Algorithms in Rings of Differential Operators with Coefficients in Local Rings

中山, 洋将

(Degree)
博士（理学）
(Date of Degree)
2008-03-25
(Date of Publication)
2012-02-15
(Resource Type)
doctoral thesis
(Report Number)
甲4163
(URL)
https://hdl.handle.net/20.500.14094/D1004163

博士論文

Algorithms in Rings of Differential Operators with Coefficients in Local Rings (局所環を係数に持つ微分作用素のなす環におけるアルゴリズム)

平成 20 年 1 月

神戸大学大学院自然科学研究科

中山 洋将

Introduction

The computational study of D-modules started about 20 years ago except a few pioneering works. However, almost all works are on the Weyl algebra, which is the ring of differential operators with polynomial coefficients.

The objective of this thesis is a computational study of D-modules by utilizing the ring of differential operators with rational function coefficients of which denominators do not vanish at the origin. We denote the ring by $\mathcal{D}_{alg}$. Especially, we are mainly interested in algorithms of computing local b-functions, related objects and the Mora division algorithm in $\mathcal{D}_{alg}$.

Here is a brief overview of each chapters.

- Chapter 1 — Computing Local b-functions by an Exhaustive Search

  In computational algebraic analysis, b-functions is very important. Oaku gave general algorithms of computing global b-functions and local b-functions for polynomials ([21]). An efficient version of Oaku's algorithm computing global b-functions was given by Noro ([19]). These algorithm utilizes Gröbner basis computations in the ring of differential operators with polynomial coefficients D.

  In this chapter, we will give a new algorithm of computing local b-functions by utilizing Gröbner basis computation in the ring of differential operators with rational function coefficients of which denominators do not vanish at the origin. $\mathcal{D}_{alg}$. This algorithm is simple and can be applied not only for the computation of local b-functions for polynomials but also for that of generic local b-functions.

- Chapter 2 — Computing Local b-Functions by an Approximate Division Algorithm

  Castro gave a constructive division theorem in the ring of differential operators with formal power series coefficients $\widehat{\mathcal{D}}$ ([4]). This division procedure generally needs infinite reductions and is not algorithm. We will give an approximate division algorithm in $\widehat{\mathcal{D}}$, which gives an approximation of the remainder by Castro's division in $\widehat{\mathcal{D}}$. And we will propose a new algorithm computing local b-functions by utilizing this approximate division.

- Chapter 3 — Computing Annihilating Ideals

  In this chapter, we will give an algorithm of computing the annihilating ideal of f^s in $\mathcal{D}_{alg}[s]$, which is denoted by $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$. It can be computed by an analogous method in the case of the Weyl algebra D.

  In fact, $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$ are generated by generators of $\mathrm{Ann}_{D[s]} f^s$. But, in some cases, the computation of $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$ is faster than that of $\mathrm{Ann}_{D[s]} f^s$.

- Chapter 4 — Interactive User Interface for Computer Algebra

  In previous chapters, we use the Buchberger algorithm and a Mora division algorithm in $\mathcal{D}_{alg}$. But in some cases, the algorithms cause hard computation. Methods to improve the algorithms have not been studied in detail. We have designed and implemented a system to understand and improve the algorithms. We call this system "interactive user interface for computer algebra". The system visualizes computations of the algorithms. And we easily control the algorithms by utilizing the system. In this chapter, we will introduce the system, and give an interesting example found by utilizing the system.

- Chapter 5 — Some Mathematical Results and Questions

  We want to apply our computational techniques to mathematical problems and hope to find new mathematical facts with a help of our computational methods.

  Sekiguchi studied the 17 weighted homogeneous polynomials in 3 variables in the paper [27], and determined the factors of the b-functions of the polynomials. In this chapter, we will give the result of computing the b-functions of the polynomials by utilizing Oaku's algorithm.

  Sekiguchi posed a question on annihilating ideals $\mathrm{Ann}_D f^\alpha$. Here, we will illustrate some computational experiments on the question.

  We have not yet tried local version of these problems. It will be a next step.

Chapter 1

Computing Local b-Functions by an Exhaustive Search

1.1 Introduction

We use the following notations in this chapter.

$$
\begin{aligned}
&x = (x_1, \cdots, x_n), \partial = (\partial_1, \cdots, \partial_n), s = (s) \\
&D[s] = \left\{ \sum_{k \in \mathbb{N}, \beta \in \mathbb{N}^n} a_{k,\beta}(x) s^k \partial^\beta \mid a_{k,\beta}(x) \in \mathbb{C}[x] \right\} \\
&\mathbb{C}[x]_{\langle x \rangle} = \left\{ \frac{f(x)}{g(x)} \mid f(x), g(x) \in \mathbb{C}[x], g(0) \neq 0 \right\} \\
&\mathcal{D}_{alg}[s] = \left\{ \sum_{k \in \mathbb{N}, \beta \in \mathbb{N}^n} a_{k,\beta}(x) s^k \partial^\beta \mid a_{k,\beta}(x) \in \mathbb{C}[x]_{\langle x \rangle} \right\} \\
&\widehat{\mathcal{D}}[s] = \left\{ \sum_{k \in \mathbb{N}, \beta \in \mathbb{N}^n} a_{k,\beta}(x) s^k \partial^\beta \mid a_{k,\beta}(x) \in \mathbb{C}[[x]] \right\}
\end{aligned}
$$

We start with the definitions of the global b-function and the local b-function at the origin for a given polynomial $f \in \mathbb{C}[x]$.

The *global b-function* $\tilde{b}(s) \in \mathbb{C}[s]$ for a polynomial f is the monic and minimal degree polynomial which satisfies $Pf^{s+1} = \tilde{b}(s) f^s$ for a differential operator $P \in D[s]$.

The local b-function is defined analogously by replacing $D[s]$ by $\mathcal{D}_{alg}[s]$. In other words, The *local b-function* $b(s) \in \mathbb{C}[s]$ at the origin for a polynomial f is the monic and minimal degree polynomial which satisfies $Pf^{s+1} = \tilde{b}(s)f^s$ for a differential operator $P \in \mathcal{D}_{alg}[s]$. Let us present examples.

Example 1.1.1. (The global and local b-function of $f = x_1(x_1 + x_2 + 1)$.)
The global b-function is $b(s) = (s+1)^2$. In fact, the following identity holds.

$$(-\partial_2^2 + \partial_1\partial_2)f^{s+1} = (s+1)^2 f^s$$

The local b-function is $\tilde{b}(s) = s + 1$. In fact, the following identity holds.

$$\frac{1}{1+2x_1+x_2}\partial_1 f^{s+1} = (s+1)f^s$$

The global b-function is defined by Mikio Sato and J.Bernstein independently. The existence of the global b-function for any polynomial f is proved by J.Bernstein ([2], [3]). Hence, the b-function is sometimes called the Bernstein-Sato polynomial.

Let us note some properties of b-functions. Put

$$B(s) = \{b(s) \in \mathbb{C}[s] \mid b(s)f^s \in Df^{s+1}\}$$
$$\tilde{B}(s) = \{\tilde{b}(s) \in \mathbb{C}[s] \mid \tilde{b}(s)f^s \in \mathcal{D}_{alg}[s]f^{s+1}\}$$

Then, these two sets are principal ideals in $\mathbb{C}[s]$. The monic generator of $B(s)$ is the global b-function $b(s)$ and the monic generator of $\tilde{B}(s)$ is the local b-function $\tilde{b}(s)$. Since $D[s] \subset \mathcal{D}_{alg}[s]$, we have $B(s) \subset \tilde{B}(s)$ and hence $\langle b(s)\rangle \subset \langle \tilde{b}(s)\rangle$. Therefore, the global b-function $b(s)$ is divided by the local b-function $\tilde{b}(s)$. This fact will be used in our algorithms of computing local b-functions. If f has no singularity at the origin, the local b-function of f at the origin is $s+1$. M.Kashiwara proved that the roots of the global b-function are negative rational numbers [13].

General algorithms of computing global b-functions for polynomials have been studied since the work of T.Oaku ([24], [21], [22]). Oaku also implemented his algorithms in computer algebra systems for differential operators Kan/sm1 ([30], `bfunction.sm1`). An efficient version of Oaku's algorithm was given by M.Noro [19]. Noro implemented his algorithm in library `bfct` for the computer algebra system Risa/Asir ([29]).

T.Oaku gave an algorithm of computing local b-functions in [24], [21], [22], [25]. This algorithm utilizes Gröbner basis computations in the ring of polynomials and in the ring of differential operators with polynomial coefficients.

In 2005, Granger, Oaku, Takayama gave a Mora division algorithm for $\mathcal{D}_{alg}$ [9], [10]. We will give a new algorithm of computing local b-functions by utilizing the Mora division algorithm for $\mathcal{D}_{alg}$. We call the algorithm the exhaustive search algorithm. It is not efficient when the degree of the global b-function increases, but it is simple and can be applied not only for the computation of local b-functions but also for the computation of generic local b-functions. We also implements these algorithms and apply them to some problems. In this chapter, we illustrate these algorithms and explain details of our implementation and examine some test computations.

1.2 A survey of general methods for computing b-functions

In this section, we summarize the general methods for computing global and local b-functions by T.Oaku.

1.2.1 Algorithm of computing global b-function

The following algorithm, which computes the global b-function, is given by T.Oaku in [24], [21], [22]. An efficient version is given by M.Noro [19]

Algorithm 1.2.1. (Computing the global b-function for a polynomial f)

1. Compute a set of generators G of the annihilating ideal $\mathrm{Ann}_{D[s]}f^s$ of f^s in $D[s]$. (We mean by the annihilating ideal the set of the differential operators $P \in D[s]$ such that $P \cdot f^s = 0$.)
2. Let J be the left ideal in $D[s]$ generated by $G \cup \{f\}$. Compute the monic generator of the principal ideal $J \cap \mathbb{C}[s]$. (The generator is the global b-function of f.)

(1.) A method to find generators of $\mathrm{Ann}_{D[s]}f^s$ is given in [24], [21], [22], which utilizes Gröbner bases in D_{n+1}. Two methods, which are given by Oaku and Noro respectively, are known to get the generator of $J \cap \mathbb{C}[s]$.

2a. Compute Gröbner basis with the elimination order $<$ to eliminate variables except s in J ($x_i, \partial_i > s$).

2b. Compute a Gröbner basis G of J with respect to an order $<$. Compute the normal form $\mathrm{NF}_<(s^i, G, <)$ of s^i with respect to G. Determine numbers a_i such that $a_l\mathrm{NF}_<(s^l, G, <) + \cdots + a_0\mathrm{NF}_<(1, G, <) = 0$ and l is taken as the minimal number such that $a_l \neq 0$. The polynomial $a_l s^l + \cdots + a_0$ is the generator of $J \cap \mathbb{C}[s]$ ([19]).

Thus, we can obtain the global b-function by utilizing Gröbner basis computations in $D[s]$. Let us see an example to illustrate the algorithm.

Example 1.2.2. (Computing a global b-function)
We will compute the global b-function of $f = x^2 + y^2$. Generators of the annihilating ideal $\mathrm{Ann}_{D[s]}f^s$ are as follows.

$$\{x\partial_x + y\partial_y - 2s, y\partial_x - x\partial_y\}$$

Let I be the left ideal in $D[s]$ generated by these generators and f. We compute the Gröbner basis of I with respect to the elimination order $x, y, \partial_x, \partial_y \succ s$ and the result is

$$\{-s^2 - 2s - 1, (-s-1)y, (-s-1)x, y\partial_x - x\partial_y, x\partial_x + y\partial_y - 2s, x^2 + y^2, -y\partial_x^2 - y\partial_y^2 + 2s\partial_y\}$$

Therefore, the generator of $I \cap \mathbb{C}[s]$ is $-s^2-2s-1$ and hence the global b-function is $(s+1)^2$.

We note that it is known that the global b-function of $f = \sum_{i=1}^{n} x_i^2$ is $(s+1)(s+\frac{n}{2})$ and the differential operator in the left hand side of the definition of the b-function is $P = \sum_{i=1}^{n} \partial_i^2$

1.2.2 Algorithm for computing local b-function

In this subsection, we illustrate the algorithm of computing local b-function by T.Oaku [21], [22]. Gröbner basis computations in the ring of polynomials $K[x,s]$ and in the ring of differential operators $D[s], D_{n+1}[y]$ are used. We also implemented this algorithm to compare with our new methods.

Algorithm 1.2.3. (Computing the local b-function [21], [22])
Let y be a parameter.

1. Put $I = D_{n+1}[y] \cdot \left\{ t - yf, \partial_i + y\frac{\partial f}{\partial x_i}\partial_t (i = 1, \cdots, n) \right\}$. Compute a Gröbner basis G for I with the elimination order with respect to y such that w-homogeneous. Here, the weight vector w is defined as follows.

x_1	$\cdots$	x_n	t	y	ξ_1	$\cdots$	ξ_n	∂_t
0	$\cdots$	0	-1	1	0	$\cdots$	0	1

2. Put $\psi(G) = \{\psi(P(1)) \mid P(y) \in G\}$. Here, $P \in D_{n+1}$ and $\mathrm{ord}_w(P) = m$. ($\mathrm{ord}_w(P)$ is the order of P with respect to w; in other words, it is the maximal order with respect to w among the terms in P.) The polynomial $\psi(P)$ is defined by

$$\psi(P)(s) = \psi(P)(t\partial_t) = \begin{cases} \mathrm{in}_w(t^m P) & (m \geq 0) \\ \mathrm{in}_w(\partial_t^{-m} P) & (m < 0) \end{cases}$$

(Here, $\mathrm{in}_w(P)$ is the sum of maximal w-order terms in P.)

3. Compute a Gröbner basis G_1 of $D[s] \cdot \psi(G)$ with the elimination order with respect to ∂_i. Put $J = K[x,s] \cdot (G_1 \cap K[x,s])$.

4. The local b function $b(-s-1)$ is the monic generator of $K[[x]][s]J \cap K[s]$. The global b-function $b(-s-1)$ is the monic generator of $J \cap K[s]$. Hence, the remaining thing to do is to find the generator of $K[[x]][s]J \cap K[s]$.

5. Computing the generator of $K[[x]][s]J \cap K[s]$.
Let generators of J be $\{f_1(x,s), \cdots, f_k(x,s)\}$.
   (a) Let $f_0(s)$ be the generator of $J(0) = K[s] \cdot \{f_1(0,s), \cdots, f_k(0,s)\}$.
   (b) Factorize $f_0(s)$ in $\mathbb{Q}[s]$; we obtain $f_0(s) = g_1(s)^{l_1} \cdots g_d(s)^{l_d}$.
   (c) For each $i = 1, \cdots, d$, compute the ideal quotient $J : g_i(s)^l$ $(l = l_i, l_i + 1, \cdots)$ until the quotient does not contain a multiple of $g_i(s)$. Let l_i' be the minimal l such that the condition above is satisfied.
   (d) Put $b(s) = g_1(s)^{l_1'} \cdots g_d(s)^{l_d'}$. Then, $b(s)$ is the generator.

1.3 Division algorithm and Gröbner basis in $\mathcal{D}_{alg}[y]$

In this section, we will introduce the Mora division algorithm in $D[y]$ (Algorithm 1.3.5). The results of this section will be used in our algorithm of computing local b-functions (Algorithm 1.4.3).

1.3.1 The Mora division algorithm in $D[y]$

In the previous sections, we utilized Gröber bases in $D[s]$. We will use the parameter s for a homogenization parameter in this section. The role of the variable s in the previous sections will be played by the variable y in the sequel.

The algorithm given in this section is a variation of the Mora division algorithm given by M.Granger, T.Oaku, N.Takayama [9],[10]. They mainly discuss the Mora division algorithm in homogenized differential operators. We will discuss the Mora division algorithm in $D[y]$, which is the differential operator with a parameter y. No new idea is necessary and our discussion is analogous to that in [10].

By virtue of our Mora division algorithm in $D[y]$, we can obtain Gröbner bases in $\mathcal{D}_{alg}[y]$ and also the ideal membership test in $\mathcal{D}_{alg}[y]$.

We want to perform a division in $D[y]$ with respect to the following monomial order $<_1$.

x_1	$\cdots$	x_n	y	ξ_1	$\cdots$	ξ_n
0	$\cdots$	0	1	1	$\cdots$	1
-1	$\cdots$	-1	0	0	$\cdots$	0

(any term order $<'$)

Since the order $<_1$ is not a term order, the division algorithm for term orders will not terminate in general. The Mora division algorithm is a division algorithm which terminates for non-term orders. This algorithm utilizes a new variable s to make $(-1,1)$-homogenization. We perform the division with the homogenization and the term order $<_1^s$ in $D[y][s]$ associated to $<_1$. The result is obtained by dehomogenization (s is replaced by 1).

Let us define $(-1,1)$-homogenization. The weights for the variables x_i are -1 and the weights for y and the variables ξ_i are 1. The weight for the variable s is -1. A power of s is added to each term to make the polynomial homogeneous with this weight vector.

Definition 1.3.1. ($(-1,1)$-homogenization)

Let $P \in D[y]$ be expressed as $P = \sum a_{\alpha\beta\gamma} x^\alpha \partial^\beta y^\gamma \quad (\alpha \in (\mathbb{Z}_{\geq 0})^n, \beta \in (\mathbb{Z}_{\geq 0})^n, \gamma \in \mathbb{Z}_{\geq 0})$. We put

$m = \min \{|\beta| - |\alpha| + |\gamma| \mid a_{\alpha\beta\gamma} \neq 0\}$ (Here, $|v|$ is the sum of each element of v)

The $(-1,1)$-homogenization $P^{(s)}$ of P is defined by

$$P^{(s)} = \sum a_{\alpha\beta\gamma} x^\alpha \partial^\beta y^\gamma s^{-|\alpha|+|\beta|+|\gamma|-m}$$

Suppose that $P \in D[y][s]$ is expressed as $P = \sum a_{\alpha\beta\gamma\nu} x^\alpha \partial^\beta y^\gamma s^\nu$. When $d = -|\alpha| + |\beta| + |\gamma| - |\nu|$ holds for all $(\alpha, \beta, \gamma, \nu)$ such that $a_{\alpha\beta\gamma\nu} \neq 0$, P is called the degree d $(-1,1)$-homogeneous operator (polynomial).

We define a term order $<_1^s$ in $D[y][s]$ associated to $<_1$ as follows.

Definition 1.3.2. (A term order associated to $<_1$)
We define a term order $<_1^s$ in $D[y][s]$ by the following weight vectors and a tie-breaker.

x_1	$\cdots$	x_n	y	s	ξ_1	$\cdots$	ξ_n
1	$\cdots$	1	0	1	0	$\cdots$	0
0	$\cdots$	0	1	0	1	$\cdots$	1
-1	$\cdots$	-1	0	0	0	$\cdots$	0
(any term order $<'$)							

Since the first weight vector and the second weight vector are positive and cover all variables, this order is a term order.

For a given $(-1,1)$-homogeneous element, the leading monomial with respect to $<_1$ and the leading monomial with respect to $<_1^s$ are related as follows.

Lemma 1.3.3. (LM w.r.t $<_1$ and w.r.t $<_1^s$)
Suppose that $P \in D[y][s]$ is $(-1,1)$-homogeneous. Then, we have

$$\mathrm{LM}_{<_1}(P|_{s=1}) = (\mathrm{LM}_{<_1^s}(P))|_{s=1}$$

Let $P, Q \in D[y][s]$ be $(-1,1)$-homogeneous elements of the same degree. Then, we have

$$\mathrm{LM}_{<_1^s}(P) <_1^s \mathrm{LM}_{<_1^s}(Q) \Leftrightarrow \mathrm{LM}_{<_1}(P|_{s=1}) <_1 \mathrm{LM}_{<_1}(Q|_{s=1})$$

We suppose that $P, Q \in D[y][s]$ are $(-1,1)$-homogeneous and P is reducible by Q (i.e., $\mathrm{LM}_{<_1^s}(P)$ is divisible by $\mathrm{LM}_{<_1^s}(Q)$). Then, the remainder R is also $(-1,1)$-homogeneous and its $(-1,1)$-degree is same with that of P. Moreover, we have $\mathrm{LM}_{<_1^s}(R) <_1^s \mathrm{LM}_{<_1^s}(P)$. It follows from the Lemma above that we have $\mathrm{LM}_{<_1}(R|_{s=1}) <_1 \mathrm{LM}_{<_1}(P|_{s=1})$. These facts are key ingredients of the Mora division algorithm in $D[y]$.

Theorem 1.3.4. (The Mora division algorithm in $D[y]$, [9] [10])
$P, P_1, \cdots, P_m \in D[y]$ are inputs. The Mora division algorithm outputs $a(x) \in \mathbb{C}[x], Q_1, \cdots, Q_m \in D[y], R \in D[y]$, which satisfy the following conditions.

$$a(x)P = Q_1 P_1 + \cdots + Q_m P_m + R$$
$$a(0) \neq 0 \ (a(x) \text{ is unit})$$
$$Q_i \neq 0 \Rightarrow \mathrm{LM}_{<_1}(Q_i P_i) \leq_1 \mathrm{LM}_{<_1}(P)$$
$$R \neq 0 \Rightarrow \mathrm{LM}_{<_1}(R) \text{ is not divisible by any of } \mathrm{LM}_{<_1}(P_i)$$

The algorithm of computing these $a(x), Q_1, \cdots, Q_m, R$ are as follows.

