from it. Let $\alpha_1, \ldots, \alpha_v$ be the largest elements of all the classes ($v \leq m$). Then (18) can be rewritten as

$$F = \sum_{i=1}^{v} \sum_{k=1}^{l_i} M_{ik}(E) \frac{1}{(x - \alpha_i)^k}. \tag{19}$$

Here $M_{ik}(E)$ is a linear difference operator with constant coefficients (a polynomial in E over $\overline{K}$). Let F have the form (19) and suppose that (16) has a solution $R \in K(x)$. The rational function R can be written in a form analogous to (19):

$$\sum_{i=1}^{v} \sum_{k=1}^{l_i} L_{ik}(E) \frac{1}{(x - \alpha_i)^k}. \tag{20}$$

This presentation is unique and therefore

$$(E - 1)L_{ik}(E) = M_{ik}(E). \tag{21}$$

Theorem 11 (Rational Summation Criterion 1). *A necessary and sufficient condition for existence of a rational solution of (16) is that for all $i = 1, \ldots, v$; $k = 1, \ldots, l_i$ there is an operator $L_{ik}(E)$ such that (21) holds. This is also equivalent to the condition that for all $i = 1, \ldots, v$; $k = 1, \ldots, l_i$ the sum of coefficients of the operator $M_{ik}(E)$ is equal to zero.*

If the conditions for the above theorem are met then equation (16) has the solution (20) and all other rational solutions of (16) can be obtained by adding arbitrary constants. Abramov [3] used this criterion to describe the structure of a universal denominator. We remark that his algorithm for rational summation did not use any factorization of the denominator of F. In fact, the criterion also works for coarser decompositions of the denominator such as the full factorization in $K[x]$ and, more importantly, the shiftless decomposition.

If at least one of operators $M_{ik}(E)$ is not divisible by $E - 1$ then (16) has no rational solution. We want then to construct a decomposition as in (17). Consider one term from (19),

$$M_{ik}(E) \frac{1}{(x - \alpha_i)^k}, \; k \geq 1,$$

and compute the quotient $L_{ik}(E)$ and the remainder r_{ik}:

$$M_{ik}(E) = (E - 1)L_{ik}(E) + r_{ik}, \quad r_{ik} \in \overline{K}. \tag{22}$$

Write the right-hand side of (17) for this term in the form

$$L_{ik}(E) \frac{1}{(x - \alpha_i)^k} + \sum_x \frac{r_{ik}}{(x - \alpha_i)^k}. \tag{23}$$

This gives a solution to the decomposition problem for this single term, since the denominator of the rational function in the indefinite sum has dispersion zero. If there are several $M_{ik}(E)$ not divisible by $E - 1$, then the non-rational part

$$H = \sum_x \sum_{i=1}^{v} \sum_{k=1}^{l_i} \frac{r_{ik}}{(x - \alpha_i)^k}$$

has the lowest possible degree for the denominator, since it also has dispersion zero.

Summability criterion using a shiftless decomposition

Consider (17), assuming again that F is a proper rational function. Suppose the denominator of F has a shiftless decomposition

$$\prod_{i=1}^{v} \prod_{j=1}^{n_i} E^{h_{ij}} g_i^{e_{ij}}.$$

The unique partial fraction decomposition of F with respect to this decomposition of the denominator is

$$F = \sum_{i=1}^{v} \sum_{j=1}^{n_i} \sum_{k=1}^{e_{ij}} \frac{f_{ijk}}{E^{h_{ij}} g_i^k}.$$