Algorithm 1.3.5. (The Mora division algorithm, [9],[10])

```
$\boxed{\text{MoraDivision}(P, \{P_1, \cdots, P_m\}, <_1)}$
$\boxed{\text{input}}$ $P, P_1, \cdots, P_m \in D[y]$
$\boxed{\text{output}}$ $a(x) \in \mathbb{C}[x], Q_1, \cdots, Q_m, R \in D[y]$, which satisfy the conditions of Theorem 1.3.4.
$\mathcal{G} \leftarrow [P_1^{(s)}, \cdots, P_m^{(s)}]$
$R \leftarrow P^{(s)}$
$A \leftarrow 1$
$Q \leftarrow [0, \cdots, 0]$   ($Q$ is an array with $m$ elements)
if ($R \neq 0$)
    $\mathcal{F} \leftarrow \{P' \in \mathcal{G} \mid \exists l \in \mathbb{Z}_{\geq 0}$ s.t. $\text{LM}_{<_1^s}(P') | \text{LM}_{<_1^s}(s^l R)\}$
else
    $\mathcal{F} \leftarrow \phi$
$\mathcal{H} \leftarrow []$
while ($\mathcal{F} \neq \phi$){
    $P' \leftarrow$ ($l$ of $P'$ is the minimum among $\mathcal{F}$ (suppose that $P'$ is the $i$-th element of $\mathcal{G}$))
    if ($l > 0$){
        $\mathcal{G} \leftarrow$ append ($\mathcal{G}, [R]$)
        $\mathcal{H} \leftarrow$ append ($\mathcal{H}, [[A, Q]]$)
    }
    $R \leftarrow$ (the remainder of $s^l R$ by the division by $P'$)
    $U \leftarrow$ (the monomial multiplied to $P'$ during the reduction above)
    if ($i \leq m$) {
        $Q[i] \leftarrow Q[i] + U$
    } else if ($i > m$) {
        $[A', Q'] \leftarrow \mathcal{H}[i - m]$
        $A \leftarrow A - UA'$
        for ($j \leftarrow 1; j \leq m; j \leftarrow j + 1$)
            $Q[j] \leftarrow Q[j] - UQ'[j]$
    }
    if ($s | R$) {
        $\nu \leftarrow$ ( the maximal $k$ such that $s^k | R$)
        $R \leftarrow R / s^\nu$
    }
    $\mathcal{F} \leftarrow \{P' \in \mathcal{G} \mid \exists l \in \mathbb{Z}_{\geq 0}$ s.t. $\text{LM}_{<_1^s}(P') | \text{LM}_{<_1^s}(s^l R)\}$
}
for ($j \leftarrow 1; j \leq m; j \leftarrow j + 1$)
    $Q[j] \leftarrow Q[j]|_{s=1}$
$R \leftarrow R|_{s=1}$
$a \leftarrow A|_{s=1}$
return $[a, Q, R]$
```

The set $\mathcal{G}$ is called *reducer set*. When this algorithm starts, $\mathcal{G} = \left\{P_1^{(s)}, \cdots, P_m^{(s)}\right\}$ holds. When a reduction is done after multiplying s^l, the remainder is added to $\mathcal{G}$. This part is different from the standard (non-Mora) division algorithm.

Let us see 3 examples of the Mora division.

Example 1.3.6. (The Mora division algorithm: the simplest example)

We will divide x by the unit $1+x$. Since $1+x$ is a unit in $\mathcal{D}_{alg}[y]$, we have

$$x = \frac{1}{1+x} \cdot (1+x) + 0 \quad ((1+x) \cdot x = (1+x) + 0)$$

In order to illustrate steps of the Mora division algorithm, we derive the expression above by the algorithm.

Make $(-1,1)$ homogenization of $1+x$. The result is $s+x$. Steps of the Mora division algorithm are as follows.

$$\underline{x} \xrightarrow[s+x]{s\cdot, x} -x^2 \xrightarrow[\underline{x}]{-x} 0$$

Polynomials under arrows are reducers. If s is written on an arrow, it means that s is multiplied. Monomials above arrows are multipliers for reductions. Polynomials with underlines are those added to the reducer set $\mathcal{G}$. This reduction process yields the following identities.

$$\begin{aligned} s \cdot x &= x \cdot (s+x) - x^2 \\ -x^2 &= -x \cdot x + 0 \end{aligned}$$

From these identities, we obtain

$$(s+x) \cdot x = x \cdot (s+x) + 0$$

of which dehomogenization is

$$(1+x) \cdot x = x \cdot (1+x) + 0$$

Note that this procedure is nothing but the Mora division algorithm in $\mathbb{C}[x]$ ([11], [16]). The Mora division algorithm in $D[y]$ is a generalization of the Mora division algorithm in $\mathbb{C}[x]$

Example 1.3.7. (The Mora division algorithm for differential operators)

We will divide ∂ by $1+x$. Since $1+x$ is a unit in $\mathcal{D}_{alg}[y]$, the remainder should be 0.

Let us apply the Mora division algorithm. The $(-1,1)$-homogenization of $1+x$ is $s+x$.

$$\underline{\partial} \xrightarrow[s+x]{s\cdot, \partial} -x\partial - 1 \xrightarrow[\underline{\partial}]{-x} \underline{-1} \xrightarrow[s+x]{s\cdot, -1} x \xrightarrow[\underline{-1}]{-x} 0$$

The reduction procedure yields

$$\begin{aligned} s \cdot \partial &= \partial \cdot (s+x) - x\partial - 1 \\ -x\partial - 1 &= -x \cdot \partial - 1 \\ s \cdot (-1) &= (-1) \cdot (s+x) + x \\ x &= (-x) \cdot (-1) + 0 \end{aligned}$$

Thus, we obtain

$$(s+x)^2 \cdot \partial = (s\partial + x\partial - 1) \cdot (s+x) + 0$$

Dehomogenization of the expression above is

$$(1+x)^2 \cdot \partial \ = (\partial + x\partial - 1) \cdot (1+x) + 0$$

Although we have seen simple examples, the Mora division procedure may require high complexity to finish even for small inputs.

Example 1.3.8. (The Mora division algorithm, a hard example)

We start with the following easy example; we will divide $P = x_1x_2\partial_1\partial_2$ by $P_1 = x_1\partial_1 + x_1x_2\partial_2, P_2 = x_2\partial_2 + x_1x_2\partial_1$, which is taken from an example in [9], [10].

Homogenize the inputs, we obtain $P^{(s)} = x_1x_2\partial_1\partial_2, P_1^{(s)} = sx_1\partial_1 + x_1x_2\partial_2, P_2^{(s)} = sx_2\partial_2 + x_1x_2\partial_1$ We obtain the following division procedure.

$$P_3 = x_1x_2\partial_1\partial_2 \xrightarrow[P_1^{(s)}]{s\cdot, x_2\partial_2} P_4 = -x_1x_2^2\partial_2^2 - x_1x_2\partial_2 \xrightarrow[P_2^{(s)}]{s\cdot, -x_1x_2\partial_2} x_1^2x_2^2\partial_1\partial_2 + x_1^2x_2\partial_1 \xrightarrow[P^{(s)}]{x_1x_2}$$

$$P_5 = x_1^2x_2\partial_1 \xrightarrow[P_1^{(s)}]{s\cdot, -x_1x_2} P_6 = -x_1^2x_2^2\partial_2 \xrightarrow[P_2^{(s)}]{s\cdot, x_1^2x_2} x_1^3x_2^2\partial_1 \xrightarrow[P_5]{x_1x_2} 0$$

Thus, we have

$$(1 - x_1x_2)^2P = (x_2\partial_2 - x_1x_2^2\partial_2 + x_1x_2)P_1 + (-x_1x_2\partial_2 + x_1^2x_2^2\partial_2 - x_1^2x_2)P_2 + 0$$

Now, we will consider harder problems. We will multiply units to P and divide it by P_1, P_2. The next table is the time and the number of reductions when we divide $(1+x_1)^n \cdot P$ by P_1, P_2 with our Mora division algorithm. We can see that the complexity becomes huge when n increases.

n	time (seconds)	the number of reductions
50	0.90 + 0.22	2762
100	11.9 + 1.1	10512
200	173.2 + 14.5	41012
300	823.9 + 61.1	91512

1.3.2 The Mora division algorithm 2 in $D[y]$ (with using yet another monomial orders)

In the previous section, we give the Mora division algorithm for the monomial order $<_1$. We will give yet another monomial orders for which the Mora division algorithm works. When the Mora division for $<_1$ is hard, these orders might work well for some problems.

Let us consider the following monomials order $<_2$

x_1	$\cdots$	x_n	y	ξ_1	$\cdots$	ξ_n
0	$\cdots$	0	0	1	$\cdots$	1
0	$\cdots$	0	1	0	$\cdots$	0
-1	$\cdots$	-1	0	0	$\cdots$	0

(any term order $<'$)

We will use $<_2$ instead of $<_1$. The difference is weights for y. Since $\xi_1, \cdots \xi_n >_2 y$, this order is an elimination order with respect to $\xi_1, \cdots, \xi_n$.

The $(-1,1)$-homogenization of the Mora division algorithm for this order is different from the case of $<_1$. In case of $<_1$, the following weight vector v_1 is used for homogenization with respect to the variable s.

x_1	$\cdots$	x_n	y	s	ξ_1	$\cdots$	ξ_n
-1	$\cdots$	-1	1	-1	1	$\cdots$	1

In case of $<_2$, we use the following weight vector v_2 for homogenization with respect to s.

x_1	$\cdots$	x_n	y	s	ξ_1	$\cdots$	ξ_n
-1	$\cdots$	-1	0	-1	1	$\cdots$	1

The term order on $D[y][s]$ associated to $<_2$ is as follows.

x_1	$\cdots$	x_n	y	s	ξ_1	$\cdots$	ξ_n
1	$\cdots$	1	0	1	0	$\cdots$	0
0	$\cdots$	0	0	0	1	$\cdots$	1
0	$\cdots$	0	1	0	0	$\cdots$	0
-1	$\cdots$	-1	0	0	0	$\cdots$	0

(any term order $<'$)

The Lemma 1.3.3 and the Theorem 1.3.4 holds for the order $<_2$ and then Algorithm 1.3.5 is correct.

We consider the following order $<_3$

x_1	$\cdots$	x_n	y	ξ_1	$\cdots$	ξ_n
0	$\cdots$	0	1	0	$\cdots$	0
0	$\cdots$	0	0	1	$\cdots$	1
-1	$\cdots$	-1	0	0	$\cdots$	0

(any term order $<'$)

The weight vector v_3 for the $(-1,1)$-homogenization is

x_1	$\cdots$	x_n	y	s	ξ_1	$\cdots$	ξ_n
-1	$\cdots$	-1	0	-1	1	$\cdots$	1

The associated term order $<_3^s$ is

x_1	$\cdots$	x_n	y	s	ξ_1	$\cdots$	ξ_n
0	$\cdots$	0	1	0	0	$\cdots$	0
1	$\cdots$	1	0	1	0	$\cdots$	0
0	$\cdots$	0	0	0	1	$\cdots$	1
-1	$\cdots$	-1	0	0	0	$\cdots$	0

(any term order $<'$)

The Mora division algorithm works for this order, too.

1.3.3 An implementation of the Mora division algorithm in $D[y]$

We have implemented the Mora division algorithm. We will present some heuristic strategies installed in our implementation to improve the performance.

- Strategies to choose a reducer: In the Algorithm 1.3.5, we choose one reducer P' among $\mathcal{G}$. The choice sometimes changes the performance drastically.

  Example 1.3.9. (Comparison of strategies to choose a reducer)

  We will compare two strategies to choose a reducer. One is to choose the minimal sugar degree reducer (min sugar) and the another is to chose the maximal sugar degree reducer. As to the definition of the sugar degree, see [8]. When no cancellation happens, the sugar degree agrees with the total degree. It is known that choosing the minimal sugar polynomial to reduce among the S-polynomials makes the Buchberger algorithm efficient ([8]).

  We apply the Mora division algorithm for $P = (1 + x_1)^{50} x_1 x_2 \partial_1 \partial_2$ with the initial reducer set $P_1 = x_1\partial_1 + x_1 x_2 \partial_2, P_2 = x_2\partial_2 + x_1 x_2 \partial_1$ and with the order $<_1$.

  The following table is a comparison of the two strategies (min sugar and max sugar).

Choice of reducer	Time(seconds)	The number of reductions	$\sharp\mathcal{G}$
min sugar	0.20	1062	6
max sugar	99.3 + 9.9	2989	21

  The "min sugar" strategy gives a better performance than the "max sugar" strategy in this example.

- Using modular computation: When coefficients swell into huge big numbers during the computation, the computation in $\mathbb{Z}/p\mathbb{Z}$ makes the performance better. However, the remainder is not always correct, this method can be used only for guessing the answer.

- Use the monomial orders $<_2$ or $<_3$: When the computation with $<_1$ fails, it might succeed with the order $<_2, <_3$. There are opposite cases, too.

1.3.4 Computation of Gröbner bases in $\mathcal{D}_{alg}[y]$

Suppose that an ideal $\mathcal{I}$ in $\mathcal{D}_{alg}[y]$ or $\widehat{D}[y]$ is generated by elements in $D[y]$. There are two known methods to compute Gröbner basis of $\mathcal{I}$.

1. Use the Buchberger algorithm in $D[y]$ with the Mora division algorithm.
2. Make $(-1,1)$-homogenization of inputs. Compute Gröbner basis by the Buchberger algorithm with the term order $<_1^s$. Dehomogenize the result.

In most examples I have tried, the latter method is faster. The number of the elements in the basis is larger in the latter method in most cases.

Example 1.3.10. (The Buchberger algorithm with the Mora division algorithm vs the Buchberger algorithm with the homogenization method)

We will compute the Gröbner basis of

$$\mathcal{I} = \left\{P_1 = \partial, P_2 = 1 + x^3\right\}$$

Since $P_1 = 1 + x^3$ is unit in $\mathcal{D}_{alg}[y]$, we have $\mathcal{I} = \mathcal{D}_{alg}[y]$.

The Buchberger algorithm with the Mora division: The remainder of $\mathrm{sp}_{<_1}(P_1, P_2)$ by $\{P_1, P_2\}$ with the Mora division and with the order $<_1$ is 0. Then, $\{P_1, P_2\}$ is a Gröbner basis.

The Buchberger algorithm with the homogenization: The Gröbner basis of $\mathcal{I}^{(s)} = \left\{P_1^{(s)} = \partial, P_2^{(s)} = s^3 + x^3\right\}$ with respect to the term order $<_1^s$ is

$$\left\{P_1^{(s)} = \partial, P_2^{(s)} = s^3 + x^3, -3x^2, 6x, -6\right\}$$

Both outputs imply that the ideal is generated by 1.

1.4 An algorithm of computing the local b functions; an exhaustive search method

Let $\mathcal{I}$ in $\mathcal{D}_{alg}[s]$ be the ideal generated by $\mathrm{Ann}_{D[s]} f^s$ and f. The monic generator of $\mathcal{I} \cap \mathbb{C}[s]$ is the local b-function. In case of the ring of polynomials or the Weyl algebra, we can obtain the generator by computing Gröbner basis with the elimination order to eliminate all variables except s. However, we cannot use the elimination order for $\mathcal{D}_{alg}[s]$. Because the order to eliminate the variables except s must satisfy the condition $x_1, \cdots, x_n > s$.

In this section, we will give an algorithm of performing this elimination and obtain the generator. The second method will be given in the next section. Let us sketch the idea of our method. Let G be a Gröbner basis of $\mathcal{I}$ in $\mathcal{D}_{alg}[s]$. We note that we can solve the ideal membership test in $\mathcal{D}_{alg}[s]$ by the Mora division algorithm. We perform the ideal membership tests by G of the all products by combinations of linear factors of the global b-function. Since the local b-function $b(s)$ is a factor of the global b-function $\tilde{b}(s)$, the minimal degree product which belongs to $\mathcal{I}$ is the local b-function.

Algorithm 1.4.1. (Computing the local b-function (exhaustive search method))

$\boxed{\text{Input}}$: $f \in \mathbb{C}[x]$

$\boxed{\text{Output}}$: The local b-function of f

```
$H \leftarrow$ a set of generators of $\mathrm{Ann}_{D[s]} f^s$
$\mathcal{I} \leftarrow$ ( the ideal in $\mathcal{D}_{alg}[s]$ generated by $H \cup \{f\}$)
$G \leftarrow$ a Gröbner basis of $\mathcal{I}$
$BF \leftarrow$ the global $b$-function of $f$
$L \leftarrow$ the set of the divisors of $BF$
$LF \leftarrow$ the set of elements of $L$ of which remainders by the Mora division with $G$ are 0
return the minimal degree polynomial in $LF$
```

Example 1.4.2. (Computing the local b-function of $f = x_1^2(x_1+1)^3$)

The global b-function of $f = x_1^2(x_1+1)^3$ is $\tilde{b}(s) = \frac{1}{18}(s+1)(2s+1)(3s+1)(3s+2)$. The set $\{g_1 = 2s + 5sx_1 - x_1\partial_1 - x_1^2\partial_1^2\}$ is a set of generators of $\mathrm{Ann}_{D[s]} f^s$. Put $J = \mathcal{D}_{alg}[s] \cdot \{f, g_1\}$. The set $G = \{f, g_1\}$ is a Gröbner basis of J. The remainders by the Mora division algorithm by G of the following divisors of $\tilde{b}(s)$ are 0.

- $\frac{1}{18}(s+1)(2s+1)(3s+1)(3s+2)$
- $\frac{1}{6}(s+1)(2s+1)(3s+1)$
- $\frac{1}{6}(s+1)(2s+1)(3s+2)$
- $\frac{1}{2}(s+1)(2s+1)$

The minimal degree polynomial is $\frac{1}{2}(s+1)(2s+1)$ and then it is the local b-function of f. In particular, by examining the division procedure of $\frac{1}{2}(s+1)(2s+1)$ by G, we obtain the following identity, too.

$$\frac{1}{2}(s+1)(2s+1)f^s = \frac{1}{2}\cdot\frac{1}{(x_1+1)^2(5x_1+2)^3}\cdot(4\partial_1+14x_1\partial_1+10x_1^2\partial_1-12-15x_1)\cdot\partial_1\cdot f^{s+1}$$

The disadvantage of the exhaustive search method is that larger the degree of the global b-function is, huger the number of combinations of factors is. We may not check all the combinations and can reduce the number of ideal membership tests. Let us explain the ideal. We note that

$$g(s) \in \mathcal{I} \cap \mathbb{C}[s] \Leftrightarrow \exists(\text{a divisor of } g(s)) \in \mathcal{I} \cap \mathbb{C}[s].$$

Therefore, if $g(s) \notin \mathcal{I}$, we do not need to make the ideal membership test for the divisors of $g(s)$.

Let us explain this method more precisely. We can suppose that the global b-function is factored as

$$\tilde{b}(s) = b_1(s)^{e_1} b_2(s)^{e_2} \cdots b_l(s)^{e_l} \quad (b_i(s) = s + a_i \quad a_i \in \mathbb{Q}_{>0})$$

by Kashiwara's rationality theorem. Put

$$f_i = \tilde{b}(s)/b_i(s)$$

We suppose that ideal membership tests outputs the following result.

$$f_{i_1}(s), \cdots, f_{i_m}(s) \in \mathcal{I} \quad f_{j_1}, \cdots, f_{j_{l-m}} \notin \mathcal{I}$$

If there is no f_i, which belongs to $\mathcal{I}$, then we conclude that the generator of $\mathcal{I} \cap \mathbb{C}[s]$ is $\tilde{b}(s)$. We will consider the other case, in which $m > 0$ holds. Put $g(s) = \gcd(f_{i_1}(s), \cdots, f_{i_m}(s))$. We have $g(s) \in \mathcal{I}$. Replace $\tilde{b}(s)$ by $g(s)$ and repeat this procedure until we succeed. Let us summarize this method by a pseudo code.

Algorithm 1.4.3. (Computing the local b-function (improved exhaustive search method))

```
$\boxed{\text{localb-rr}(f)}$
$\boxed{\text{input}}$ $f \in \mathbb{C}[x]$
$\boxed{\text{Output}}$ : The local $b$-function of $f$
$H \leftarrow$ a set of generators of $\mathrm{Ann}_{D[s]} f^s$
$\mathcal{I} \leftarrow$ ( the ideal in $\mathcal{D}_{alg}[s]$ generated by $H \cup \{f\}$)
$G \leftarrow$ a Gröbner basis of $\mathcal{I}$
$BF \leftarrow$ the global $b$-function of $f$
$LF \leftarrow BF$
while (true) {
    $LF = b_1(s)^{e_1} \cdots b_l(s)^{e_l}$  $(b_i(s) = s + a_i, a_i \in \mathbb{Q}_{>0})$
    $f_i \leftarrow LF/b_i(s)$  $(1 \le i \le l$, here, we choose $i$ such that $b_i(s)|LF)$
    Check if $f_i$ belongs to $\mathcal{I}$ by the Mora division algorithm by $G$.
    if $(f_i \notin \mathcal{I}(1 \le \forall i \le l))$
        return $LF$
    If $f_{i_1}, \cdots, f_{i_m} \in \mathcal{I}$, then
    $LF \leftarrow \gcd(f_{i_1}, \cdots, f_{i_m})$
}
```

Example 1.4.4. (The local b-function of $f = (x-1)^3 + (y+1)^2$)

The global b-function of $f = (x-1)^3 + (y+1)^2$ is $\tilde{b}(s) = (s+1)(s+\frac{5}{6})(s+\frac{7}{5})$. Since the polynomial f is smooth at the origin, the local b-function is $s+1$. We will derive this result by applying the algorithm above.

A set of generators of $\mathrm{Ann}_{D[s]} f^s$ is

$$\{g_1 = -6s - 2\partial_x + 3\partial_y + 2x\partial_x + 3y\partial_y, g_2 = 2\partial_x - 3\partial_y + 2y\partial_x + 6x\partial_y - 3x^2\partial_y\}.$$

Put $\mathcal{I} = \mathcal{D}_{alg}[s] \cdot \{f, g_1, g_2\}$. The set $\{f, g_1, g_2\}$ is a Gröbner basis of $\mathcal{I}$.

We denote by $f \xrightarrow{G} g$ that the remainder of f by the Mora division algorithm by G is g.

By applying the division algorithm, we have

$$\tilde{b}(s) = (s+1)(s+\frac{5}{6})(s+\frac{7}{5}) \xrightarrow{G} 0$$

For each degree 2 factors of $\tilde{b}(s)$, we have

$$f_1 = (s+1)(s+\frac{5}{6}) \xrightarrow{G} 0$$
$$f_2 = (s+1)(s+\frac{7}{5}) \xrightarrow{G} 0$$
$$f_3 = (s+\frac{5}{6})(s+\frac{7}{5}) \xrightarrow{G} \text{non-zero}$$

We have $(s+1) = \gcd(f_1, f_2)$ and

$$s+1 \xrightarrow{G} 0$$

Therefore, we conclude that the local b-function is $s+1$.

1.5 Algorithm of computing local generic b-functions

1.5.1 Global generic b-function

Let K be a field.

Definition 1.5.1. (global generic b-function, [26])

Let I be a left ideal in D. The ***global generic b-function*** of I with respect to the weight vector $w \in \mathbb{Z}^n_{\geq 0} \setminus \{0\}$ is the monic generator $b(s)$ of $\mathrm{in}_{(-w,w)}(I) \cap K[s]$ where $s = w_1\theta_1 + \cdots + w_n\theta_n$. The generic b-function is also called the *indicial polynomial.*

The global b-function of a polynomial f is the generic b-function of the holonomic D_{n+1}-ideal $D_{n+1} \cdot \{t - f, \partial_1 + \frac{\partial f}{\partial x_1}\partial_t, \cdots, \partial_n + \frac{\partial f}{\partial x_n}\partial_t\}$ with respect to the weight vector $w = (0, \cdots, 0, 1)$ (the weight of t is 1).

It is known that the generic b-function for holonomic D-ideal is not zero ([26, Theorem 5.1.2,Corollary 5.1.4]). Roots of generic b-functions are used in the restriction and integration algorithms for holonomic D-modules. Algorithms of computing the generic b-function are given in [22], [25].

1.5.2 A definition of local generic b-function

We suppose that the weight vector $w \in \mathbb{Z}^n_{\geq 0} \setminus \{0\}$ satisfies the condition

$$w_1 = \cdots = w_p = 0, w_{p+1} > 0, \cdots w_n > 0$$

Then, we have

$$\mathrm{gr}_{(-w,w)}(\widehat{\mathcal{D}}) \cong K[[x_1, \cdots, x_p]][x_{p+1}, \cdots, x_n]\langle \partial_1, \cdots, \partial_n \rangle$$

For given $\widehat{\mathcal{D}}$-ideal $\mathcal{I}$, $\mathrm{in}_{(-w,w)}(\mathcal{I})$ is the ideal in $\mathrm{gr}_{(-w,w)}(\widehat{\mathcal{D}})$ generated by $\{\mathrm{in}_{(-w,w)}(P) \mid P \in \mathcal{I}\}$.

Definition 1.5.2. (local generic b-function) The local generic b-function $b(s)$ of $\widehat{\mathcal{D}}$-ideal $\mathcal{I}$ with respect to the weight vector w is the monic generator of $\mathrm{in}_{(-w,w)}(\mathcal{I}) \cap K[s]$ where $s = w_{p+1}\theta_{p+1} + \cdots + w_n\theta_n$.

The local b-function for f is nothing but the generic local b-function for the holonomic $\widehat{\mathcal{D}}_{n+1}$-ideal $\widehat{\mathcal{D}}_{n+1} \cdot \{t - f, \partial_1 + \frac{\partial f}{\partial x_1}\partial_t, \cdots, \partial_n + \frac{\partial f}{\partial x_n}\partial_t\}$ with respect to the weight $w = (0, \cdots, 0, 1)$ (the weight for t is 1). Here, K is supposed to be $\mathbb{C}$.

We suppose that both of D-ideal I and $\widehat{\mathcal{D}}$-ideal $\mathcal{I}$ are generated by $P_1, \cdots, P_m$.

Lemma 1.5.3. Retain the assumption above. Then, the local generic b-function with respect to the weight w is a divisor of the global generic b-function of I with respect to the weight w.

Let us prove this lemma. Since $\mathrm{in}_{(-w,w)}(I) \subset \mathrm{in}_{(-w,w)}(\mathcal{I})$, we have $\mathrm{in}_{(-w,w)}(I) \cap K[s] \subset \mathrm{in}_{(-w,w)}(\mathcal{I}) \cap K[s]$. Any ideal in $K[s]$ is principal, then the global generic b-function is divided by the local b-function.

It follows from the lemma that the exhaustive search method described in the previous section can also be applied to the computation of generic local b-functions.

Note that we need a factorization method in $K[x]$ for our algorithm. When $K = \mathbb{C}$ and the global generic b-function lies in $\mathbb{Q}[x]$, this can be done by factorization algorithms in algebraic extension fields.

1.5.3 Computation of the local generic b-function

Theorem 1.5.4. Suppose that generators P_i have coefficients in $\mathbb{Q}$. When $K = \mathbb{C}$, the local generic b-function is computable with the following algorithm.

Let us illustrate our algorithm. In the first step, we need to find generators of $\mathrm{in}_{(-w,w)}(\mathcal{I})$. We use the Mora division algorithm in $\mathrm{gr}_{(-w,w)}(\widehat{\mathcal{D}}) = K[[x_1, \cdots, x_p]][x_{p+1}, \cdots, x_n]\langle\partial_1, \cdots, \partial_n\rangle$. We can use the following monomial order $<$ for the Mora division algorithm in $K[[x_1, \cdots, x_p]][x_{p+1}, \cdots, x_n]\langle\partial_1, \cdots, \partial_n\rangle$

x_1	$\cdots$	x_p	x_{p+1}	$\cdots$	x_n	∂_1	$\cdots$	∂_p	∂_{p+1}	$\cdots$	∂_n
0	$\cdots$	0	1	$\cdots$	1	0	$\cdots$	0	1	$\cdots$	1
0	$\cdots$	0	0	$\cdots$	0	1	$\cdots$	1	0	$\cdots$	0
-1	$\cdots$	-1	0	$\cdots$	0	0	$\cdots$	0	0	$\cdots$	0
(					any term order						)

The following weight vector is used for the homogenization with respect to s.

x_1	$\cdots$	x_p	x_{p+1}	$\cdots$	x_n	s	∂_1	$\cdots$	∂_p	∂_{p+1}	$\cdots$	∂_n
-1	$\cdots$	-1	0	$\cdots$	0	-1	1	$\cdots$	1	0	$\cdots$	0

The following order $<^s$ is a term order and we apply the Mora division algorithm with this order.

x_1	$\cdots$	x_p	x_{p+1}	$\cdots$	x_n	s	∂_1	$\cdots$	∂_p	∂_{p+1}	$\cdots$	∂_n
0	$\cdots$	0	1	$\cdots$	1	0	0	$\cdots$	0	1	$\cdots$	1
1	$\cdots$	1	0	$\cdots$	0	1	0	$\cdots$	0	0	$\cdots$	0
0	$\cdots$	0	0	$\cdots$	0	0	1	$\cdots$	1	0	$\cdots$	0
-1	$\cdots$	-1	0	$\cdots$	0	0	0	$\cdots$	0	0	$\cdots$	0
(						any term order						)

By utilizing this order and the Mora division, we can perform Gröbner basis computation in $K[[x_1, \cdots, x_p]][x_{p+1}, \cdots, x_n]\langle\partial_1, \cdots, \partial_n\rangle$ and also ideal membership tests.

Let G be a Gröbner basis of I with respect to $<_{(-w,w)}$ ($<$ a term order). Then, $\mathrm{in}_{(-w,w)}(G)$ is a set of generators of $\mathrm{in}_{(-w,w)}(I)$. Analogous statement holds for $\widehat{\mathcal{D}}$-ideal $\mathcal{I}$

Proposition 1.5.5. Let G be a Gröbner basis of $\mathcal{I}$ with respect to $<_{(-w,w)}$ ($<$ is the monomial order defined above). Then, $\mathrm{in}_{(-w,w)}(G)$ is a set of generators of $\mathrm{in}_{(-w,w)}(\mathcal{I})$.

To finish the first step, we need to compute Gröbner basis of $\mathcal{I}$ with respect to $<_{(-w,w)}$. Let us explain our method. It can be performed by a homogenization as follows, which is an analogous homogenization method in case of D.

1. Homogenize generators of $\mathcal{I}$ with respect to $(-w, w)$. The new variable x_0 is used for the homogenization. The homogenized ideal is denoted by $\mathcal{I}^h$.

2. Let $<^h$ be the monomial oder in $K[[x_1, \cdots, x_p]][x_{p+1}, \cdots, x_n, x_0]\langle\partial_1, \cdots, \partial_n\rangle$ defined by the following weight vectors.

x_1	$\cdots$	x_p	x_{p+1}	$\cdots$	x_n	x_0	∂_1	$\cdots$	∂_p	∂_{p+1}	$\cdots$	∂_n
0	$\cdots$	0	0	$\cdots$	0	1	0	$\cdots$	0	0	$\cdots$	0
0	$\cdots$	0	1	$\cdots$	1	0	0	$\cdots$	0	1	$\cdots$	1
0	$\cdots$	0	0	$\cdots$	0	0	1	$\cdots$	1	0	$\cdots$	0
-1	$\cdots$	-1	0	$\cdots$	0	0	0	$\cdots$	0	0	$\cdots$	0
(						any term order						)

   This monomial order is an elimination order with respect to x_0.

   Compute Gröbner basis of $\mathcal{I}^h$ with respect to $<^h$. (We may use either the Buchberger algorithm with the Mora division or that with the Lazard homogenization method.) Let G^h be the Gröbner basis obtained.

3. The set $G = \mathrm{in}_{(-w,w)}(G^h|_{h=1})$ is a set of generators of $\mathrm{in}_{(-w,w)}(\mathcal{I})$.

The set G is nothing a set of generators of $\mathrm{in}_{(-w,w)}(\mathcal{I})$, but also a Gröbner basis of the initial ideal with respect to the monomial order $<$.

In the second step, by utilizing this Gröbner basis and the Mora division algorithm, we can determine the local generic b-function by trying to reduce all combinations of factors of the global generic function. The minimal degree polynomial of which remainder is 0 is the local generic b-function. This step is analogous with that of the exhaustive search method to determine the local b-function for a polynomial f.

Note that we need the factorization of the global b function into first order factors. In other words, we need the factorization in the polynomial ring with algebraic extension fields of $\mathbb{Q}$. We have not yet implemented this algorithm completely. Our factorization is done only in $\mathbb{Q}[s]$. Note that in case of the b-function of a polynomial we need factorizations only in $\mathbb{Q}[s]$ by virtue of Kashiwara's theorem.

Let us show one example.

Example 1.5.6. (Computation of the local generic b-function)

Put $f = x^3-(y+1)^2$. The annihilating ideal of f^{-1} is of the form $\mathrm{Ann}_D f^{-1} = D\cdot\{-3x^2\partial_y-2(y+1)\partial_x, 3(y+1)\partial_y+2x\partial_x-6\}$, which is known to be a holonomic ideal. We denote this ideal by I and by $\mathcal{I}$ the $\widehat{\mathcal{D}}$-ideal generated by the same set of generators. Let us compute the local generic b-function of $\mathcal{I}$ with respect to the weight vector $w = (1,0)$.

Apply our first step for this input. A set of generators of $\mathrm{in}_{(-w,w)}(\mathcal{I})$ is

$$\{3(y+1)\partial_y + 2x\partial_x + 6, (y+1)\partial_y + 2, \partial_x\}$$

which is also a Gröbner basis. We put it G. On the other hand, the global generic b-function of I with respect to w is $s(2s+3)$. We try to divide all combinations of the linear factors of the global generic b-function by the Mora division.

$$\begin{aligned} s(2s+3) &\xrightarrow{G} 0 \\ s &\xrightarrow{G} 0 \\ 2s+3 &\xrightarrow{G} \text{non-zero} \\ 1 &\xrightarrow{G} \text{non-zero} \end{aligned}$$

Thus, we conclude that the local generic b-function is s.

Chapter 2

Computing Local b-Functions by an Approximate Division Algorithm

2.1 Introduction

Let K be a field of characteristic 0. We denote the differential operator $\frac{\partial}{\partial x_i}$ by ∂_i. The ring of differential operators with polynomial coefficient

$$D = \left\{ \sum_{\beta \in \mathbb{N}^n} a_{k,\beta}(x)\partial^\beta \mid a_{k,\beta}(x) \in K[x] \right\}$$

is denoted by D, and that with formal power series coefficient

$$\widehat{\mathcal{D}} = \left\{ \sum_{\beta \in \mathbb{N}^n} a_{k,\beta}(x)\partial^\beta \mid a_{k,\beta}(x) \in K[[x]] \right\}$$

is denoted by $\widehat{\mathcal{D}}$. Here $x = (x_1, \cdots, x_n)$ and $\partial = (\partial_1, \cdots, \partial_n)$.

Division theorems are fundamental in several construction for D-modules. Castro gave a constructive division theorem in $\widehat{\mathcal{D}}$, which gives unique quotients and remainder [4]. However, the division procedure needs infinite reductions. On the other hand, Granger, Oaku and Takayama gave a division algorithm for algebraic data in $\widehat{\mathcal{D}}$ ([9], [10]), which is an analogous algorithm with the Mora division algorithm in the polynomial ring [11]. We call this division the Mora division algorithm in D. The division algorithm stops in finite steps, but the remainder is not completely reduced. By using the division algorithm, we can get a Gröbner basis and solve the ideal membership problem for algebraic data in $\widehat{\mathcal{D}}$.

The local b function of a polynomial $f \in K[x]$ is the minimum degree monic polynomial $b(s) \in K[s]$ which satisfies

$$\exists P \in \widehat{\mathcal{D}}[s] \text{ such that } Pf^{s+1} = b(s)f^s.$$

Oaku gave algorithms computing the local b function of a given polynomial by using a Gröbner basis method in D ([21], [22], [25]). In this paper, we propose a new algorithm to compute the local b function. We use the Mora division algorithm in D and an approximate division algorithm in $\widehat{\mathcal{D}}$, which gives an approximation of the remainder by Castro's division in $\widehat{\mathcal{D}}$.

2.2 Division theorem and approximate division algorithm in the ring of power series $K[[x]]$

There are several kinds of division theorem in $K[[x]]$. Among them, we need a division theorem which gives unique quotients and unique remainder for our approximate division algorithm in $\widehat{\mathcal{D}}$. We start with explaining this kind of division theorem. As to details about the division theorem including a history, we refer [6].

We define a monomial order $<_r$ on monomials in $K[[x]]$ by the following $1 \times n$ matrix and a term order $<$ as the tie-breaker.

$$\begin{array}{ccc} x_1 & \cdots & x_n \\ \hline -1 & \cdots & -1 \\ (& \text{term order} < &) \end{array}$$

In other words, we define the order $<_r$ as

$$x^\alpha <_r x^\beta \Leftrightarrow \begin{cases} -\alpha_1 - \cdots - \alpha_n < -\beta_1 - \cdots - \beta_n \text{ or} \\ (-\alpha_1 - \cdots - \alpha_n = -\beta_1 - \cdots - \beta_n \text{ and } x^\alpha < x^\beta) \end{cases}$$

Since the order is not a well order, ordinary division algorithm does not generally stop in finite steps.

Let $\alpha(1), \cdots, \alpha(s) \in (\mathbb{Z}_{\geq 0})^n$. For $(\alpha(1), \cdots, \alpha(s))$, we define a partition $\Delta^1, \cdots, \Delta^s, \overline{\Delta}$ of $(\mathbb{Z}_{\geq 0})^n$ as

$$\begin{aligned} \Delta^1 &= \alpha(1) + (\mathbb{Z}_{\geq 0})^n, \\ \Delta^2 &= (\alpha(2) + (\mathbb{Z}_{\geq 0})^n) \setminus \Delta^1, \cdots \\ \Delta^s &= (\alpha(s) + (\mathbb{Z}_{\geq 0})^n) \setminus (\Delta^1 \cup \cdots \cup \Delta^{s-1}), \\ \overline{\Delta} &= (\mathbb{Z}_{\geq 0})^n \setminus (\Delta^1 \cup \cdots \cup \Delta^s). \end{aligned}$$

Let $f \in K[[x]]$. We define the following notations.

$$\begin{aligned}
\mathrm{LM}_{<_r}(f) &: \text{leading monomial of } f \text{ with respect to } <_r \\
\mathrm{LT}_{<_r}(f) &: \text{leading term of } f \text{ with respect to } <_r \\
\mathrm{LE}_{<_r}(f) &: \text{leading exponent of } f \text{ with respect to } <_r \\
\mathrm{Exps}(f) &: \text{set of exponents appearing in } f \\
\mathrm{Rest}_{<_r}(f) &: f - \mathrm{LT}_{<_r}(f)
\end{aligned}$$

For example, let $f = 3x_1 + x_1x_2$, then $\mathrm{LM}_{<_r}(f) = x_1, \mathrm{LT}_{<_r}(f) = 3x_1, \mathrm{LE}_{<_r}(f) = (1,0), \mathrm{Exps}(f) = \{(1,0),(1,1)\}$, and $\mathrm{Rest}_{<_r}(f) = x_1x_2$.

Lemma 2.2.1. (Division by terms in $K[[x]]$)
Let $f \in K[[x]]$, $g_1, \cdots, g_s \in K[[x]]$ and $\Delta^1, \cdots, \Delta^s, \overline{\Delta}$ be the partition of $(\mathbb{Z}_{\geq 0})^n$ with respect to $(\mathrm{LE}_{<_r}(g_1), \cdots, \mathrm{LE}_{<_r}(g_s))$. Then, there exist quotients $q_1, \cdots, q_s \in K[[x]]$ and remainder $r \in K[[x]]$ which satisfies the following conditions :

$$\begin{aligned}
&f = q_1\mathrm{LT}_{<_r}(g_1) + \cdots + q_s\mathrm{LT}_{<_r}(g_s) + r, \\
&\mathrm{Exps}(g_i) + \mathrm{Exps}(q_i) \subset \Delta^i, \\
&\mathrm{Exps}(r) \subset \overline{\Delta}.
\end{aligned}$$

Namely, this is a division of f by leading terms $\mathrm{LT}_{<_r}(g_1), \cdots, \mathrm{LT}_{<_r}(g_s)$. Especially, $(q_1, \cdots, q_s, r)$ is uniquely determined by $(f, \mathrm{LT}_{<_r}(g_1), \cdots, \mathrm{LT}_{<_r}(g_s), <_r)$. We call this computation **term-div**$(f, g_1, \cdots, g_s)$.

Theorem 2.2.2. (Division theorem in $K[[x]]$)
Let $f \in K[[x]]$, $g_1, \cdots, g_s \in K[[x]]$, and $\Delta^1, \cdots, \Delta^s, \overline{\Delta}$ be the partition of $(\mathbb{Z}_{\geq 0})^n$ with respect to $(\mathrm{LE}_{<_r}(g_1), \cdots, \mathrm{LE}_{<_r}(g_s))$. Then, there exist quotients $q_1, \cdots, q_s \in K[[x]]$ and remainder $r \in K[[x]]$ which satisfies the following conditions.

$$\begin{aligned}
&f = q_1g_1 + \cdots + q_sg_s + r \\
&\mathrm{LE}_{<_r}(g_i) + \mathrm{Exps}(q_i) \subset \Delta^i \\
&\mathrm{Exps}(r) \subset \overline{\Delta}
\end{aligned}$$

Especially, $(q_1, \cdots, q_s, r)$ is uniquely determined by $(f, g_1, \cdots, g_s, <_r)$.