Let $l_i = \max_j e_{ij}$ and specify that $f_{ijk} = 0$ in case $k > e_{ij}$. Then interchanging the second and third summation gives

$$F = \sum_{i=1}^{v} \sum_{k=1}^{l_i} \left(\sum_{j=1}^{n_i} \frac{f_{ijk}}{E^{h_{ij}} g_i^k} \right) = \sum_{i=1}^{v} \sum_{k=1}^{l_i} M_{ik}(E) \frac{1}{g_i^k}, \tag{24}$$

where $M_{ik}(E) = \sum_{j=1}^{n_i} f_{ijk} E^{h_{ij}}$. Note that the coefficients f_{ijk} are in $K[x]$ with $0 \leq \deg(f_{ijk}) < \deg g_i$. Let F have the form (24) and suppose that (16) has a solution $R \in K(x)$. Because the denominators of the reduced fractions R and $(E - 1)R$ have the same shift classes with the same multiplicities, the rational function R can be written in a form analogous to (24):

$$\sum_{i=1}^{v} \sum_{k=1}^{l_i} L_{ik}(E) \frac{1}{g_i^k}. \tag{25}$$

This presentation is unique because the decomposition is shiftless and therefore

$$(E - 1)L_{ik}(E) = M_{ik}(E). \tag{26}$$

Theorem 12 (Rational Summation Criterion 2). *A necessary and sufficient condition for existence of a rational solution of (16) is that for all $i = 1, \ldots, v$; $k = 1, \ldots, l_i$ there is an operator $L_{ik}(E)$ such that (26) holds. This is also equivalent to the condition that for all $i = 1, \ldots, v$; $k = 1, \ldots, l_i$ the remainder from left division of the operator $M_{ik}(E)$ by $E - 1$ is equal to zero.*

Let $M_{ik}(E) = a_p E^p + a_{p-1} E^{p-1} + \ldots + a_1 E + a_0$. Then the left remainder from the division of $M_{ik}(E)$ by $E - 1$ is simply $r_{ik} = E^{-p} a_p + E^{-(p-1)} a_{p-1} + \ldots + E^{-1} a_1 + a_0$. The summability criterion states that this last polynomial must be identically equal to zero, for all i, k.

If at least one of the operators $M_{ik}(E)$ is not divisible by $E - 1$, then writing

$$M_{ik}(E) = (E - 1)L_{ik}(E) + r_{ik}, \quad r_{ik} \in K[x], \tag{27}$$

we obtain the nonrational part in (17)

$$H = \sum_x \sum_{i=1}^{v} \sum_{k=1}^{l_i} \frac{r_{ik}}{g_i^k},$$

which has dispersion zero.

Note that although $M_{ik}(E)$ is a sparse operator of order h_{in_i}, the left quotient $L_{ik}(E)$ may be a dense operator of order $h_{in_i} - 1$.

5. AN ALGORITHM FOR RATIONAL SUMMATION

```
Algorithm: RatSum
Input:   ▸ $f, g \in K[x]$, $f \perp g$, $\deg f < \deg g$.
Output:  ▸ $H, R \in K(x)$ as in (17) with denominator of
           $H$ of lowest possible degree.
(1) ShiftlessFactorization($g$)
        $\to g = \prod_{i=1}^{v} \prod_{j=1}^{n_i} E^{h_{ij}} g_i^{e_{ij}}$;
(2) Compute a partial fraction decomposition
        $\frac{f}{g} = \sum_{i=1}^{v} \sum_{k=1}^{l_i} M_{ik}(E) \frac{1}{g_i^k}$ where $M_{ik}(E) = \sum_{j=1}^{n_i} f_{ijk} E^{h_{ij}}$.
(3) for $i$ to $v$ do
      for $k$ to $l_i$ do
        # Write $M_{ik}(E) = a_p E^p + a_q E^q + \ldots + a_s E^s$.
        $r_{ik} := E^{-p} a_p + E^{-q} a_q + \ldots + E^{-s} a_s$;
        $\tilde{M}_{ik} := M_{ik} - r_{ik}$
      od
    od;
(4) $H := \sum_{i=1}^{v} \sum_{k=1}^{l_i} \frac{r_{ik}}{g_i^k}$;
(5) Output
        $\sum_{i=1}^{v} \sum_{k=1}^{l_i} \mathrm{LQ}(\tilde{M}_{ik}(E), E-1) \frac{1}{g_i^k} + \sum_x H$
```