We show a procedure to obtain $q_1, \cdots, q_s, r$.

```
$K[[x]]$-division$(f, g_1, \cdots, g_s)$
Input : $f \in K[[x]], g_1, \cdots, g_s \in K[[x]]$
Output : $q_1, \cdots, q_s, r \in K[[x]]$ which satisfies
         $f = q_1 g_1 + \cdots q_s g_s + r$
         $\mathrm{LE}_{<_r}(g_i) + \mathrm{Exps}(q_i) \subset \Delta^i$
         $\mathrm{Exps}(r) \subset \overline{\Delta}$
1 :  $F \leftarrow f, q_i \leftarrow 0\ (1 \le i \le s), r \leftarrow 0$
2 :  while $(F \neq 0)$ {
3 :      $[q', r'] \leftarrow$ term-div$(F, g_1, \cdots, g_s)$
4 :      $q_i \leftarrow q_i + q'_i\ (1 \le i \le s)$
5 :      $r \leftarrow r + r'$
6 :      $F \leftarrow -\sum_{i=1}^{s} q'_i \mathrm{Rest}_{<_r}(g_i)$
7 :  }
8 :  return $[q, r]$
```

This procedure generally does not stop in finite steps. Therefore this procedure is not an algorithm in a strict sense. If input f and g_i are polynomials, we can give an algorithm which returns approximate quotients and approximate remainder which are correct up to given total degree N. Approximate algorithm of this kind has been discussed by several authors, for example, see [15]. After the manner of the approximate algorithm, we will give an algorithm which gives approximations of quotients and remainder by $K[[x]]$-division.

Algorithm 2.2.3. ($K[[x]]$-approximate-division)

```
$K[[x]]$-approximate-division$(f, g_1, \cdots, g_s, N)$
Input : $f \in K[x], g_1, \cdots, g_s \in K[x], N \in \mathbb{Z}_{>0}$
Output : $\overline{q_1}, \cdots, \overline{q_s}, \overline{r}, \overline{F} \in K[x]$ which satisfies the following conditions.
         $f = \overline{q_1} g_1 + \cdots + \overline{q_s} g_s + \overline{r} + \overline{F}$
         $\overline{F} \neq 0 \Rightarrow |\mathrm{LE}_{<_r}(\overline{F})| \ge N$
         Terms of $\overline{q_i}$ whose total degree is smaller than $N - |\mathrm{LE}_{<_r}(g_i)|$ agree
         with those of $q_i$.
         Terms of $\overline{r}$ whose total degree is smaller than $N$ agree with those of $r$.
Here $|\alpha|$ is $\sum_{i=1}^{n} \alpha_i$, where $\alpha = (\alpha_1, \cdots, \alpha_n)$,
and $q_i, r$ are quotients and remainder of $K[[x]]$-division$(f, g_1, \cdots, g_s)$.
1 :  $\overline{F} \leftarrow f, \overline{q_i} \leftarrow 0\ (1 \le i \le s), \overline{r} \leftarrow 0$
2 :  while $(\overline{F} \neq 0$ and $|\mathrm{LE}_{<_r}(\overline{F})| < N)$ {
3 :      $[q', r'] \leftarrow$ term-div$(\overline{F}, g_1, \cdots, g_s)$
4 :      $\overline{q_i} \leftarrow \overline{q_i} + q'_i\ (1 \le i \le s)$
5 :      $\overline{r} \leftarrow \overline{r} + r'$
6 :      $\overline{F} \leftarrow -\sum_{i=1}^{s} q'_i \mathrm{Rest}_{<_r}(g_i)$
7 :  }
8 :  return $[\overline{q}, \overline{r}, \overline{F}]$
```

Example 2.2.4. (Example of $K[[x]]$-approximate-division)

The example of the case of $f = x, g_1 = 1 + x, N = 5$ (execute $K[[x]]$-approximate-division$(x, 1 + x, 5)$)

term-div	q'	r'	$\overline{q}$	$\overline{r}$	$\overline{F}$
			0	0	x
$x = x \cdot 1 + 0$	x	0	x	0	$-x \cdot x$
$-x^2 = -x^2 \cdot 1 + 0$	$-x^2$	0	$x - x^2$	0	$-(-x^2) \cdot x$
$x^3 = x^3 \cdot 1 + 0$	x^3	0	$x - x^2 + x^3$	0	$-x^3 \cdot x$
$-x^4 = -x^4 \cdot 1 + 0$	$-x^4$	0	$x - x^2 + x^3 - x^4$	0	$-(-x^4) \cdot x$
$x^5 = x^5 \cdot 1 + 0$	x^5	0	$x - x^2 + x^3 - x^4 + x^5$	0	$-x^5 \cdot x$

The output of the approximate division algorithm is

$$x = (x - x^2 + x^3 - x^4 + x^5) \cdot (1 + x) - x^6.$$

In this case, the quotients and the remainder by $K[[x]]$-division are

$$\begin{aligned} x &= \frac{x}{1+x} \cdot (1 + x) + 0 \\ &= (x - x^2 + x^3 - x^4 + x^5 - \cdots) \cdot (1 + x) + 0 \end{aligned}$$

2.3 Division theorem and approximate division algorithm in $\widehat{\mathcal{D}}[y]$

Castro gave a constructive division procedure in $\widehat{\mathcal{D}}$ [4]. His division procedure returns unique quotients and remainder. In this section, we give an algorithm which gives approximations of the quotients and the remainder by his division. The approximate quotients and remainder are correct up to a given total degree.

Let y be a parameter, and ξ be a commutative variable which stands for ∂. We define a monomial order $<$ on $\widehat{\mathcal{D}}[y]$ by using the following $2 \times (2n + 1)$ matrix and the tie-breaker $<_1$.

x_1	$\cdots$	x_n	y	ξ_1	$\cdots$	ξ_n
0	$\cdots$	0	1	1	$\cdots$	1
-1	$\cdots$	-1	0	0	$\cdots$	0
(		term order	$<_1$			)

And we define an other monomial order $<_r$ on $\widehat{\mathcal{D}}[y]$ by using the following $1 \times (2n + 1)$ matrix and tie-breaker $<_1$.

x_1	$\cdots$	x_n	y	ξ_1	$\cdots$	ξ_n
-1	$\cdots$	-1	-1	-1	$\cdots$	-1
(		term order $<_1$				)

We define a weight vector e for monomials $\{x^\alpha \xi^\beta y^\gamma\}$ as

$$\begin{array}{ccccccccc} & x_1 & \cdots & x_n & y & \xi_1 & \cdots & \xi_n & \\ (& 0 & \cdots & 0 & 1 & 1 & \cdots & 1 &) \end{array}$$

If $|\beta| + |\gamma| = |\beta'| + |\gamma'|$ (in other words, monomials have the same order with respect to the weight vector e), then it holds that

$$x^\alpha \xi^\beta y^\gamma < x^{\alpha'} \xi^{\beta'} y^{\gamma'} \Leftrightarrow x^\alpha \xi^\beta y^\gamma <_r x^{\alpha'} \xi^{\beta'} y^{\gamma'}$$

Therefore, we get

$$\mathrm{LM}(P) = \mathrm{LM}_{<_r}(\mathrm{in}_e(P))$$

Here $\mathrm{in}_e(P)$ is the initial part of P with respect to the weight vector e. For example, when $P = x_1\partial_1^2 + x_1^2\partial_1^2 + x_1\partial_1 + 1$, we have $\mathrm{in}_e(P) = x_1\xi_1^2 + x_1^2\xi_1^2$. And $\mathrm{LM}_{<_r}(\mathrm{in}_e(P)) = x_1\xi_1^2, \mathrm{LM}(P) = x_1\xi_1^2$

We review Castro's division procedure in $\widehat{\mathcal{D}}[y]$. Of course, since the monomial order $<$ is not a well order, the procedure needs infinite reductions.

Theorem 2.3.1 ([4]). (Division theorem in $\widehat{\mathcal{D}}[y]$)

Let $P, P_1, \cdots, P_s \in \widehat{\mathcal{D}}[y]$, and $\Delta^1, \cdots, \Delta^s, \overline{\Delta}$ be the partition of $\mathbb{Z}_{\geq 0}^{2n+1}$ with respect to $(\mathrm{LE}_<(P_1), \cdots, \mathrm{LE}_<(P_s))$. There exist $Q_1, \cdots, Q_s, R \in \widehat{\mathcal{D}}[y]$ which satisfies the following conditions.

$$P = Q_1P_1 + \cdots + Q_sP_s + R$$
$$\mathrm{LE}_<(P_i) + \mathrm{Exps}(Q_i) \subset \Delta^i$$
$$\mathrm{Exps}(R) \subset \overline{\Delta}$$

Especially, $(Q_1, \cdots, Q_s, R)$ are uniquely determined by $(P, P_1, \cdots, P_s, <)$.

We show procedure for the division theorem.

```
D̂-division(P, P_1, ..., P_s)
Input : $P, P_1, \cdots, P_s \in \widehat{\mathcal{D}}[y]$
Output : $Q_1, \cdots, Q_s, R \in \widehat{\mathcal{D}}[y]$ which satisfies
         $P = Q_1P_1 + \cdots + Q_sP_s + R$
         $\mathrm{LE}_<(P_i) + \mathrm{Exps}(Q_i) \subset \Delta^i$
         $\mathrm{Exps}(R) \subset \overline{\Delta}$
1 :   $Q_i \leftarrow 0 \ (1 \leq i \leq s), R \leftarrow P$
2 :   $m_0 \leftarrow \mathrm{ord}_e(R)$
3 :   for $(k \leftarrow m_0; k \geq 0; k \leftarrow k-1)$ {
4 :       $r \leftarrow$ (total symbol of the part of $R$ whose $e$-order is $k$)
5 :       $[q_i', r'] \leftarrow K[[x]]$-division $(r, \mathrm{in}_e(P_1), \cdots, \mathrm{in}_e(P_s), <_r)$
6 :       $Q_i' \leftarrow$ (replace $\xi$ with $\partial$ in $q_i'$)  $(1 \leq i \leq s)$
7 :       $R \leftarrow R - \sum_{i=1}^{s} Q_i'P_i$
8 :       $Q_i \leftarrow Q_i + Q_i'$  $(1 \leq i \leq s)$
9 :   }
10 :  return $[Q_i, R]$
```

Here, $\mathrm{ord}_e(R)$ is e-order of R, defined by

$$\mathrm{ord}_e(R) = \max\{e \cdot \alpha \mid \alpha \in \mathrm{Exps}(R)\},$$

and the total symbol of $P(x,y,\partial) = \sum_\alpha a_\alpha(x,y)\partial^\alpha$ $(a_\alpha(x,y) \in K[[x]][y])$ is $P(x,y,\xi) = \sum_\alpha a_\alpha(x,y)\xi^\alpha$. For example, when $P = (x_1 + x_1x_2)\partial_1^2 + x_1\partial_1 + 1$, $\mathrm{ord}_e(P) = 2$ and the total symbol $P(x,\xi) = (x_1 + x_1x_2)\xi_1^2 + x_1\xi_1 + 1$.

Since the procedure uses $K[[x]]$-division at the 5th line, it does not stop in finite steps. We suppose that inputs are algebraic data, in other words, inputs are elements in D. By replacing the $K[[x]]$-division with $K[[x]]$-approximate-division (Algorithm 2.2.3), we can get approximations of the quotients and the remainder by $\widehat{\mathcal{D}}$-division.

Algorithm 2.3.2. ($\widehat{\mathcal{D}}$-approximate-division)

```
$\widehat{\mathcal{D}}$-approximate-division$(P, P_1, \cdots, P_s, N)$
Input : $P, P_1, \cdots, P_s \in D[y], N \in \mathbb{Z}_{>0}$
Output : $\overline{Q_1}, \cdots, \overline{Q_s}, \overline{R} \in D[y]$ satisfies
$P = \overline{Q_1}P_1 + \cdots + \overline{Q_s}P_s + \overline{R}$
Terms of $\overline{Q_i}$ whose total degree is smaller than $N - |\mathrm{LE}_<(P_i)|$ agree with those of $Q_i$
Terms of $\overline{R}$ whose total degree is smaller than $N$ agree with those of $R$
Here, $Q_i$ and $R$ are the result of $\widehat{\mathcal{D}}$-division$(P, P_1, \cdots, P_s)$.
1 :   $\overline{Q_i} \leftarrow 0$  $(1 \le i \le s), \overline{R} \leftarrow P$
2 :   $m_0 \leftarrow \mathrm{ord}_e(\overline{R})$
3 :   for $(k \leftarrow 0; k \le m_0; k \leftarrow k+1)$
4 :       $M_k \leftarrow \max(|\mathrm{LE}_<(P_i)| + 2(k - \mathrm{ord}_e(P_i)) \mid i$ such that $k \ge \mathrm{ord}_e(P_i))$
5 :   Bound $\leftarrow N + \sum_{i=0}^{m_0} M_i$
6 :   for $(k \leftarrow m_0; k \ge 0; k \leftarrow k-1)$ {
7 :       $\overline{r} \leftarrow$ (total symbol of the part of $\overline{R}$ whose $e$-order is k and whose
              total degree is smaller than Bound)
8 :       $[\overline{q_i'}, \overline{r'}] \leftarrow K[[x]]$-approximate-division$(\overline{r}, \mathrm{in}_e(P_1), \cdots, \mathrm{in}_e(P_s), <_r,$
              Bound)
9 :       $\overline{Q_i'} \leftarrow$ (replace $\xi$ with $\partial$ in $\overline{q_i'}$)  $(1 \le i \le s)$
10 :      $\overline{R} \leftarrow \overline{R} - \sum_{i=0}^{s} \overline{Q_i'}P_i$
11 :      $\overline{Q_i} \leftarrow \overline{Q_i} + \overline{Q_i'}$  $(1 \le i \le s)$
12 :      Bound $\leftarrow$ Bound $-M_k$
13 :  }
14 :  return $[\overline{Q_i}, \overline{R}]$
```

Since $K[[x]]$-approximate-division stops in finite steps, the algorithm stops in finite steps. In the remainder of this section, we will prove the correctness of the algorithm.

We put

$$D(m) = \left\{P \in D[y] \text{ (or } \widehat{\mathcal{D}}[y]) \mid \mathrm{ord}_e(P) = m\right\},$$

$$T(i) = \left\{P \in D[y] \text{ (or } \widehat{\mathcal{D}}[y], K[x,\xi], K[[x]][y,\xi]) \mid \text{(the total degree of every term of } P) \ge i\right\}.$$

It follows from the definition that $P - Q \in T(i)$ is equivalent to that P and Q

are the same up to total degree $i-1$. From the Leibniz rule, we get the following property of the multiplication of approximate elements.

Lemma 2.3.3. (Multiplication of approximate elements)

$$(T(i) \cap D(m)) \cdot T(j) \subset T(i+j-2m)$$

Here, dot $\cdot$ means multiplication in the ring of differential operators.

(Proof) We suppose that $P \in T(i) \cap D(m)$ and $Q \in T(j)$. Then we have

$$P = \sum_{|\beta| \le m, |\alpha|+|\beta| \ge i} a_{\alpha\beta} x^\alpha \partial^\beta \ , \ Q = \sum_{|\gamma|+|\delta| \ge j} b_{\gamma\delta} x^\gamma \partial^\delta \quad (a_{\alpha\beta}, b_{\gamma\delta} \in K)$$

To compute the multiplication PQ, we use the Leibniz rule. We can get the following total symbol for PQ.

$$\begin{aligned}(PQ)(x,\xi) &= \sum_\nu \frac{1}{\nu!} \frac{\partial^{|\nu|}}{\partial \xi^\nu} \Big(\sum_{(\alpha,\beta) \in \mathrm{Exps}(P)} a_{\alpha\beta} x^\alpha \xi^\beta \Big) \frac{\partial^{|\nu|}}{\partial x^\nu} \Big(\sum_{(\gamma,\delta) \in \mathrm{Exps}(Q)} b_{\gamma\delta} x^\gamma \xi^\delta \Big) \\ &= \sum_{(\alpha,\beta) \in \mathrm{Exps}(P), (\gamma,\beta) \in \mathrm{Exps}(Q), \nu \le \beta, \nu \le \gamma} c_{\alpha\beta\gamma\delta\nu} x^{\alpha+\gamma-\nu} \xi^{\beta+\delta-\nu} \quad (c_{\alpha\beta\gamma\delta\nu} \in K).\end{aligned}$$

Here, $\nu \le \beta$ means $\nu_i \le \beta_i$ for all i.

We consider the minimum total degree of terms of PQ.

$$\min(|\alpha| + |\gamma| - |\nu| + |\beta| + |\delta| - |\nu| \mid (\alpha, \beta) \in \mathrm{Exps}(P), (\gamma, \delta) \in \mathrm{Exps}(Q), \nu \le \beta, \nu \le \gamma) \tag{2.1}$$

Since $|\alpha| + |\beta| \ge i, |\gamma| + |\delta| \ge j$ and $|\nu| \le |\beta| \le \mathrm{ord}_e(P) = m$ hold, the minimum total degree (2.1) is more than or equal to $i + j - 2m$. Therefore, we conclude $PQ \in T(i + j - 2m)$. □

Lemma 2.3.4. Let q_i and r be the quotients and remainder of $K[[x]]$-division$(f, g_1, \cdots, g_s, <_r)$. If $f \in T(N)$, then $q_i \in T(N - |\mathrm{LE}_{<_r}(g_i)|), r \in T(N)$.

(Proof) It holds that

$$\begin{aligned} f &= q_1 g_1 + \cdots + q_s g_s + r \\ \mathrm{LE}_{<_r}(g_i) &+ \mathrm{Exps}(q_i) \subset \Delta^i \\ \mathrm{Exps}(r) &\subset \overline{\Delta} \end{aligned}$$

where $\Delta^1, \cdots, \Delta^s, \overline{\Delta}$ are the partition of $\mathbb{Z}_{\ge 0}^{2n+1}$ with respect to $(\mathrm{LE}_{<_r}(g_1), \cdots, \mathrm{LE}_{<_r}(g_s))$. From these properties, we get

$$\begin{aligned} \mathrm{LE}_{<_r}(q_i g_i) &\le_r \mathrm{LE}_{<_r}(f) \\ \mathrm{LE}_{<_r}(r) &\le_r \mathrm{LE}_{<_r}(f) \end{aligned}$$

From the definition of the monomial order $<_r$, we have

$$|\mathrm{LE}_{<_r}(q_i)| \geq N - |\mathrm{LE}_{<_r}(g_i)| \Leftrightarrow q_i \in T(N - |\mathrm{LE}_{<_r}(g_i)|)$$

and

$$|\mathrm{LE}_{<_r}(r)| \geq |\mathrm{LE}_{<_r}(f)| = N \Leftrightarrow r \in T(N).$$

□

Lemma 2.3.5. Let $f, g_1, \cdots, g_s \in K[[x]]$ and $\overline{f}$ be the part of f whose total degree is less than N, that is, $f - \overline{f} \in T(N)$. Let q_i and r be the quotients and remainder of $K[[x]]$-division$(f, g_1, \cdots, g_s, <_r)$. And let $\overline{q_i}$ and $\overline{r}$ be the quotients and remainder of $K[[x]]$-approximate-division$(\overline{f}, g_1, \cdots, g_s, <_r, N)$. Then $q_i - \overline{q_i} \in T(N - |\mathrm{LE}_{<_r}(g_i)|), r - \overline{r} \in T(N)$.

(Proof) Let q'_i and r' be the quotients and remainder of $K[[x]]$-division$(\overline{f}, g_1, \cdots, g_s, <_r)$. Since $K[[x]]$-division uniquely gives the quotient and remainder, the quotients and remainder of $K[[x]]$-division$(f - \overline{f}, g_1, \cdots, g_s, <_r)$ are $q_i - q'_i$ and $r - r'$. From $f - \overline{f} \in T(N)$ and Lemma 2.3.4, $q_i - q'_i \in T(N - |\mathrm{LE}_{<_r}(g_i)|)$ and $r - r' \in T(N)$ holds.

From the property of $K[[x]]$-approximate-division, $q'_i - \overline{q_i} \in T(N - |\mathrm{LE}_{<_r}(g_i)|)$ and $r' - \overline{r} \in T(N)$ hold. So we get $q_i - \overline{q_i} \in T(N - |\mathrm{LE}_{<_r}(g_i)|)$ and $r - \overline{r} \in T(N)$.
□

(Proof of Algorithm 2.3.2) We will compare steps in the for-loop in $\widehat{\mathcal{D}}$-division with those in $\widehat{\mathcal{D}}$-approximate division. We put

$$M_k = \max(|\mathrm{LE}_{<}(P_i)| + 2(k - \mathrm{ord}_e(P_i)) \mid i \text{ such that } k \geq \mathrm{ord}_e(P_i)).$$

We suppose that $R - \overline{R} \in T(N)$ in the $(k-1)$-th step. Let us compare k-th steps in the for-loop in $\widehat{\mathcal{D}}$-division and $\widehat{\mathcal{D}}$-approximate-division.

```
$\widehat{\mathcal{D}}$-division
$r \leftarrow$ (total symbol of the part of $R$ whose $e$-order is $k$)
$[q'_i, r'] \leftarrow K[[x]]$-division$(r, \mathrm{in}_e(P_1), \cdots, \mathrm{in}_e(P_s), <_r)$
$Q'_i \leftarrow q'_i(x, \partial)$
$R_{new} \leftarrow R - \sum Q'_i P_i$
```

```
$\widehat{\mathcal{D}}$-approximate-division
$\overline{r} \leftarrow$ (total symbol of the part of $\overline{R}$ whose $e$-order is $k$ and total degree
    is less than $N$)
$[\overline{q'_i}, \overline{r'}] \leftarrow K[[x]]$-approximate-division$(\overline{r}, \mathrm{in}_e(P_1), \cdots, \mathrm{in}_e(P_s), <_r, N)$
$\overline{Q'_i} \leftarrow \overline{q'_i}(x, \partial)$
$\overline{R_{new}} \leftarrow \overline{R} - \sum \overline{Q'_i} P_i$
```

We will prove that $R_{new} - \overline{R_{new}} \in T(N - M_k)$ holds after these computations have been done.