THEOREM 13. *The algorithm* **RatSum** *works as stated. If* $\deg g = n$ *then the algorithm uses* $O(n^4)$ *field operations from* K*, plus the cost of computing the auto-dispersion set of* g*.*

PROOF. Correctness of algorithm `RatSum` follows from the criterion of Theorem 12 and the discussion following it. Since the dispersion of H is zero, the denominator of H will have lowest possible degree. □

Now suppose $K = \mathbb{Q}$ and assume without loss of generality the input $f, g \in \mathbb{Z}[x]$. In the next section we give a fast algorithm for computing the auto-dispersion set of g. It is clear the algorithm can be made deterministic and still have running time bounded by $(\log \|f\| + \log \|g\| + \deg g)^{O(1)}$ bit operations, where the (max) norm of $f = \sum_{0 \le i \le n} f_i x^i \in \mathbb{Z}[x]$ is defined as $\|f\| = \max_i |f_i|$. The same is true for all other steps in algorithms `RatSum` and `ShiftlessFactoriziation`. This gives the following

COROLLARY 5.1. *Let* $K = \mathbb{Q}$ *and* $f, g \in \mathbb{Z}[x]$ *be valid input to algorithm* **RatSum**. *The algorithm can be implemented to be deterministic and use* $(\log ||f||_\infty + \log ||g||_\infty + \deg g)^{O(1)}$ *bit operations.*

6. COMPUTING THE DISPERSION SET OVER $\mathbb{Z}[X]$

In this section we present an algorithm for computing the dispersion set of two integer polynomials. A similar algorithm for integer polynomials is given by the first author in [7], Section 7.2. The idea is related to that of Man & Wright [13], who describe an algorithm for dispersion based on polynomial factorization in $\mathbb{Z}[x]$. The main idea of their algorithm is that if $\gcd(f(x+h), g(x))$ is non-constant for some $h \in \mathbb{Z}$, then there exist irreducible factors $\bar{f} \in \mathbb{Z}[x]$ of f and $\bar{g} \in \mathbb{Z}[x]$ of g such that $\bar{f}(x+h) = \bar{g}(x)$. Since irreducible factors remain irreducible under a shift, this condition is easily tested if the irreducible factors are given. Our algorithm uses a similar idea, but only requires a (faster) p-adic factorization of the input polynomials for a small prime p.

We first note that the size of any $h \in \mathrm{DS}(f, g)$ is at most $|f(0)| + |g(0)|$, since $\bar{f}(x+h) = \bar{g}(x)$, so h is an integer root of $\bar{f}(y) - \bar{g}(0)$, and hence is either 0 or a divisor of the constant term. We define the (max) norm of $f = \sum_{0 \le i \le n} f_i x^i \in \mathbb{Z}[x]$ as $\|f\| = \max_i |f_i|$.

```
Algorithm: pDispersionSet
Input:   ▸ $f, g \in \mathbb{Z}[x]$;
Output:  ▸ a set $S \supseteq \mathrm{DS}(f, g)$, $S \subseteq \mathbb{Z}$;
(1) Let $n := \max\{\deg f, \deg g\}$;
(2) Let $f :=$ SquareFreePart($f$), $g :=$ SquareFreePart($g$);
(3) Choose a prime $p > n$ not dividing the leading coefficients or discriminants of either $f$ or $g$;
(4) Factor
        $f \equiv a \cdot f_1 \cdots f_k \bmod p^\mu$
            $f_i \in \mathbb{Z}_{p^\mu}[x]$ monic, irreducible, $a \in \mathbb{Z}_{p^\mu}$
        $g \equiv b \cdot g_1 \cdots g_\ell \bmod p^\mu$
            $g_j \in \mathbb{Z}_{p^\mu}[x]$ monic, irreducible, $b \in \mathbb{Z}_{p^\mu}$
    where $\mu > \log_p(2(|f(0)| + |g(0)|))$;
(5) For $1 \le i \le k$ assume $f_i = x^{d_i} + f_{i,d_i-1} x^{d_i-1} + \cdots + f_{i1} x + f_{i0}$; Let $\bar{f}_i := f_i(x - f_{i,d_i-1}/d_i) \bmod p^\mu$;
(6) For $1 \le j \le \ell$ assume $g_j = x^{e_j} + g_{j,e_j-1} x^{e_j-1} + \cdots + g_{j1} x + g_{j0}$; Let $\bar{g}_j := g_j(x - g_{j,e_j-1}/e_j) \bmod p^\mu$;
(7) Find $S := \{(i, j) : \bar{f}_i = \bar{g}_j\}$;
(8) Output
        $\mathrm{DS}_{p^\mu}(f, g) := \{g_{j,e_j-1}/e_j - f_{i,d_i-1}/d_i : (i, j) \in S\}$
                $\subseteq \{-p^\mu + 1, \ldots, p^\mu - 1\}$.
```