It follows from Lemma 2.3.5 that

$$q'_i - \overline{q'_i} \in T(N - |\mathrm{LE}_{<_r}(\mathrm{in}_e(P_i))|) = T(N - |\mathrm{LE}_<(P_i)|),$$
$$r' - \overline{r'} \in T(N)$$

Remainders r and $\overline{r}$ are e-homogeneous elements of order k, and $\mathrm{in}_e(P_i)$ is e-homogeneous element of order $\mathrm{ord}_e(P_i)$. So, if $q'_i, \overline{q'_i} \neq 0$, then $q'_i, \overline{q'_i}$ are e-homogeneous elements of order $k - \mathrm{ord}_e(P_i) (\geq 0)$. And, if $r', \overline{r'} \neq 0$, then $r', \overline{r'}$ are e-homogeneous elements of order k. Therefore, if $Q'_i, \overline{Q'_i} \neq 0$, then $Q'_i, \overline{Q'_i}$ have the same e-order $k - \mathrm{ord}_e(P_i)$ and are the same up to total degree $N - |\mathrm{LE}_<(P_i)| - 1$. In other words, it holds that

$$Q'_i - \overline{Q'_i} \in T(N - |\mathrm{LE}_<(P_i)|) \cap D(k - \mathrm{ord}_e(P_i)).$$

From Lemma 2.3.3, we have

$$(T(N-|\mathrm{LE}_<(P_i)|) \cap D(k-\mathrm{ord}_e(P_i))) \cdot T(0) \subset T(N-|\mathrm{LE}_<(P_i)|-2(k-\mathrm{ord}_e(P_i))).$$

Therefore it holds that

$$Q'_iP_i - \overline{Q'_i}P_i \in T(N - |\mathrm{LE}_<(P_i)| - 2(k - \mathrm{ord}_e(P_i)))$$

Since $M_k = \max(|\mathrm{LE}_<(P_i)| + 2(k - \mathrm{ord}_e(P_i)) \mid i \text{ such that } k \geq \mathrm{ord}_e(P_i))$, we get

$$\sum_{i=1}^{s} Q'_iP_i - \sum_{i=1}^{s} \overline{Q'_i}P_i \in T(N - M_k).$$

Therefore $R_{new} - \overline{R_{new}} \in T(N - M_k)$.

The accuracy of approximation decreases by M_k at each step of the for loop. To keep the accuracy, we beforehand add $\sum_{i=0}^{m_0} M_i$ to N. □

Example 2.3.6. (Example of $\widehat{\mathcal{D}}$-division and $\widehat{\mathcal{D}}$-approximate-division)

Let $n = 1$. We put $P = \partial^2, P_1 = (1+x)\partial + x$. We compare $\widehat{\mathcal{D}}$-division(P, P_1) and $\widehat{\mathcal{D}}$-approximate-division$(P, P_1, 5)$.

At first, we show the procedure of $\widehat{\mathcal{D}}$-division(P, P_1).

```
$R \leftarrow P, m_0 \leftarrow \mathrm{ord}_e(R) = 2$
$r \leftarrow$ (part of $R$ whose $e$-order is 2) $= \xi^2$
$[q_1', r'] \leftarrow K[[x]]$-division$(r, \mathrm{in}_e(P_1), <_r)$
$(q_1' = \frac{1}{1+x}\xi, r' = 0)$
$Q_1' \leftarrow \frac{1}{1+x}\partial$
$R \leftarrow R - Q_1'P_1 = -\partial - \frac{1}{1+x}$
$r \leftarrow$ (part of $R$ whose $e$-order is 1) $= -\xi$
$[q_1', r'] \leftarrow K[[x]]$-division$(r, \mathrm{in}_e(P_1), <_r)$
$(q_1' = -\frac{1}{1+x}, r' = 0)$
$Q_1' \leftarrow -\frac{1}{1+x}$
$R \leftarrow R - Q_1'P_1 = \frac{-1+x}{1+x}$
$r \leftarrow$ (part of $R$ whose $e$-order is 0) $= \frac{-1+x}{1+x}$
$[q_1', r'] \leftarrow K[[x]]$-division$(r, \mathrm{in}_e(P_1), <_r)$
$(q_1' = 0, r' = \frac{-1+x}{1+x})$
$Q_1' \leftarrow 0$
$R \leftarrow R - Q_1'P_1 = \frac{-1+x}{1+x}$
```

The output of $\widehat{\mathcal{D}}$-division(P, P_1) is

$$\partial^2 = (\frac{1}{1+x}\partial - \frac{1}{1+x})((1+x)\partial + x) + \frac{-1+x}{1+x}.$$

The quotient is

$$\frac{1}{1+x}\partial - \frac{1}{1+x} = (1 - x + x^2 - x^3 + x^4 - \cdots)(\partial - 1)$$

and the remainder is

$$\frac{-1+x}{1+x} = -1 + 2x - 2x^2 + 2x^3 - 2x^4 + 2x^5 - \cdots .$$

Next, we show the procedure of $\widehat{\mathcal{D}}$-approximate-division $(P, P_1, 5)$.

$\overline{R} \leftarrow P, m_0 \leftarrow \mathrm{ord}_e(R) = 2$
$M_0 \leftarrow 0, M_1 \leftarrow 1, M_2 \leftarrow 3$
Bound $\leftarrow 5 + (0 + 1 + 3) = 9$

$\overline{r} \leftarrow$ (part of $\overline{R}$ whose e-order is 2 and total degree is less than 9) $= \xi^2$
$[\overline{q'_1}, \overline{r'}] \leftarrow K[[x]]$-approximate-division($\overline{r}, \mathrm{in}_e(P_1), <_r, 9)$
$(\overline{q'_1} = (1 - x + x^2 - \cdots + x^6)\xi, \overline{r'} = -x^7\xi^2)$
$\overline{Q'_1} \leftarrow (1 - x + x^2 - \cdots + x^6)\partial$
$\overline{R} \leftarrow \overline{R} - \overline{Q'_1}P_1 = -x^7\partial^2 - \partial - x^7\partial - 1 + x - x^2 + x^3 - x^4 + x^5 - x^6$
Bound $\leftarrow 9 - 3 = 6$

$\overline{r} \leftarrow$ (part of $\overline{R}$ whose e-order is 1 and total degree is less than 6) $= -\xi$
$[\overline{q'_1}, \overline{r'}] \leftarrow K[[x]]$-approximate-division($\overline{r}, \mathrm{in}_e(P_1), <_r, 6)$
$(\overline{q'_1} = -1 + x - x^2 + x^3 - x^4, \overline{r'} = x^5\xi)$
$\overline{Q'_1} \leftarrow -1 + x - x^2 + x^3 - x^4$
$\overline{R} \leftarrow \overline{R} - \overline{Q'_1}P_1 = -x^7\partial^2 + x^5\partial - x^7\partial - 1 + 2x - 2x^2 + 2x^3 - 2x^4 + 2x^5 - x^6$
Bound $\leftarrow 6 - 1 = 5$

$\overline{r} \leftarrow$ (part of $\overline{R}$ whose e-order is 0 and total degree is less than 5)
$= -1 + 2x - 2x^2 + 2x^3 - 2x^4$
$[\overline{q'_1}, \overline{r'}] \leftarrow K[[x]]$-approximate-division($\overline{r}, \mathrm{in}_e(P_1), <_r, 5)$
$(\overline{q'_1} = 0, \overline{r'} = \overline{r} = -1 + 2x - 2x^2 + 2x^3 - 2x^4)$
$\overline{Q'_1} \leftarrow 0$
$\overline{R} \leftarrow \overline{R} = -x^7\partial^2 + x^5\partial - x^7\partial - 1 + 2x - 2x^2 + 2x^3 - 2x^4 + 2x^5 - x^6$

The output of $\widehat{\mathcal{D}}$-approximate-division$(P, P_1, 5)$ is

$$\begin{aligned}\partial^2 =&((1 - x + x^2 + \cdots + x^6)\partial + (-1 + x - x^2 + x^3 - x^4)) \cdot ((1 + x)\partial + x) + \\ &- x^7\partial^2 - x^7\partial + x^5\partial - 1 + 2x - 2x^2 + 2x^3 - 2x^4 + 2x^5 - x^6\end{aligned}$$

The correct part of the remainder is

$$-1 + 2x - 2x^2 + 2x^3 - 2x^4.$$

And this part is the same with the part of the remainder of $\widehat{\mathcal{D}}$-division

2.4 Computation of the local b function by the approximate division algorithm in $\widehat{\mathcal{D}}[s]$

We will apply the approximate division algorithm in $\widehat{\mathcal{D}}$ (Algorithm 2.3.2) to obtain the local b function of a polynomial. In the sequel, we use the parameter variable s instead of y, and K denotes the field of complex numbers.

2.4.1 Definition and algorithm of the b function

Definition 2.4.1. (Global b function)
For given $f \in K[x]$, we define the global b function as the monic polynomial $\tilde{b}(s)$ of the least degree which satisfies $\exists P \in D[s]$ s.t. $P \cdot f^{s+1} = \tilde{b}(s)f^s$.

Definition 2.4.2. (Local b function)

For given $f \in K[x]$, we define the local b function at the origin as the monic polynomial $b(s)$ of the least degree which satisfies $\exists P \in \widehat{\mathcal{D}}[s]$ s.t. $P \cdot f^{s+1} = b(s)f^s$.

Example 2.4.3. (b function)

Let $f = x(x+y+1) = x^2 + xy + x$. The global b function of f is $(s+1)^2$, and $(\partial_x^2 + \partial_x\partial_y) \cdot f^{s+1} = (s+1)^2 f^s$ holds. The local b function of f is $s+1$, and $\frac{1}{1+2x+y}\partial_x \cdot f^{s+1} = (s+1)f^s$ holds.

Oaku gave algorithms computing the global b function and the local b function for a given polynomial ([21], [22], [25]). A Gröbner basis method in D is used in these algorithm.

Noro gave an efficient algorithm computing the global b function [19]. In this algorithm, a Gröbner basis method in D, and modular method are used. Especially, to eliminate variables, normal forms with respect to a Gröbner basis are used.

In this paper, we propose an algorithm computing the local b function by utilizing an approximate normal form with respect to a Gröbner basis. Our algorithm is an analogous algorithm with Noro's algorithm. For this purpose, we review a Gröbner basis and a normal form in $\widehat{\mathcal{D}}$ and define an approximate normal form.

2.4.2 Gröbner basis and normal form in $\widehat{\mathcal{D}}[s]$

Although $\widehat{\mathcal{D}}[s]$ is a transcendental object, for ideals in $\widehat{\mathcal{D}}[s]$ generated by elements in $D[s]$, we can compute a Gröbner basis in finite steps by either of the following algorithms.

- Lazard's method in D (using a homogenization)
- method using the Mora division algorithm in D ([9], [10])

Definition 2.4.4. (Normal form in $\widehat{\mathcal{D}}[s]$)

For $P \in \widehat{\mathcal{D}}[s]$ and a Gröbner basis G in $\widehat{\mathcal{D}}[s]$, the remainder of $\widehat{\mathcal{D}}$-division(P, G) (Theorem 2.3.1) is uniquely determined. We call the remainder the normal form of P by G with respect to the monomial order $<$, and we denote it by $\mathrm{NF}(P, G, <)$.

We note that a normal form does not have a finite representation in general.

Lemma 2.4.5. (Ideal membership)

Let G be a Gröbner basis of an ideal $\mathcal{I}$ in $\widehat{\mathcal{D}}[s]$. For $P \in \widehat{\mathcal{D}}[s]$, $P \in \mathcal{I}$ is equivalent to $\mathrm{NF}(P, G, <) = 0$.

Lemma 2.4.6. (Sum of normal forms)

Let G be a Gröbner basis in $\widehat{\mathcal{D}}[s]$. For $P, Q \in \widehat{\mathcal{D}}[s]$, it holds that

$$\mathrm{NF}(P+Q, G, <) = \mathrm{NF}(P, G, <) + \mathrm{NF}(Q, G, <)$$

2.4.3 Algorithm computing $\mathcal{I} \cap K[s]$

For an ideal $\mathcal{I}$ in $\widehat{\mathcal{D}}[s]$, we propose an algorithm computing a generator of $\mathcal{I} \cap K[s]$. In this algorithm, we use approximate normal forms. We fix a Gröbner basis G of $\mathcal{I}$.

For $g(s) = a_l s^l + a_{l-1}s^{l-1} + \cdots + a_0 \in \mathcal{I}$, it holds that

$$\begin{aligned} g(s) \in \mathcal{I} &\Leftrightarrow \mathrm{NF}(g(s), G, <) = 0 \\ &\Leftrightarrow a_l\mathrm{NF}(s^l, G, <) + a_{l-1}\mathrm{NF}(s^{l-1}, G, <) + \cdots + a_0\mathrm{NF}(1, G, <) = 0. \end{aligned}$$

We compute $\mathrm{NF}(s^i, G, <)$ beforehand, and we compute the minimum l and coefficients $a_l, \cdots, a_0 \in K, a_l \neq 0$ which satisfies

$$a_l\mathrm{NF}(s^l, G, <) + a_{l-1}\mathrm{NF}(s^{l-1}, G, <) + \cdots + a_0\mathrm{NF}(1, G, <) = 0.$$

Then the polynomial $a_l s^l + \cdots + a_0$ is a generator of the intersection.

However we cannot generally compute normal forms in $\widehat{\mathcal{D}}[s]$ in finite steps. The normal form $\mathrm{NF}(s^i, G, <)$ has infinite terms in general. Therefore we use an approximate normal form.

Definition 2.4.7. (Approximate normal form)

Let $P \in \widehat{\mathcal{D}}[s]$ and G be a Gröbner basis of an ideal $\mathcal{I}$ in $\widehat{\mathcal{D}}[s]$. We define the approximate normal form of P by G up to total degree N as the remainder of $\widehat{\mathcal{D}}$-approximate-division(P, G, N) (Algorithm 2.3.2). We denote it by $\mathrm{NF}(P, G, <, N)$. By definition, $\mathrm{NF}(P, G, <)$ and $\mathrm{NF}(P, G, <, N)$ are the same up to the total degree $N - 1$.

We suppose that

$$a_l\mathrm{NF}(s^l, G, <, N) + a_{l-1}\mathrm{NF}(s^{l-1}, G, <, N) + \cdots + a_0\mathrm{NF}(1, G, <, N) = 0.$$

Since we use approximate normal forms, it is not always true that $a_l s^l + a_{l-1}s^{l-1} + \cdots + a_0 \in \mathcal{I}$. However we can solve an ideal membership problem for algebraic data in $\widehat{\mathcal{D}}[s]$ by utilizing the Mora division algorithm in $D[s]$. Therefore we need to apply the Mora division algorithm to the gotten candidate $a_l s^l + \cdots + a_0$ in order to check if it is a member of $\mathcal{I}$.

We will explain an algorithm computing $\mathcal{I} \cap K[s]$ by using these ideas. We suppose $\mathcal{I} \cap K[s] \neq \{0\}$. We put

$$\begin{aligned} L_{i,N} &= \left\{(a_0, \cdots, a_i) \in K^{i+1} \mid a_i\mathrm{NF}(s^i, G, <, N) + \cdots + a_0\mathrm{NF}(1, G, <, N) = 0\right\} \\ L_i &= \left\{(a_0, \cdots, a_i) \in K^{i+1} \mid a_i\mathrm{NF}(s^i, G, <) + \cdots + a_0\mathrm{NF}(1, G, <) = 0\right\}, \end{aligned}$$

where G is a Gröbner basis of $\mathcal{I}$ and $N \in \mathbb{N}$. It holds that

$$L_{i,0} \supset L_{i,1} \supset L_{i,2} \supset \cdots \supset L_i.$$

The polynomial $a_i s^i + \cdots + a_0$ which satisfies $L_i \neq \{0\}, L_{i-1} = \{0\}$ and $(a_0, \cdots, a_i) \in L_i$ is a generator of $\mathcal{I} \cap K[s]$. In summary, an algorithm to compute the generator is as follows:

Algorithm 2.4.8. (Computing the generator of $\mathcal{I} \cap K[s]$)

(1) N $\leftarrow$ (a natural number), $d \leftarrow 0$

(2) Find i which satisfies $L_{i-1,N} = \{0\}, L_{i,N} \neq \{0\}$. (Compute $L_{d,N}, L_{d+1,N}, \cdots$. Note that $L_{i-1,N} = \{0\} \Rightarrow L_0 = L_1 = \cdots = L_{i-1} = \{0\}$.)

(3) Take an element $(a_i, \cdots, a_0) \in L_{i,N}$ and put $g(s) = a_i s^i + \cdots + a_0$.

(4) Divide $g(s)$ by G by using the Mora division algorithm. If the remainder is 0, then $g(s)$ is the generator. If not, then set $N \leftarrow N+1, d \leftarrow i$ and goto (2).

2.4.4 Algorithm computing the local b function

For a polynomial $f \in K[x]$, an algorithm of computing the local b function of f at the origin is as follows:

(1) Compute a set of generators of $\mathrm{Ann}_{\widehat{\mathcal{D}}[s]} f^s$. Denote it by H.

(2) Set $\mathcal{I} = \widehat{\mathcal{D}}[s](H \cup \{f\})$. Compute the generator $b(s)$ of $\mathcal{I} \cap K[s]$. The polynomial $b(s)$ is the local b function.

In (1), we take a set of generators of $\mathrm{Ann}_{D[s]} f^s$ as a set of generators of $\mathrm{Ann}_{\widehat{\mathcal{D}}[s]} f^s$. We can compute a set of generators of $\mathrm{Ann}_{D[s]} f^s$ by Oaku's algorithm [22]. In (2), since $\mathcal{I} \cap K[s] \neq \{0\}$ holds (from the existence of the local b function), we can utilize the Algorithm 2.4.8.

We show an example of computing the local b function.

Example 2.4.9. (Local b function of $f = x^2(y+1)^2 z^2$ at the origin)

The global b function of f is $(s+1)^3(s+\frac{1}{2})^3$, and the local b function is $(s+1)^2(s+\frac{1}{2})^2$.

A set of generators H of $\mathrm{Ann}_{\widehat{\mathcal{D}}[s]} f^s$ is

$$\{P_1 = -2s + z\partial_z, P_2 = -x\partial_x + z\partial_z, P_3 = -\partial_y + x\partial_x - y\partial_y\}$$

A Gröbner basis G of $\mathcal{J} = \widehat{\mathcal{D}}[s](H \cup \{f\})$ with respect to $<$ is

$$\{f, P_1, P_2, P_3, P_4 = -xz^3\partial_z - 2xz^2, P_5 = -z^4\partial_z^2 - 2xz^2\partial_x - 2z^3\partial_z - 2z^2\}$$

We set $N = 1$ and will apply the Algorithm 2.4.8 to compute the intersection $\mathcal{J} \cap K[s]$. We note that approximate normal forms of s^i by G up to total degree

$N = 7$ are

$$
\begin{aligned}
\mathrm{NF}(1, G, <, 7) &= 1 \\
\mathrm{NF}(s, G, <, 7) &= \frac{1}{2} z\partial_z \\
\mathrm{NF}(s^2, G, <, 7) &= \frac{1}{4} z^2\partial_z^2 + \frac{1}{4} z\partial_z \\
\mathrm{NF}(s^3, G, <, 7) &= \frac{1}{8} z^3\partial_z^3 + \frac{3}{8} z^2\partial_z^2 + \frac{1}{8} z\partial_z \\
\mathrm{NF}(s^4, G, <, 7) &= -\frac{3}{8} z^3\partial_z^3 - \frac{31}{16} z^2\partial_z^2 - \frac{31}{16} z\partial_z - \frac{1}{4}.
\end{aligned}
$$

Steps of Algorithm 2.4.8 are

$$
\begin{array}{ll}
L_{1,1} = \{0\} & L_1 = \{0\} \\
L_{2,1} \ni (0, 1) & s \xrightarrow[G]{\text{Mora division}} \text{non-zero} \\
L_{2,2} \ni (0, 1) & s \xrightarrow[G]{\text{Mora division}} \text{non-zero} \\
L_{2,3} = \{0\} & L_2 = \{0\} \\
L_{3,3} \ni (0, -\frac{1}{2}, 1) & s^2 - \frac{1}{2}s \xrightarrow[G]{\text{Mora division}} \text{non-zero} \\
L_{3,4} \ni (0, -\frac{1}{2}, 1) & s^2 - \frac{1}{2}s \xrightarrow[G]{\text{Mora division}} \text{non-zero} \\
L_{3,5} = \{0\} & L_3 = \{0\} \\
L_{4,5} \ni (0, \frac{1}{2}, -\frac{3}{2}, 1) & s^3 - \frac{3}{2}s^2 + \frac{1}{2}s \xrightarrow[G]{\text{Mora division}} \text{non-zero} \\
L_{4,6} \ni (0, \frac{1}{2}, -\frac{3}{2}, 1) & s^3 - \frac{3}{2}s^2 + \frac{1}{2}s \xrightarrow[G]{\text{Mora division}} \text{non-zero} \\
L_{4,7} = \{0\} & L_4 = \{0\} \\
L_{5,7} \ni (\frac{1}{4}, \frac{3}{2}, \frac{13}{4}, 3, 1) & s^4 + 3s^3 + \frac{13}{4}s^2 + \frac{3}{2}s + \frac{1}{4} \xrightarrow[G]{\text{Mora division}} 0.
\end{array}
$$

Therefore, the local b function is $s^4 + 3s^3 + \frac{13}{4}s^2 + \frac{3}{2}s + \frac{1}{4} = (s+1)^2(s+\frac{1}{2})$.

2.5 Implementation and timing Data

Our algorithm computing the local b function has been implemented by utilizing computer algebra system "Risa/Asir". The name of our package is "`nk_mora/local-b.rr`". We show the timing data.

Our algorithm computing the local b function consists of the following 3 parts.

(1) Computation of a set of generators of $\mathrm{Ann}_{\widehat{\mathcal{D}}[s]} f^s$ (In fact, computation of a set of generators of $\mathrm{Ann}_{D[s]} f^s$)

(2) Computation of the Gröbner basis of $\mathcal{I} = \mathrm{Ann}_{\widehat{\mathcal{D}}[s]} f^s + \widehat{\mathcal{D}}[s] f$

(3) Computation of the intersection $\mathcal{I} \cap K[s]$ (Algorithm 2.4.8)

To perform the step (1), we use Oaku's algorithm computing $\mathrm{Ann}_{D[s]} f^s$ [21]. In the timing table, we denote this by $(\mathcal{I})$. To perform the step (2), we have the following 2 algorithms.

(2a) Buchberger algorithm utilizing the Mora division algorithm in $D[s]$

(2b) Lazard's homogenized method in $D[s]$

We denote this parts by **(GB-a)** and **(GB-b)**. To perform the step (3), we use the Algorithm 2.4.8. We denote this part by **(localb-nf)**.

We took some examples from [22].

f	$\mathcal{I}$	GB-a	GB-b	localb-nf	localb	deg
$x + y^2 + z^2$	0.00504	0.0190	0.00494	0.00503	0.00612	1
$x^2 + y^2 + z^2$	0.00524	0.00157	0.00224	0.0229	0.0119	2
$x^3 + y^2 + z^2$	0.00637	0.0298	0.0694 + 0.0192	0.0417+0.0244	0.0226	3
$x^4 + y^2 + z^2$	0.00631	0.0317	1.41 + 0.27	0.131 + 0.0884	0.0384	4
$x^5 + y^2 + z^2$	0.00624	0.0306	7.23 + 1.51	0.325 + 0.272	0.0612	5
$x^6 + y^2 + z^2$	0.00619	0.0291	33.0 + 3.25	0.726 + 0.598	0.129	6
$x^7 + y^2 + z^2$	0.00610	0.0281 + 0.0170	137.5 + 24.1	1.26 + 0.674	0.247	7
$x^3 + xy^2 + z^2$	0.0260	0.172	0.107	0.114 + 0.0756	0.0767	4
$x^4 + xy^2 + z^2$	0.0566	0.140	2.29 + 0.338	0.501 + 0.284	0.184	6
$x^5 + xy^2 + z^2$	0.0903	0.194	32.6 + 5.47	0.505 + 0.284	0.173	6
$x^3 + y^4 + z^2$	0.00858	0.118	1.45	1.43 + 0.821	0.197	7
$x^3 + xy^3 + z^2$	0.0110	0.236 + 0.337	1.09	2.94 + 1.40	1.06	8
$x^3 + y^5 + z^2$	0.00979	0.123	18.3 + 2.82	4.36 + 1.80	0.615	9
$x^5 + x^3y^3 + y^5$	0.70	0.28	0.052	—	7.06	7
$x^4 + x^2y^2 + y^4$	0.0061	0.0138	0.0019	0.163	0.153	6
$x^3 + x^2y^2 + y^3$	0.031	0.0748	0.018	33.4	0.0932	4
$x^3 + y^3 + z^3$	0.0062	0.00162	0.0025	0.142	0.112	5
$x^6 + y^4 + z^3$	0.0134	0.052	76.5	131.1	23.2	18
$x^3 + y^2z^2$	0.029	0.0462	0.016	0.868 + 0.108	0.144	7
$(x^3 - y^2z^2)^2$	0.102	7.49 + 1.67	0.0932	17.4 + 1.40	3.99	14
$x^5 - y^2z^2$	0.013	0.0827	0.092	3.77 + 0.279	0.817	13
$x^5 - y^2z^3$	0.0088	0.0265	0.0028	15.9 + 1.22	3.53	17
$x^3 + y^3 - 3xyz$	0.019	0.473	0.020	0.641 + 0.0540	0.0946	5
$x^3 + y^3 + z^3 - 3xyz$	0.013	0.140	0.012	0.0658	0.0207	3
$y(x^5 - y^2z^2)$	0.014	16.7 + 3.43	0.58	58.6 + 5.24	6.10	18
$y(x^3 - y^2z^2)$	0.644	2.65 + 0.668	0.15	4.18 + 0.338	0.711	10
$y((y+1)x^3 - y^2z^2)$	0.284	18.8 + 1.34	3.6 + 0.14	—	10.2	10
$x^6 + x^3y^3 + y^6$	0.004	0.012	0.002	1.38	1.73	10
$x^8 + x^4y^4 + y^8$	0.003	0.058	0.0039	5.25	16.3	14
$x^5 + y^5 + z^5$	0.004	0.018	0.003	6.86	8.58	11
$x^6 + y^6 + z^6$	0.003	0.018	0.003	25.2	56.4	14

- localb — total time of Oaku's algorithm computing the local b function [21]
- deg — degree of the local b function
- machine — CPU : Athlon MP 1800+ (1533MHz) (2 CPU), Memory : 3GB, OS : FreeBSD 4.8

The total time of our algorithm is ($\mathcal{I}$ + **GB-a** + **localb-nf**) and the total time of Oaku's algorithm is **local-b**. Oaku's algorithm(localb) is faster than our algorithm for the first 27 examples in the table. The reasons seems to be as follows:

1. $\widehat{\mathcal{D}}$-approximate-division algorithm (Algorithm 2.3.2) is heavy computation in our theory and implementation. The variable Bound in Algorithm 2.3.2 becomes very large and $K[[x]]$-approximate-division becomes heavy. (It is a question if we can make the Bound smaller with a sharper estimate)

2. Mora division algorithm in D is heavy computation.

Although there are these negative aspects in practice in our algorithm, our algorithm is faster than Oaku's one for the last 4 examples in the table. It is a future question to find conditions so that our algorithm is faster than Oaku's one. One simple observation is that the last 4 examples are homogeneous.

Although, our new method is not always faster than Oaku's one, however the author thinks that it is promising method for some class of problems. In fact, for the last example, Oaku's method spends a lot of time to compute a Groebner basis computation in the Weyl algebra D which is done with highly optimized built-in function in "Risa/Asir". Our implementations of the Mora division in D and $\widehat{\mathcal{D}}$-approximate-division are written in the user language of "Risa/Asir" and further reduction strategies in these algorithm have not yet been highly optimized, which seems to be an interesting research subject.

Chapter 3

Computing Annihilating Ideals

In this chapter, we give an algorithm of computing the annihilating ideal of f^s in $\mathcal{D}_{alg}$. It can be computed by an analogous method in the case of the Weyl algebra [22], [25]. Correctness proofs are also analogous.

We use the following notations :

$$x = (x_1, \cdots, x_n), \partial = (\partial_1, \cdots, \partial_n)$$
$$D = \left\{ \sum a_\alpha(x)\partial^\alpha \mid \alpha \in \mathbb{N}^n, a_\alpha(x) \in \mathbb{C}[x] \right\}$$
$$D\langle t, \partial_t \rangle = \left\{ \sum a_\alpha(x) t^\beta \partial^\alpha \partial_t^\beta \mid \alpha \in \mathbb{N}^n, \beta \in \mathbb{N}, a_\alpha(x) \in \mathbb{C}[x] \right\}$$
$$\mathbb{C}[x]_{\langle x \rangle} = \left\{ \frac{f(x)}{g(x)} \mid f(x), g(x) \in \mathbb{C}[x], g(0) \neq 0 \right\}$$
$$\mathcal{D}_{alg} = \left\{ \sum a_\alpha(x)\partial^\alpha \mid \alpha \in \mathbb{N}^n, a_\alpha(x) \in \mathbb{C}[x]_{\langle x \rangle} \right\}$$
$$\mathcal{D}_{alg}\langle t, \partial_t \rangle = \left\{ \sum a_\alpha(x) t^\beta \partial^\alpha \partial_t^\beta \mid \alpha \in \mathbb{N}^n, \beta \in \mathbb{N}, a_\alpha(x) \in \mathbb{C}[x]_{\langle x \rangle} \right\}$$

3.1 Computing generators of annihilating ideals $\mathrm{Ann}_{D[s]} f^s$

We summarize the algorithm to compute generators of annihilating ideals $\mathrm{Ann}_{D[s]} f^s$ by T.Oaku [22].

A weight vector w is defined as follows.

t	∂_t	u	v	other variables
-1	1	-1	1	0

For a w-homogeneous element $P \in D\langle t, \partial_t \rangle$ whose w-weight is m, $\psi(P)(t\partial_t)$

is defined by

$$\psi(P)(t\partial_t) = \begin{cases} t^m P & (m \geq 0) \\ \partial_t^{-m} P & (m < 0) \end{cases} .$$

We put

$$I_f = D\langle t, \partial_t\rangle \cdot \left\{ t - f, \partial_1 + \frac{\partial f}{\partial x_1}\partial_t, \cdots, \partial_n + \frac{\partial f}{\partial x_n}\partial_t \right\}.$$

It is easy to show that $\mathrm{Ann}_{D\langle t,\partial_t\rangle} f^s = I_f$.

Theorem 3.1.1. (Computing generators of $\mathrm{Ann}_{D[s]} f^s$, [22])
We put

$$J = D\langle t, \partial_t\rangle[u, v] \cdot \left\{ t - uf, \partial_1 + \frac{\partial f}{\partial x_1}u\partial_t, \cdots, \partial_n + \frac{\partial f}{\partial x_n}, 1 - uv \right\}.$$

A Gröbner basis of J with respect to an elimination order for u, v which consists of w-homogeneous elements is denoted by G. We put

$$G_0 = \{\psi(P)(-s-1) \in D[s] \mid P \in G \cap D\langle t, \partial_t\rangle\}.$$

Then, G_0 generates $\mathrm{Ann}_{D[s]} f^s$.

3.2 Computing generators of annihilating ideals $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$

It is analogous to show that

$$\mathrm{Ann}_{\mathcal{D}_{alg}\langle t,\partial_t\rangle} f^s = \mathcal{D}_{alg}\langle t, \partial_t\rangle \cdot \left\{ t - f, \partial_1 + \frac{\partial f}{\partial x_1}\partial_t, \cdots, \partial_n + \frac{\partial f}{\partial x_n}\partial_t \right\}$$

in the case of $\mathcal{D}_{alg}\langle t, \partial_t\rangle$. We denote this ideal by $\mathcal{I}_f$. We can analogously compute generators of $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$ as in the case of $\mathrm{Ann}_{D[s]} f^s$.

Theorem 3.2.1. (Computing generators of $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$)
We put

$$\mathcal{J} = \mathcal{D}_{alg}\langle t, \partial_t\rangle[u, v] \cdot \left\{ t - uf, \partial_1 + \frac{\partial f}{\partial x_1}u\partial_t, \cdots, \partial_n + \frac{\partial f}{\partial x_n}u\partial_t, 1 - uv \right\}$$

We denote an elimination order of u, v in $\mathcal{D}_{alg}\langle t, \partial_t\rangle[u, v]$ by $<$. A Gröbner basis of J with respect to $<$ which consists of w-homogeneous elements is denoted by G. We put $\mathcal{G}_0$

$$\mathcal{G}_0 = \{\psi(P)(-s-1) \mid P \in \mathcal{G} \cap \mathcal{D}_{alg}\langle t, \partial_t\rangle\}.$$

Then, $\mathcal{G}_0$ generates $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$.

The proof of this theorem is almost the same as that of Theorem 3.1.1.
(proof)

We prove $\mathcal{G}_0 \subset \mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$. For $P \in \mathcal{G} \cap \mathcal{D}_{alg}\langle t, \partial_t \rangle$, since $P \in \mathcal{J}$, there exist $Q_0, \cdots, Q_{n+1} \in \mathcal{D}_{alg}\langle t, \partial_t \rangle[u,v]$ such that

$$P = Q_0(t - uf) + \sum_{i=1}^{n} Q_i(\partial_i + \frac{\partial f}{\partial x_i} u \partial_t) + Q_{n+1}(1 - uv).$$

We substitute $u = 1, v = 1$ in the above formula,

$$P = Q_0'(t - f) + \sum_{i=1}^{n} Q_i'(\partial_i + \frac{\partial f}{\partial x_i} \partial_t) \in \mathcal{I}_f$$

Since P is w-homogeneous, $\psi(P)(t\partial_t)$ is $t^m P$ or $\partial_t^m P$. Therefore, $\psi(P)(t\partial_t) \in \mathcal{I}_f$ holds. In other words, we have $\psi(P)(-s-1) \cdot f^s = 0$, and hence $\mathcal{G}_0 \subset \mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$ holds.

Conversely, we prove that $\mathcal{G}_0$ generates $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$. For $P(s) \in \mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$, since $P(-\partial_t t) \in \mathcal{I}_f$, there exist $Q_0, Q_1, \cdots Q_n \in \mathcal{D}_{alg}\langle t, \partial_t \rangle$ such that

$$P(-\partial_t t) = Q_0(t - f) + \sum_{i=1}^{n} Q_i(\partial_i + \frac{\partial f}{\partial x_i} \partial_t).$$

We homogenize the both sides in the above formula by this weight vector $\dfrac{t \quad \partial_t \quad \text{other variable}}{-1 \quad 1 \qquad 0}$, where the variable for homogenization is u. Then, there exist $l, l_0, \cdots, l_n \in \mathbb{N}$ such that

$$u^l P(-\partial_t t) = u^{l_0} Q_0^h(t - uf) + \sum_{i=1}^{n} u^{l_i} Q_i^h(\partial_i + \frac{\partial f}{\partial x_i} u \partial_t) \in \mathcal{J},$$

where Q_i^h is the homogenization of Q_i with respect to the above weight vector.

$$\begin{aligned} P(-\partial_t t) &= (1 - u^l v^l) P(-\partial_t t) + u^l v^l P(-\partial_t t) \\ &= (1 - uv)(1 + uv + \cdots + u^{l-1} v^{l-1}) P(-\partial_t t) + v^l u^l P(-\partial_t t) \end{aligned}$$

$P(-\partial_t t) \in \mathcal{J}$ holds. We put

$$\mathcal{G} = \{P_1, \cdots, P_k, P_{k+1} \cdots, P_r\},$$

and assume that $P_1, \cdots, P_k$ do not include u, v and $P_{k+1}, \cdots, P_r$ include u or v. Hence, we have

$$\mathcal{G}_0 = \{\psi(P_1)(-\partial_t t), \cdots, \psi(P_k)(-\partial_t t)\}.$$

Since $\mathcal{G}$ is a Gröbner basis of $\mathcal{J}$ and the elements in $\mathcal{G}$ are w-homogeneous, when we divide $P(-\partial_t t)$ by $\mathcal{G}$ with respect to $<$ (the Mora division in $\mathcal{D}_{alg}\langle t, \partial_t \rangle[u,v]$), there exist w-homogeneous elements $U_1, \cdots, U_r \in \mathcal{D}_{alg}\langle t, \partial_t \rangle[u,v]$ such that

$$P(-\partial_t t) = \sum_{i=1}^{r} U_i P_i$$

We assume that $U_{k+1} \neq 0$ or $\cdots$ or $U_r \neq 0$. Since $<$ is an elimination order of u, v, it holds that

$$\mathrm{LM}_<(P(-\partial_t t)) < \mathrm{LM}_<(U_i P_i).$$

This is a contradiction. Therefore $U_{k+1} = \cdots = U_r = 0$ holds, and

$$P = \sum_{i=1}^{k} U_i P_i \quad (U_i \in \mathcal{D}_{alg}\langle t, \partial_t \rangle)$$

$P(-\partial_t t)$ is a w-homogeneous element with w-weight 0. We denote the w-weight of P_i by w_i. Then, the w-weight of U_i is $-m_i$.

$$\exists U_i'(t\partial_t) \in \mathcal{D}_{alg}\langle t, \partial_t \rangle \text{ s.t. } U_i = U_i'(t\partial_t) S_i \quad \text{, where } S_i = \begin{cases} U_i'(t\partial_t) t^{m_i} & (m_i \geq 0) \\ U_i'(t\partial_t) \partial_t^{-m_i} & (m_i < 0) \end{cases}$$

And we get

$$P(-\partial_t t) = \sum_{i=1}^{k} U_i'(t\partial_t) S_i P_i = \sum_{i=1}^{k} U_i'(t\partial_t) \psi(P_i)(t\partial_t),$$

$$P(s) = \sum_{i=1}^{k} U_i'(-s-1) \psi(P_i)(-s-1),$$

which implies that the $P(s)$ in $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$ is generated by $\mathcal{G}_0$ in $\mathcal{D}_{alg}[s]$. $\square$

Proposition 3.2.2. (Generators of $\mathrm{Ann}_{D[s]} f^s$ and $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$)
We denote by G_0 a set of generators of $\mathrm{Ann}_{D[s]} f^s$.

$$\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s = \mathcal{D}_{alg}[s] \cdot G_0$$

(proof) It is trivial that $G_0 \subset \mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$.

We take an element $P(s)$ in $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$. There exists a polynomial $g(x)$ such that $g(0) \neq 0$ and $g(x)P(x) \in D[s]$. And $g(x)P(s) \in \mathrm{Ann}_{D[s]} f^s$. We put $G_0 = \{P_1, \cdots, P_k\}$. There exist $Q_1, \cdots, Q_k \in D[s]$ such that

$$g(x)P(s) = \sum_{i=1}^{k} Q_i P_i$$

$$P(s) = \sum_{i=1}^{k} \frac{1}{g(x)} Q_i P_i \quad (\frac{1}{g(x)} Q_i \in \mathcal{D}_{alg}[s])$$

Therefore $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s = \mathcal{D}_{alg}[s] \cdot G_0$ holds. $\square$

3.3 Algorithm of computing $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$

We explain details of our algorithm computing generators of $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$.

Algorithm 3.3.1. (Algorithm computing generators of $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$)
Input : a polynomial f in n variables
Output : generators of $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$

1. We put the ideal

$$\mathcal{J} = \mathcal{D}_{alg}\langle t, \partial_t\rangle[u, v] \cdot \{t - uf, \partial_1 + \frac{\partial f}{\partial x_1} u \partial_t, \cdots, \partial_n + \frac{\partial f}{\partial x_n} u \partial_t, 1 - uv\},$$

   and take the following monomial order $<$:

x_1	$\cdots$	x_n	t	u	v	∂_1	$\cdots$	∂_n	∂_t
0	$\cdots$	0	0	1	1	0	$\cdots$	0	0
0	$\cdots$	0	1	0	0	0	$\cdots$	0	1
0	$\cdots$	0	0	0	0	1	$\cdots$	1	0
-1	$\cdots$	-1	0	0	0	0	$\cdots$	0	0

   (a term order)

   We note that this monomial order $<$ is an elimination order of u, v, a global order with respect to t and a local order with respect to $x_1, \cdots x_n$. We compute a Gröbner basis of $\mathcal{J}$ with respect to $<$. The Gröbner basis is denoted by $\mathcal{G}$.

2. We put

$$\mathcal{G}_0 = \{\psi(P)(-s-1) \mid P \in \mathcal{G} \cap \mathcal{D}_{alg}\langle t, \partial_t\rangle\}.$$

   Then $\mathcal{G}_0$ generates $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$.

The correctness of this algorithm follows from Theorem reflocalAnn.
Let us present an example.

Example 3.3.2. (Computing generators of $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$)
We put $f = x + y(1+x)$. $\mathrm{Ann}_{\mathcal{D}[s]} f^s$ is generated by the following 3 elements :

$$P_1 = -(x + y(1+x))\partial_x + (1+y)s$$
$$P_2 = \partial_y - (1+x)^2 \partial_x + (1+x)s$$
$$P_3 = -(1+y)\partial_y + (1+x)\partial_x.$$

On the other hand, we get the following smaller set of generators of $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$ by Algorithm 3.3.1 :

$$P_3 = -(1+y)\partial_y + (1+x)\partial_x$$
$$P_4 = x(1+y)\partial_y + ((1+x)y - x^2)\partial_x - (1+y)s$$

From Proposition 3.2.2, it holds that

$$\mathcal{D}_{alg}[s] \cdot \{P_1, P_2, P_3\} = \mathcal{D}_{alg}[s] \cdot \{P_3, P_4\}.$$

We note that they generate different $D[s]$ ideals;

$$D[s] \cdot \{P_1, P_2, P_3\} \supsetneq D[s] \cdot \{P_3, P_4\}$$

We have given algorithms computing the local b-function of a polynomial f in chapters 2 and 3. These algorithms need generators of $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$. We use generators of $\mathrm{Ann}_{D[s]} f^s$, which also generate $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$, in these algorithms. For some polynomials f, computing $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$ is faster than that of $\mathrm{Ann}_{D[s]} f^s$ as we see below. This implies that we can obtain local b-functions faster. Moreover, by using Algorithm 3.3.1, we can get the local b-function only by computation in $\mathcal{D}_{alg}$.