THEOREM 14. *The algorithm* **pDispersionSet** *works as stated. If* $\deg f, \deg g \le n$ *and* $\|f\|, \|g\| \le \beta$ *then the algorithm uses* $O(n^3 \log^2 n + n^2 \log n \log^2 \beta)$ *bit operations.*

PROOF. This algorithm relies on the fact that if there is an $h \in \mathrm{DS}(f, g)$, then there exists a $\bar{g} \in \mathbb{Z}[x]$ dividing g and $\bar{f} \in \mathbb{Z}[x]$ dividing f such that $\bar{f}(x+h) = \bar{g}(x)$. This relation holds modulo p^μ as well, though $\bar{f}$ and $\bar{g}$ may factor further modulo p^μ. Note, however, that $\mathrm{DS}_{p^\mu}(f, g)$ may be larger than $\mathrm{DS}(f, g)$.

In step (3) we choose a prime $p > n$ not dividing the discriminant of f and g. It is easily shown that $\mathrm{disc}(f) = \mathrm{res}(f, f') \le 2^{2n} n^{3n} \|f\|^{2n}$, and that the condition is that our prime not divide a number $w \le 2^{4n} n^{6n} \|f\|^{2n+1} \|g\|^{2n+1}$, the product of the leading coefficients and discriminants of f and g. Since w has at most $\log_2 w = O(n(\log n + \log \|f\| + \log \|g\|))$ many prime factors, a random prime less than $2 \log w \log \log w$ will not divide w with high probability (see [14]). The selected prime will have $O(\log n + \log \log \beta)$ bits, That p does not divide the discriminant of f or g is quickly checked by factorization modulo p, which is required in step (4).

In step (4) we factor f and g modulo p (using Berlekamp's algorithm) and lift to a factorization modulo p^μ. In steps (5) and (6) we normalize each f_i and g_j to $\bar{f}_i$ and $\bar{g}_j$ respectively, whose second highest coefficient is 0. For each f_i, the unique shift $x \mapsto x - f_{i,d_i-1}/d_i$ ensures $\bar{f}_i := f_i(x - f_{i,d_i-1}/d_i)$ has second highest coefficient zero. Any shift is an automorphism of $\mathbb{Z}[x]$, so clearly if $\bar{f}_i = \bar{g}_j$ then f_i and g_j are shift equivalent modulo p^μ. Conversely, since the shift which ensures the second highest coefficient is zero always exists and is unique, if f_i and g_j are shift equivalent, then $\bar{f}_i$ must equal $\bar{g}_j$ (the shift from f_i to g_j can be composed with the shift from g_j to $\bar{g}_j$ to get a shift from f_i to $\bar{g}_j$). In step (8) we determine the dispersion set over $\mathbb{Z}_{p^\mu}$.