Example 3.3.3. (Timing data of computing $\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$)

f	$\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$	$\mathrm{Ann}_{D[s]} f^s$
$x^7 + y^7 + x^4y^4$	0.51 s	8.3 s
$x^{11} + y^{11} + x^6y^6$	6.0 s	759 s
$x^{13} + y^{13} + x^7y^7$	7.6 s	—
$x^{15} + y^{15} + x^8y^8$	10.0s	—
$x^5 + y^5 + x^3y^3$	0.32 s	0.52 s
$x^5 + y^5 + x^4y^4$	—	0.096 s
$x^{1000} + y^3$	0.18 s	1.6 s
$x^{2000} + y^3$	0.18 s	5.3 s
$x^{3000} + y^3$	0.18 s	12.1 s

$\mathrm{Ann}_{D[s]} f^s$: Algorithm based on Theorem 3.1.1
$\mathrm{Ann}_{\mathcal{D}_{alg}[s]} f^s$: Algorithm 3.3.1
Software – Risa/Asir, Machine – CPU : Core2(1.06GHz), Memory : 2 GB

Chapter 4

Interactive User Interface for Computer Algebra

4.1 Introduction

Our objectives of building the interactive user interface are as follows:

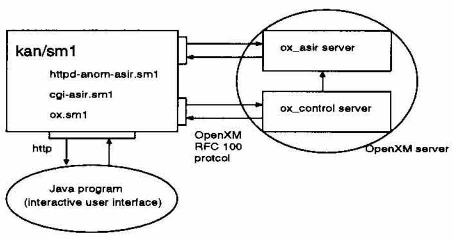
(1) To select reducers in division algorithms and S-pairs in the Buchberger algorithm interactively.

(2) To visualize division algorithms and the Buchberger algorithm and to understand the algorithms intuitively.

(3) To create a user interface of division algorithms and the Buchberger algorithm without using computer algebra system languages.

Objective (1) has a mathematical background. We have studied and implemented division algorithms and the Buchberger algorithm in the ring of differential operators with rational function coefficients whose denominators do not vanish at the origin, $\mathcal{D}_{alg}$ ([17], [10]). In the ring of polynomials and the local ring of that, methods of efficiently computing a remainder and a Gröbner basis have been studied in detail ([7], [8], [11]). However, in the ring $\mathcal{D}_{alg}$, methods of those have not been studied in detail. As far as we have known, no system has satisfied our objectives. Therefore, we have designed an interactive user interface as a tool to understand and improve these algorithms. This system is a tool for us to study algorithms, however it may be useful for educational purposes.

4.2 System Architecture



The proposed system consists of two parts: polynomial computation and user interface. The Polynomial computation part is performed by Risa/Asir. Risa/Asir is an open source computer algebra system, and it is efficient at factorization, Gröbner basis computation ([29]). The user interface part is written in Java. These two parts are connected by OpenXM over HTTP ([31]). We can concentrate on the user interface with this architecture.

4.3 Interactive User Interface for Division Algorithms

As shown in the screenshot (Fig.4.1, Fig.4.2), each ball stands for a monomial. Each row stands for a polynomial. The first row is a divident. The rest are divisors. In this case, $x^{10} + 1$ is divided by $x^2 + xy, xy + y^2$ with respect to the lex order such that $x > y$. The Monomial balls are sorted by the lex order. The possible reducers are emphasized in blue. In the manual mode, when we click a blue ball, the system executes a reduction by the selected reducer. In the automatic mode, the system automatically executes reductions with the current strategy on the clicking of the start button. We can easily switch between these two modes.

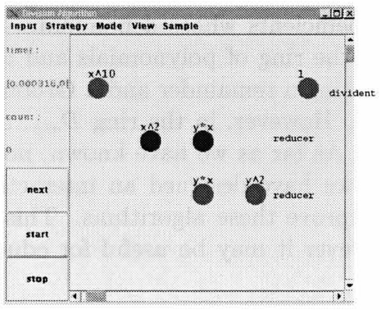


Figure 4.1: initial state

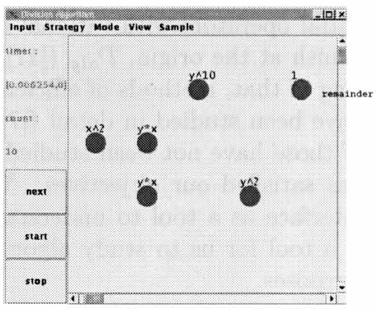
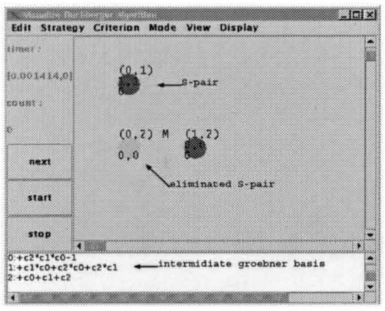


Figure 4.2: result

4.4 Interactive User Interface for the Buchberger Algorithm

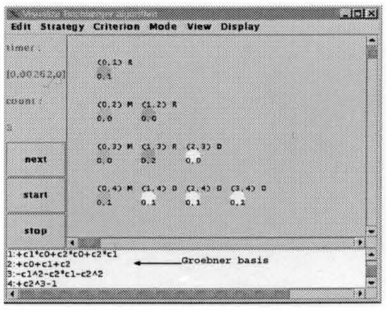
As shown in the screenshots (Fig.4.3, Fig.4.4), each ball stands for an S-pair. A red ball has not yet been divided. A sky blue ball has been divided. The polynomials in the bottom window are intermediate Gröbner basis, and will finally become a Gröbner basis.

When we click a ball, the system executes the division of the ball by the intermediate Gröbner basis. If the remainder is 0, then the ball is eliminated. Else, then the set of S-pairs and the intermediate Gröbner basis are updated. When we click the start button, the system switches to the automatic mode and an S-pair is automatically chosen with the current strategy, either the normal strategy or the sugar strategy. The pairs eliminated by the Buchberger criterion or the Gebauer-Möller criterion are visualized. These pairs are colored white or gray.



```
0:+c2*c1*c0-1
1:+c1*c0+c2*c0+c2*c1
2:+c0+c1+c2
```

Figure 4.3: initial state



```
1:+c1*c0+c2*c0+c2*c1
2:+c0+c1+c2
3:-c1^2-c2*c1-c2^2
4:+c2^3-1
```

Figure 4.4: result

4.5 Application and Conclusion

We show an example of computing a Gröbner basis in the ring $\mathcal{D}_{alg}[s]$. We consider the computation of the Gröbner basis of the left ideal in $\mathcal{D}_{alg}[s]$ generated by $xy + x^3 + y^3, x\partial_x - y\partial_y + 3y^2\partial_x - 3x^2\partial_y, sx + 3sy^2 - xy\partial_y - x^3\partial_y - y^3\partial_y, -3s + 2x\partial_x + y\partial_y + 27sxy - 3x^2\partial_y - 9x^2y\partial_x - 9xy^2\partial_y, 3sy + 9sx^2 - 2xy\partial_x - y^2\partial_y - 3x^3\partial_x - 3x^2y\partial_y$. This example appears in the computation of the local b function of $x^3 + xy + y^3$. We cannot compute the Gröbner basis with the normal strategy or the sugar strategy.

Using the interactive user interface, we can easily try all orders of S-pair selections. However, in any case, the computation stops in the Mora division of an S-pair. We conclude that S-pair selection strategies do not solve the problem, and the implementation of the Mora division algorithm in $\mathcal{D}_{alg}[s]$ should be improved.

This negative discovery was made with our interactive user interface. Nevertheless, our interface can be a useful tool to test new methods of computation such as for playing video games.

Chapter 5

Some Mathematical Results and Questions

In this chapter, we will apply our computational methods to two mathematical questions.

5.1 b-functions of 17 polynomials by Sekiguchi

In the paper [27], Jiro Sekiguchi classified weighted homogeneous polynomials in 3 variables with some nice conditions as discriminants of Weyl groups. The 17 polynomials classified in [27] are decomposed into 3 families by the weights (p, q, r); (I) $(p, q, r) = (2, 3, 4)$, (II) $(p, q, r) = (1, 2, 3)$, (III) $(p, q, r) = (1, 3, 5)$. These families are related with the reflection groups of rank three. In fact, there are three irreducible reflection groups of rank three and their types are A_3, B_3, H_3. The discriminant of the reflection group $W(A_3)$ (resp., $W(B_3)$, $W(H_3)$) is contained in (I) (resp., (II), (III)). Let x, y, z be the variables such that their weights are p, q, r, respectively and let $F(x, y, z)$ be one of the seventeen polynomials. Then he finds not only that the hypersurface of $\mathbb{C}^3$ defined by $F(x, y, z) = 0$ is regarded as a space of 1-parameter deformations of curves in the yz-space of a simple curve singularity whose type is one of the types E_6, E_7, E_8 but also that it defines a Saito free divisor ([27]). The following polynomials are the 17 polynomials.

(I) The case of $(p, q, r) = (2, 3, 4)$

$F_{A,1} = 16x^4z - 4x^3y^2 - 128x^2z^2 + 144xy^2z - 27y^4 + 256z^3$

$F_{A,2} = 2x^6 - 3x^4z + 18x^3y^2 - 18xy^2z + 27y^4 + z^3$

(II) The case of $(p, q, r) = (1, 2, 3)$

$F_{B,1} = z(x^2y^2 - 4y^3 - 4x^3z + 18xyz - 27z^2)$
$F_{B,2} = z(-2y^3 + 4x^3z + 18xyz + 27z^2)$
$F_{B,3} = z(-2y^3 + 9xyz + 45z^2)$
$F_{B,4} = z(9x^2y^2 - 4y^3 + 18xyz + 9z^2)$
$F_{B,5} = xy^4 + y^3z + z^3$
$F_{B,6} = 9xy^4 + 6x^2y^2z - 4y^3z + x^3z^2 - 12xyz^2 + 4z^3$
$F_{B,7} = 1/2xy^4 - 2x^2y^2z - y^3z + 2x^3z^2 + 2xyz^2 + z^3$
(III) The case of $(p, q, r) = (1, 3, 5)$
$F_{H,1} = -50z^3 + (4x^5 - 50x^2y)z^2 + (4x^7y + 60x^4y^2 + 225xy^3)z-$
$135/2y^5 - 115x^3y^4 - 10x^6y^3 - 4x^9y^2$
$F_{H,2} = 100x^3y^4 + y^5 + 40x^4y^2z - 10xy^3z + 4x^5z^2 - 15x^2yz^2 + z^3$
$F_{H,3} = 8x^3y^4 + 108y^5 - 36xy^3z - x^2yz^2 + 4z^3$
$F_{H,4} = y^5 - 2xy^3z + x^2yz^2 + z^3$
$F_{H,5} = x^3y^4 - y^5 + 3xy^3z + z^3$
$F_{H,6} = x^3y^4 + y^5 - 2x^4y^2z - 4xy^3z + x^5z^2 + 3x^2yz^2 + z^3$
$F_{H,7} = xy^3z + y^5 + z^3$
$F_{H,8} = x^3y^4 + y^5 - 8x^4y^2z - 7xy^3z + 16x^5z^2 + 12x^2yz^2 + z^3$

As a next step to study properties of these polynomials, we focus our attention on the determination of their b-functions and make clear the roles of the roots of the b-functions ([18]). There are several methods to compute b-functions. In the paper([18]), we employ two methods. One is the well-known method developed by M. Kashiwara (cf. [14]) which is quite theoretical. The other is an algorithmic method established by Oaku ([21]); This is improved and implemented to the computer algebra system "Risa/Asir" by Noro ([19]). As an advantage of the first method, it is possible to understand the role of the factors of b-functions. By the first method we can find factors of b-functions but cannot completely determine b-functions. By the second method we can completely determine b-functions. In fact, for each of the seventeen polynomials, its b-function $b_F(s)$ is expected to be the product of three factors associated to a stratification, namely, the first one corresponds to the contribution of the hypersurface, the second does to that of the singular locus of the hypersurface and the third does to that of the origin.

Applying these two methods to our case, we obtain the following result.

Theorem 5.1.1. ([18])
If $F(x, y, z)$ is one of the seventeen polynomials, then

$$b_F(s) = (s+1)\tilde{b}^2_{F,a}(s)\tilde{b}^3_{F,a}(s),$$

where $\tilde{b}^2_{F,a}(s)$ and $\tilde{b}^3_{F,a}(s)$ are associated with the singular locus of the hypersurface $F(x, y, z) = 0$ and the origin respectively.

The following polynomials are $\tilde{b}^2_{F,a}(s)$ and $\tilde{b}^3_{F,a}(s)$ by using the first method.

F	$\tilde{b}^2_{F,a}(s)$
$F_{A,1}$	$(s+\frac{5}{6})(s+\frac{7}{6})\cdot(s+1)$
$F_{A,2}$	$(s+\frac{2}{3})(s+\frac{5}{6})(s+1)(s+\frac{7}{6})(s+\frac{4}{3})$
$F_{B,1}$	$(s+1)(s+\frac{5}{6})(s+\frac{3}{4})(s+\frac{5}{4})(s+\frac{7}{6})$
$F_{B,2}$	$(s+1)(s+\frac{2}{3})(s+\frac{5}{6})(s+\frac{4}{3})(s+\frac{7}{6})$
$F_{B,3}$	$(s+1)(s+\frac{3}{5})(s+\frac{4}{5})(s+\frac{6}{5})(s+\frac{7}{5})$
$F_{B,4}$	$(s+1)(s+\frac{5}{8})(s+\frac{7}{8})(s+\frac{9}{8})(s+\frac{11}{8})$
$F_{B,5}$	$(s+\frac{7}{12})(s+\frac{5}{6})(s+\frac{11}{12})(s+\frac{13}{12})(s+\frac{7}{6})(s+\frac{17}{12})$
$F_{B,6}$	$(s+\frac{7}{10})(s+\frac{9}{10})(s+\frac{11}{10})(s+\frac{13}{10})\cdot(s+\frac{5}{6})(s+\frac{7}{6})$
$F_{B,7}$	$(s+\frac{9}{14})(s+\frac{11}{14})(s+\frac{13}{14})(s+\frac{15}{14})(s+\frac{17}{14})(s+\frac{19}{14})$
$F_{H,1}$	$(s+\frac{7}{10})(s+\frac{9}{10})(s+\frac{11}{10})(s+\frac{13}{10})\cdot(s+\frac{5}{6})(s+\frac{7}{6})\cdot(s+1)$
$F_{H,2}$	$(s+\frac{7}{10})(s+\frac{9}{10})(s+\frac{11}{10})(s+\frac{13}{10})\cdot(s+\frac{3}{4})(s+1)(s+\frac{5}{4})$
$F_{H,3}$	$(s+\frac{5}{8})(s+\frac{7}{8})(s+1)(s+\frac{9}{8})(s+\frac{11}{8})\cdot(s+\frac{5}{6})(s+\frac{7}{6})$
$F_{H,4}$	$(s+\frac{7}{12})(s+\frac{3}{4})(s+\frac{11}{12})(s+1)(s+\frac{13}{12})(s+\frac{5}{4})(s+\frac{17}{12})$
$F_{H,5}$	$(s+\frac{7}{12})(s+\frac{5}{6})(s+\frac{11}{12})(s+\frac{13}{12})(s+\frac{7}{6})(s+\frac{17}{12})\cdot(s+1)$
$F_{H,6}$	$(s+\frac{5}{8})(s+\frac{3}{4})(s+\frac{7}{8})(s+1)(s+\frac{9}{8})(s+\frac{5}{4})(s+\frac{11}{8})$
$F_{H,7}$	$(s+\frac{5}{9})(s+\frac{7}{9})(s+\frac{8}{9})(s+1)(s+\frac{10}{9})(s+\frac{11}{9})(s+\frac{13}{9})$
$F_{H,8}$	$(s+\frac{9}{14})(s+\frac{11}{14})(s+\frac{13}{14})(s+\frac{15}{14})(s+\frac{17}{14})(s+\frac{19}{14})\cdot(s+1)$

F	$\tilde{b}^3_{F,a}(s)$
$F_{A,j}$ $(j=1,2)$	$(s+\frac{3}{4})(s+\frac{5}{4})$
$F_{B,j}$ $(j=1,2,3,4)$	$(s+\frac{2}{3})(s+1)(s+\frac{4}{3})$
$F_{B,j}$ $(j=5,6,7)$	$(s+\frac{2}{3})(s+\frac{4}{3})$
$F_{H,j}$ $(j=1,2,\dots,8)$	$(s+\frac{3}{5})(s+\frac{2}{3})(s+\frac{4}{5})(s+1)(s+\frac{6}{5})(s+\frac{4}{3})(s+\frac{7}{5})$

The following polynomials are b-functions of the 17 polynomials computed by the Oaku's algorithm.

F	bfunction of F $(b_F(s))$
$F_{A,1}$	$(s+1)^2(s+\frac{3}{4})(s+\frac{5}{4})(s+\frac{5}{6})(s+\frac{7}{6})$
$F_{A,2}$	$(s+1)^2(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{3}{4})(s+\frac{5}{4})(s+\frac{5}{6})(s+\frac{7}{6})$
$F_{B,1}$	$(s+1)^3(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{3}{4})(s+\frac{5}{4})(s+\frac{6}{6})(s+\frac{7}{6})$
$F_{B,2}$	$(s+1)^3(s+\frac{2}{3})^2(s+\frac{4}{3})^2(s+\frac{5}{6})(s+\frac{7}{6})$
$F_{B,3}$	$(s+1)^3(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{3}{5})(s+\frac{4}{5})(s+\frac{6}{5})(s+\frac{7}{5})$
$F_{B,4}$	$(s+1)^3(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{5}{8})(s+\frac{7}{8})(s+\frac{9}{8})(s+\frac{11}{8})$
$F_{B,5}$	$(s+1)(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{5}{6})(s+\frac{7}{6})(s+\frac{7}{12})(s+\frac{11}{12})(s+\frac{13}{12})(s+\frac{17}{12})$
$F_{B,6}$	$(s+1)(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{5}{6})(s+\frac{7}{6})(s+\frac{7}{10})(s+\frac{9}{10})(s+\frac{11}{10})(s+\frac{13}{10})$
$F_{B,7}$	$(s+1)(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{9}{14})(s+\frac{11}{14})(s+\frac{13}{14})(s+\frac{15}{14})(s+\frac{17}{14})(s+\frac{19}{14})$
$F_{H,1}$	$(s+1)^3(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{3}{5})(s+\frac{4}{5})(s+\frac{6}{5})(s+\frac{7}{5})(s+\frac{5}{6})(s+\frac{7}{6})(s+\frac{7}{10})(s+\frac{9}{10})(s+\frac{11}{10})(s+\frac{13}{10})$
$F_{H,2}$	$(s+1)^3(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{3}{4})(s+\frac{5}{4})(s+\frac{3}{5})(s+\frac{4}{5})(s+\frac{6}{5})(s+\frac{7}{5})(s+\frac{7}{10})(s+\frac{9}{10})(s+\frac{11}{10})(s+\frac{13}{10})$
$F_{H,3}$	$(s+1)^3(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{3}{5})(s+\frac{4}{5})(s+\frac{6}{5})(s+\frac{7}{5})(s+\frac{5}{6})(s+\frac{7}{6})(s+\frac{5}{8})(s+\frac{7}{8})(s+\frac{9}{8})(s+\frac{11}{8})$
$F_{H,4}$	$(s+1)^3(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{3}{4})(s+\frac{5}{4})(s+\frac{3}{5})(s+\frac{4}{5})(s+\frac{6}{5})(s+\frac{7}{5})(s+\frac{7}{12})(s+\frac{11}{12})(s+\frac{13}{12})(s+\frac{17}{12})$
$F_{H,5}$	$(s+1)^3(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{3}{5})(s+\frac{4}{5})(s+\frac{6}{5})(s+\frac{7}{5})(s+\frac{5}{6})(s+\frac{7}{6})(s+\frac{7}{12})(s+\frac{11}{12})(s+\frac{13}{12})(s+\frac{17}{12})$
$F_{H,6}$	$(s+1)^3(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{3}{4})(s+\frac{5}{4})(s+\frac{3}{5})(s+\frac{4}{5})(s+\frac{6}{5})(s+\frac{7}{5})(s+\frac{5}{8})(s+\frac{7}{8})(s+\frac{9}{8})(s+\frac{11}{8})$
$F_{H,7}$	$(s+1)^3(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{3}{5})(s+\frac{4}{5})(s+\frac{6}{5})(s+\frac{7}{5})(s+\frac{5}{9})(s+\frac{7}{9})(s+\frac{8}{9})(s+\frac{10}{9})(s+\frac{11}{9})(s+\frac{13}{9})$
$F_{H,8}$	$(s+1)^3(s+\frac{2}{3})(s+\frac{4}{3})(s+\frac{3}{5})(s+\frac{4}{5})(s+\frac{6}{5})(s+\frac{7}{5})(s+\frac{9}{14})(s+\frac{11}{14})(s+\frac{13}{14})(s+\frac{15}{14})(s+\frac{17}{14})(s+\frac{19}{14})$

We prove our theorem by case by case checks.