The cost of the algorithm follows easily. See [6], Theorem 14.32 and 15.18. $\square$

The above algorithm returns a superset of the dispersion set, which is sufficient for our intended application in Algorithm RatSum. To determine if some $h \in \mathrm{DS}_{p^\mu}(f,g)$ actually has $\deg\gcd(f(x+h),g(x)) \geq 1$ (i.e., $h \in \mathrm{DS}(f,g)$), we can test it directly by computing the resultant $R = \mathrm{res}(f(x+h),g(x))$. The shifted polynomial $f(x+h)$ satisfies (crudely) $\|f(x+h)\| \leq n\beta 2^n h^n$ and $\log\|f(x+h)\| = O(n\log\beta)$. Thus

$$|R| \leq (2n)^{2n}(n\beta 2^n h^n)^n \beta^n \leq (2\beta)^{n^2+2n} n^{3n},$$

and $\log|R| \leq \varrho := (n^2+2n)\log(2\beta) + 3n\log n = O(n^2\log\beta + n\log n)$. By [14], the number of primes less than or equal to z is at least $z/\log(z)$ for $z \geq 17$, from which it is easily derived that there are at least 10ϱ primes in $\mathcal{L} = \{2, \ldots, 20\varrho\log\varrho + 300\}$. Compute $R \bmod p$ for a subset $\mathcal{L}'$ of $\mathcal{L}$ such that $\prod_{p\in\mathcal{L}'} p > |R|$. Clearly $\#\mathcal{L}' = O(\varrho)$. Then reconstruct R using the Chinese remainder theorem. This allows us to certify $R = 0$ with $O(\varrho n^2 \cdot \log^2\varrho)$ bit operations.

A faster Monte Carlo probabilistic approach is to choose a second random prime q, and hope that q does not divide R (assuming $R \neq 0$). We choose a random prime $q \in \mathcal{L}$ and compute $R_q = \mathrm{res}(f(x+h),g(x)) \bmod q$ and output whether or not $R_q \equiv 0 \bmod q$. If $R = 0$ then we always output the correct answer 0. If $R \neq 0$, we output "non-zero" with probability at least 9/10 on any invocation. This can, of course, be repeated to obtain greater probability of correct output.

Theorem 15. *Let $f, g \in \mathbb{Z}[x]$ with $\deg f, \deg g \leq n$ and $\|f\|, \|g\| \leq \beta$, and $h \in \mathbb{Z}$.*

- *We can certify whether $\gcd(f(x+h),g(x)) = 1$ with $O(n^4\log\beta\cdot(\log n + \log\log\beta)^2)$ bit operations.*
- *If $\gcd(f(x+h),g(x)) \neq 1$ the randomized method described above always outputs this fact correctly. If $\gcd(f(x+h),g(x)) = 1$ it outputs correctly with probability at least 9/10. The cost of this randomized method is $O(n^2(\log n + \log\log\beta)^2)$ bit operations using standard arithmetic.*

By way of comparison with the algorithm of [13], we note that algorithm is dominated, at least in theory, by the cost of factoring f and g over $\mathbb{Z}[x]$. This alone takes approximately $O(n^{10} + n^8\log^2(\|f\| + \|g\|))$ for a rigorously analyzed algorithm: see [6], Corollary 16.25. In practice, this can often be done much more quickly; for example the algorithm of [10] performs very well. However, all known factoring algorithms over $\mathbb{Z}[x]$ essentially lift a factorization modulo p and then attempt some form of factor combining to recover integral factors. The dominant cost in pDispersionSet is exactly this factorization modulo p and lifting (to about the same bound as for factoring in $\mathbb{Z}[x]$), and so in some sense is intrinsically faster than the full factorization in $\mathbb{Z}[x]$.

7. EXAMPLES

We have a prototype implementation of algorithms ShiftlessFactorization and RatSum in Maple. Here we show several examples of summation. When the input expression is

$$\frac{2x+3}{(x+101)^2+1} - \frac{1}{(x+1)^2+1} - \frac{2x+1}{(x+100)^2+1} + \frac{4}{(x^2+1)}$$

(the input is given in decomposed form here for readability only), our procedure RatSum returns the answer

$$\mathrm{LQ}\left((2x+3)E^{101} - (2x+1)E^{100} - E + 1, E - 1\right)\frac{1}{x^2+1} + \sum_x \frac{3}{x^2+1}$$

in 0.16 seconds[1]. Conversion to the expanded form

$$\frac{2x+1}{(x+100)^2+1} - \frac{1}{(x^2+1)} + \sum_x \frac{3}{x^2+1}$$

takes 0.01 seconds. We note that only the cost of this step depends on the dispersion of the denominator of summand. In this case the left quotient is a sparse operator and the expanded output is small. The standard Maple summation routine returns the answer in 434.63 seconds.

For the input

$$\frac{x^4 - 149x^2 - 14999x - 500050}{(x^3+300x^2+30001x+1000101)(x^3+x+1)},$$

the answer

$$\mathrm{LQ}\left((x+\frac{1}{2})E^{100} - x + \frac{199}{2}, E - 1\right)\frac{1}{x^3+x+1} + \sum_x \frac{x-100}{x^3+x+1}$$

is returned in 0.09 seconds. Observe, that after expansion the left quotient here will be a dense operator with 100 terms. Since computation of this left quotient involves only very simple operations of substitution and addition of polynomials, expansion takes only 0.06 seconds for this example; the answer is not shown to save space. Maple's standard summation does not return after 20 minutes on the same input.

Acknowledgements

The authors thank the referees for their valuable comments and suggestions.

8. REFERENCES

[1] S.A. Abramov. On the summation of rational functions. *U.S.S.R. Comput. Maths. Math. Phys.* **11**, pp. 324–330, 1971. Transl. from *Zh. vychisl. mat. mat. fiz.* **11**, pp. 1071–1075, 1971.

[2] S.A. Abramov. The rational component of the solution of a first-order linear recurrence relation with a rational right-hand side. *U.S.S.R. Comput. Maths. Math. Phys.* **15**, pp. 216–221, 1975. Transl. from *Zh. vychisl. mat. mat. fiz.* **15**, pp. 1035–1039, 1975.

[3] S.A. Abramov. Indefinite sums of rational functions. *Proceedings ISSAC'95*, pp. 303–308.

[1]All timings are taken on a 650 MHz Intel Pentium III processor.

[4] E. Bach, J. Driscoll and J. O. Shallit. Factor Refinement. *Journal of Algorithms*. **15**, pp. 199–222, 1993.

[5] E. Bach and J. Shallit. *Algorithmic Number Theory. Volume 1: Efficient Algorithms*. MIT Press, Boston MA, 1996.

[6] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, Cambridge, U.K., 1999.

[7] J. Gerhard. *Modular algorithms in symbolic summation and symbolic integration*. PhD thesis, Universität Paderborn, Germany, 2001.

[8] R.W. Gosper. Decision procedures for indefinite hypergeometric summation. *Proc. Natl. Acad. Sci. U.S.A.* **75**(1), pp. 40–42, 1978.

[9] M. van Hoeij. Rational solutions of linear difference equations. *Proceedings ISSAC'98*, pp. 120–123, 1998.

[10] M. van Hoeij. Factoring polynomials and the knapsack problem. *J. Number Theory.* **95**, pp. 167-189, 2002.

[11] P. Lisoněk, P. Paule and V. Strehl. Improvement of the Degree Setting in Gosper's Algorithm. *J. Symbolic Comput.* **16**, pp. 243–258, 1993.

[12] Y.K Man. On computing closed forms for indefinite summation. *J. Symbolic Comput.* **16**, pp. 355–376, 1993.

[13] Y.K. Man and F.J. Wright. Fast polynomial dispersion computation and its application to indefinite summation. *Proceedings ISSAC'94*, pp. 175–180, 1994.

[14] J.B. Rosser and L. Schoenfeld. Approximate formulas for some functions of prime numbers. *Ill. J. Math.* **6**, pp. 64–94, 1962.

[15] P. Paule. Greatest factorial factorization and symbolic summation. *J. Symbolic Comput.* **20**(3), pp. 235-268, 1995.

[16] R. Pirastu. *On combinatorial identities: symbolic summation and umbral calculus.* PhD thesis, Johannes Kepler Universität Linz, Austria, July 1996.