5.2 Annihilating ideals of powers of polynomials

Let f be a polynomial and $b(s)$ the b-function. We denote by R the set of the roots of $b(s) = 0$.

The following fact is well known.

Theorem 5.2.1. If α is not in $R + \mathbb{Z}_{>0}$,

$$\mathrm{Ann}_{D[s]} f^s|_{s \to \alpha} = \mathrm{Ann}_D f^\alpha, \tag{5.1}$$

where $\mathrm{Ann}_{D[s]} f^s|_{s\to\alpha} = \left\{P(\alpha) \in D \mid P(s) \in \mathrm{Ann}_{D[s]} f^s\right\}$.

When we have worked together on the question in the previous section, Jiro Sekiguchi posed the following question.
"Study the relation between $\mathrm{Ann}_{D[s]} f^s|_{s\to\alpha}$ and $\mathrm{Ann}_D f^\alpha$ when $\alpha \in R + \mathbb{Z}_{>0}$".

We will call it the "Sekiguchi's question on annihilating ideal". As the first step to consider his question, we will computationally study if the next relation holds for polynomials with Klein singularities. Our conjecture is the following :

Conjecture 5.2.2.

$$\mathrm{Ann}_{D[s]} f^s|_{s\to\alpha} \subsetneq \mathrm{Ann}_D f^\alpha \tag{5.2}$$

for all polynomials f with Klein singularities and $\alpha \in R + \mathbb{Z}_{>0}$.

We have an algorithm computing $\mathrm{Ann}_{D[s]} f^s|_{s\to\alpha}$ ([26, Algorithm5.3.15]). This algorithm uses syzygies in D.

Next examples are computational evidences for the conjecture. We checked the above relation in the case of $f(x,y,z) = A_1, \cdots, A_5, D_4, D_5, E_6, E_7, E_8$ and $\alpha \in R + 1$.

Example 5.2.3. ($f = x^2 + y^2 + z^2, A_1$)
Let $f = x^2 + y^2 + z^2$. The roots of the b-function of f are $-1, -\frac{3}{2}$.

$$\mathrm{Ann}_{D[s]} f^s = D \cdot \{x\partial_x + y\partial_y + z\partial_z - 2s, y\partial_x - x\partial_y, z\partial_x - x\partial_z, z\partial_y - y\partial_z\}$$

$$\mathrm{Ann}_D f^0 = D \cdot \{\underline{\partial_x}, \underline{\partial_y}, \underline{\partial_z}, x\partial_x + y\partial_y + z\partial_z, y\partial_x - x\partial_y, z\partial_x - x\partial_z, z\partial_y - y\partial_z\}$$

$$\mathrm{Ann}_D f^{-\frac{1}{2}} = D \cdot \{\underline{\partial_x^2 + \partial_y^2 + \partial_z^2}, x\partial_x + y\partial_y + z\partial_z - 1, y\partial_x - x\partial_y, z\partial_x - x\partial_z, z\partial_y - y\partial_z\}$$

The underlined elements are not in $\mathrm{Ann}_D[s] f^s|_{s\to\alpha}$.

Example 5.2.4. ($n = 3, A_1, A_2, A_3, A_4$)
Let $\alpha \in R + 1$ and f be $x^{k+1} + y^2 + z^2$ $(k = 1, 2, 3, 4)$. These result shows that there are elements in $\mathrm{Ann}_D f^\alpha \setminus (\mathrm{Ann}_{D[s]} f^s|_{s\to\alpha})$.

```
x^2+y^2+z^2
A :
0
[-dz,-dy,dx]

F :
x^2+y^2+z^2
A :
-1/2
[-dx^2-dy^2-dz^2]
```

```
F :
x^3+y^2+z^2
A :
0
[-dz,-dy,dx]

F :
x^3+y^2+z^2
A :
-1/3
[8*dx^3-27*y*dy^3-9*dy^2-54*y*dz^2*dy-27*z*dz^3-63*dz^2]

F :
x^3+y^2+z^2
A :
-2/3
[-4*dx^2-9*x*dy^2-9*x*dz^2,
8*dx^3-27*y*dy^3-45*dy^2-54*y*dz^2*dy-27*z*dz^3-99*dz^2]

F :
x^4+y^2+z^2
A :
0
[-dz,-dy,dx]

F :
x^4+y^2+z^2
A :
-1/4
[-dx^4-16*y^2*dy^4-24*y*dy^3-48*y^2*dz^2*dy^2+(-48*y^2-16*z^2)*dz^4-72*z*dz^3-48*dz^2]

F :
x^4+y^2+z^2
A :
-1/2
[dx^3-8*y*x*dy^3-4*x*dy^2-16*y*x*dz^2*dy-8*z*x*dz^3-20*x*dz^2,
dx^4+16*y^2*dy^4+48*y*dy^3+(48*y^2*dz^2+12)*dy^2+48*y*dz^2*dy+(48*y^2+16*z^2)*dz^4+96*z*dz^3+108*dz^2]

F :
x^4+y^2+z^2
A :
-3/4
[-dx^2-4*x^2*dy^2-4*x^2*dz^2,
-dx^3+8*y*x*dy^3+12*x*dy^2+16*y*x*dz^2*dy+8*z*x*dz^3+28*x*dz^2,
-dx^4-16*y^2*dy^4-72*y*dy^3+(-48*y^2*dz^2-48)*dy^2-96*y*dz^2*dy+(-48*y^2-16*z^2)*dz^4-120*z*dz^3-192*dz^2]


F :
x^5+y^2+z^2
A :
0
[-dz,-dy,dx]

F :
x^5+y^2+z^2
A :
-1/5
[32*dx^5-3125*y^3*dy^5-11250*y^2*dy^4+(-12500*y^3*dz^2-4875*y)*dy^3+(-15000*y^2*dz^2+75)*dy^2+(-18750*y^3*dz^4-13500*y*dz^2)*dy+
(-12500*z*y^2-3125*z^3)*dz^5+(-40000*y^2-23750*z^2)*dz^4-38625*z*dz^3-9675*dz^2]

F :
x^5+y^2+z^2
A :
-2/5
[-16*dx^4-625*y^2*x*dy^4-1125*y*x*dy^3+(-1875*y^2*x*dz^2-75*x)*dy^2-375*y*x*dz^2*dy+(-1875*y^2-625*z^2)*x*dz^4-3000*z*x*dz^3-2325*x*dz^2,
-32*dx^5+3125*y^3*dy^5+16250*y^2*dy^4+(12500*y^3*dz^2+13875*y)*dy^3+(30000*y^2*dz^2+525)*dy^2+
(18750*y^3*dz^4+16500*y*dz^2)*dy+(12500*z*y^2+3125*z^3)*dz^5+(55000*y^2+28750*z^2)*dz^4+62625*z*dz^3+28275*dz^2]

F :
x^5+y^2+z^2
A :
-3/5
[8*dx^3-125*y*x^2*dy^3-75*x^2*dy^2-250*y*x^2*dz^2*dy-125*z*x^2*dz^3-325*x^2*dz^2,
16*dx^4+625*y^2*x*dy^4+1875*y*x*dy^3+(1875*y^2*x*dz^2+525*x)*dy^2+1875*y*x*dz^2*dy+(1875*y^2+625*z^2)*x*dz^4+3750*z*x*dz^3+4275*x*dz^2,
32*dx^5-3125*y^3*dy^5-21250*y^2*dy^4+(-12500*y^3*dz^2-28875*y)*dy^3+(-45000*y^2*dz^2-4725)*dy^2+
(-18750*y^3*dz^4-31500*y*dz^2)*dy+(-12500*z*y^2-3125*z^3)*dz^5+(-70000*y^2-33750*z^2)*dz^4-92625*z*dz^3-62475*dz^2]

F :
x^5+y^2+z^2
A :
-4/5
[-4*dx^2-25*x^3*dy^2-25*x^3*dz^2,
-8*dx^3+125*y*x^2*dy^3+175*x^2*dy^2+250*y*x^2*dz^2*dy+125*z*x^2*dz^3+425*x^2*dz^2,
-16*dx^4-625*y^2*x*dy^4-2625*y*x*dy^3+(-1875*y^2*x*dz^2-1575*x)*dy^2-3375*y*x*dz^2*dy+(-1875*y^2-625*z^2)*x*dz^4-4500*z*x*dz^3-6825*x*dz^2,
-32*dx^5+3125*y^3*dy^5+26250*y^2*dy^4+(12500*y^3*dz^2+49875*y)*dy^3+(60000*y^2*dz^2+17325)*dy^2+
(18750*y^3*dz^4+58500*y*dz^2)*dy+(12500*z*y^2+3125*z^3)*dz^5+(85000*y^2+38750*z^2)*dz^4+128625*z*dz^3+117075*dz^2]
```

Example 5.2.5. $(n = 3, D_4, D_5)$

Let $\alpha \in R+1$ and f be $x^2y + y^k + z^2$ $(k = 4, 5)$. These result shows there are elements in $\mathrm{Ann}_D f^\alpha \setminus (\mathrm{Ann}_{D[s]} f^s|_{s\to\alpha})$.

```
F :
y*x^2+y^3+z^2
A :
0
[-dz,dy,-dx]

F :
y*x^2+y^3+z^2
A :
-1/6
[-72*y*dx^4+(-48*y*dy^2+(-180*z*dz+12)*dy)*dx^2+24*y*dy^4+(12*z*dz+76)*dy^3+81*z^2*dz^4+81*z*dz^3-9*dz^2]

F :
y*x^2+y^3+z^2
A :
-1/2
[-2*dx^3+(-2*dy^2-3*y*dz^2)*dx-x*dz^2*dy,
(-4*y*dy+6*z*dz-2)*dx^2-4*y*dy^3+(-2*z*dz-10)*dy^2+9*z*y*dz^3+3*y*dz^2,
-4*y*dx^3+(-4*y*dy^2+(-8*z*dz-8)*dy)*dx-3*z*x*dz^3-x*dz^2,
24*y*dx^4+(16*y*dy^2+(60*z*dz+36)*dy)*dx^2-8*y*dy^4+(-4*z*dz-28)*dy^3-27*z^2*dz^4-81*z*dz^3-15*dz^2]

F :
y*x^2+y^3+z^2
A :
-5/6
[-6*y*dx^2-6*y*dy^2+(-3*z*dz-11)*dy-9*y^2*dz^2,
(-12*y*dy+18*z*dz+6)*dx^2-12*y*dy^3+(-6*z*dz-34)*dy^2+27*z*y*dz^3+45*y*dz^2,
-12*y*dx^3+(-12*y*dy^2+(-24*z*dz-40)*dy)*dx-9*z*x*dz^3-15*x*dz^2,
72*y*dx^4+(48*y*dy^2+(180*z*dz+228)*dy)*dx^2-24*y*dy^4+(-12*z*dz-92)*dy^3-81*z^2*dz^4-405*z*dz^3-315*dz^2]
0.004sec(0.009452sec)

F :
y*x^2+y^4+z^2
A :
0
[-dz,dy,dx]

F :
y*x^2+y^4+z^2
A :
-1/8
[216*y^2*dx^4+(-18*y^2*dy^3-66*y*dy^2+(348*z*dz-204)*dy)*dx^2-8*y*dy^5+(-16*z*dz-45)*dy^4+6*x^2*dz^2*dy^3+(80*z*y*dz^3+8*y*dz^2)*dy-
128*z^2*dz^4-16*z*dz^3+24*dz^2,

(27*z*dy^2+216*y^2*dz)*dx^4+((36*z*y*dz^2+24*y*dz)*dy^2+(420*z*dz^2+216*dz)*dy)*dx^2-8*y*dz*dy^5+(-16*z*dz^2-61*dz)*dy^4+
6*x^2*dz^3*dy^3+(80*z*y*dz^4+88*y*dz^3)*dy-128*z^2*dz^5-272*z*dz^4+8*dz^3,

-243*z*y*dx^6+(-216*z*dy^2+108*y^2*dz)*dx^4+(93*y*dz*dy^2+(504*z^2*dz^3+84*z*dz^2-162*dz)*dy)*dx^2+32*y*dz*dy^5+(52*z*dz^2+220*dz)*dy^4-
18*x^2*dz^3*dy^3+(-248*z*y*dz^4-268*y*dz^3)*dy-192*z^3*dz^6-784*z^2*dz^5-544*z*dz^4-188*dz^3]


F :
y*x^2+y^4+z^2
A :
-3/8
[(18*y^2*dy^2+108*y*dy-72*z*dz+36)*dx^2+8*y*dy^4+(16*z*dz+39)*dy^3-6*x^2*dz^2*dy^2-96*z*y*dz^3-48*y*dz^2,

-27*z*dy*dx^4+((-36*z*y*dz^2+36*y*dz)*dy-108*z*dz^2-72*dz)*dx^2+8*y*dz*dy^4+(16*z*dz^2+55*dz)*dy^3-6*x^2*dz^3*dy^2-96*z*y*dz^4-144*y*dz^3,

243*z*y*dx^6+(216*z*dy^2-810*y^2*dz)*dx^4+(-351*y*dz*dy^2+(-504*z^2*dz^3-1476*z*dz^2-1017*dz)*dy)*dx^2-32*y*dz*dy^5+(-52*z*dz^2-228*dz)*dy^4+18*x^2*dz^3*dy^3+
(312*z*y*dz^4+456*y*dz^3)*dy+192*z^3*dz^6+1584*z^2*dz^5+3168*z*dz^4+1296*dz^3]

F :
y*x^2+y^4+z^2
A :
-1/2
[-9*y^2*dx^3+(-4*y*dy^2+(-10*z*dz-8)*dy)*dx-4*z*x*dz^3-x*dz^2,

27*z*dx^5+(36*z*y*dz^2+9*y*dz)*dx^3+(-8*y*dz*dy^3+(-20*z*dz^2-44*dz)*dy^2)*dx+(-8*z*x*dz^4-10*x*dz^3)*dy,

27*z*y*dx^5+12*z*dy^2*dx^3+((-20*z*y*dz^2+4*y*dz)*dy^2+(-48*z^2*dz^3-124*z*dz^2+18*dz)*dy)*dx-16*z^2*x*dz^5-32*z*x*dz^4+5*x*dz^3,

-81*z*y*dx^6-36*z*dy^2*dx^4+((48*z*y*dz^2-36*y*dz)*dy^2+(168*z^2*dz^3+462*z*dz^2-18*dz)*dy)*dx^2+(-4*z*dz^2-8*dz)*dy^4+2*x^2*dz^3*dy^3+(96*z*y*dz^4+136*y*dz^3)*dy-
64*z^3*dz^6-432*z^2*dz^5-364*z*dz^4+191*dz^3]

F :
y*x^2+y^4+z^2
A :
-5/8
[(18*y^2*dy+54*y)*dx^2+8*y*dy^3+(16*z*dz+33)*dy^2-6*x^2*dz^2*dy+24*y^2*dz^2,

27*z*dx^4+36*z*y*dz^2*dx^2-8*y*dz*dy^3+(-16*z*dz^2-49*dz)*dy^2+6*x^2*dz^3*dy-24*y^2*dz^3,

243*z*y*dx^6+(216*z*dy^2-648*y^2*dz)*dx^4+(-225*y*dz*dy^2+(-504*z^2*dz^3-1908*z*dz^2-1368*dz)*dy)*dx^2-32*y*dz*dy^5+(-52*z*dz^2-236*dz)*dy^4+
18*x^2*dz^3*dy^3+(120*z*y*dz^4+324*y*dz^3)*dy+192*z^3*dz^6+1872*z^2*dz^5+4416*z*dz^4+2268*dz^3]
```

```
F :
y*x^2+y^4+z^2
A :
-7/8
[-18*y^2*dx^2-8*y*dy^2+(-16*z*dz-27)*dy+6*x^2*dz^2,

-27*z*dx^4+(-36*z*y*dz^2-36*y*dz)*dx^2+8*y*dz*dy^3+(16*z*dz^2+51*dz)*dy^2-6*x^2*dz^3*dy,

27*z*y*dx^4+12*z*dy^2*dx^2+(-20*z*y*dz^2-11*y*dz)*dy^2+(-40*z^2*dz^3-168*z*dz^2-58*dz)*dy+12*z*x^2*dz^4+6*x^2*dz^3,

27*z*y*dx^5+12*z*dy^2*dx^3+((-20*z*y*dz^2-11*y*dz)*dy^2+(-48*z^2*dz^3-196*z*dz^2-66*dz)*dy)*dx-16*z^2*x*dz^5-68*z*x*dz^4-22*x*dz^3,

243*z*y*dx^6+216*z*dy^2*dx^4+(117*y*dz*dy^2+(-504*z^2*dz^3-1800*z*dz^2-522*dz)*dy)*dx^2-32*y*dz*dy^5+(-52*z*dz^2-244*dz)*dy^4+18*x^2*dz^3*dy^3+
(-216*z*y*dz^4-588*y*dz^3)*dy+192*z^3*dz^6+1872*z^2*dz^5+3720*z*dz^4+336*dz^3]
```

Example 5.2.6. $(n = 3, E_6, E_7, E_8)$

Let $\alpha \in R + 1$ and f be $x^3 + y^4 + z^2, x^3y + xy^3 + z^2$ or $x^3 + y^5 + z^2$. These result shows that there are elements in $\mathrm{Ann}_D f^\alpha \setminus (\mathrm{Ann}_{D[s]} f^s|_{s\to\alpha})$.

```
F :
x^3+y^4+z^2
A :
0
[-dz,-dy,-dx]

F :
x^3+y^4+z^2
A :
-1/12
[(-576*y^2*dy^2+(768*z*y*dz-2688*y)*dy-768*z^2*dz^2+256*z*dz-1664)*dx^3+243*y*dy^5+(648*z*dz+567)*dy^4+(648*z*y*dz^3+72*y*dz^2)*dy+
2592*z^3*dz^5+7776*z^2*dz^4+3384*z*dz^3+216*dz^2]

F :
x^3+y^4+z^2
A :
-1/3
[(64*y^2*dy-64*z*y*dz+128*y)*dx^3-27*y*dy^4+(-81*z*dz-72)*dy^3+18*y^2*dz^2*dy+216*z^2*y*dz^4+288*z*y*dz^3+36*y*dz^2,

(-576*y^2*dy^2+(768*z*y*dz-2112*y)*dy-768*z^2*dz^2-1280*z*dz-1472)*dx^3+243*y*dy^5+(648*z*dz+810)*dy^4+(2592*z*y*dz^3+1368*y*dz^2)*dy+
2592*z^3*dz^5+15552*z^2*dz^4+20232*z*dz^3+3528*dz^2]

F :
x^3+y^4+z^2
A :
-5/12
[(16*y^2*dy^2+(-32*z*y*dz+64*y)*dy+64*z^2*dz^2+64*z*dz+32)*dx^2+9*x*dy^4+144*z^2*x*dz^4+216*z*x*dz^3,
(576*y^2*dy^2+(-768*z*y*dz+2688*y)*dy+
768*z^2*dz^2-1280*z*dz+640)*dx^3-243*y*dy^5+(-648*z*dz-1215)*dy^4+(4536*z*y*dz^3+2520*y*dz^2)*dy-2592*z^3*dz^5-7776*z^2*dz^4+12168*z*dz^3+7560*dz^2]

F :
x^3+y^4+z^2
A :
-7/12
[64*y^2*dx^3-27*y*dy^3+(-108*z*dz-99)*dy^2-216*z*y^2*dz^3-72*y^2*dz^2,

(-64*y^2*dy+64*z*y*dz-64*y)*dx^3+27*y*dy^4+(81*z*dz+99)*dy^3-126*y^2*dz^2*dy-216*z^2*y*dz^4-720*z*y*dz^3-522*y*dz^2,

(-576*y^2*dy^2+(768*z*y*dz-1536*y)*dy-768*z^2*dz^2-2816*z*dz-2432)*dx^3+243*y*dy^5+(648*z*dz+1053)*dy^4+(4536*z*y*dz^3+6552*y*dz^2)*dy+
2592*z^3*dz^5+23328*z^2*dz^4+52632*z*dz^3+24984*dz^2]

F :
x^3+y^4+z^2
A :
-2/3
[(16*y^2*dy-32*z*y*dz+16*y)*dx^2+9*x*dy^3-72*z*y*x*dz^3-36*y*x*dz^2,

(64*y^2*dy-64*z*y*dz+128*y)*dx^3-27*y*dy^4+(-81*z*dz-144)*dy^3-126*y^2*dz^2*dy+216*z^2*y*dz^4+576*z*y*dz^3-252*y*dz^2,

(16*y^2*dy^2+(-32*z*y*dz+48*y)*dy+64*z^2*dz^2+128*z*dz+48)*dx^2+9*x*dy^4+144*z^2*x*dz^4+432*z*x*dz^3+108*x*dz^2,

(576*y^2*dy^2+(-768*z*y*dz+2112*y)*dy+768*z^2*dz^2+256*z*dz+448)*dx^3-243*y*dy^5+(-648*z*dz-1458)*dy^4+(2592*z*y*dz^3+3816*y*dz^2)*dy-
2592*z^3*dz^5-15552*z^2*dz^4-9864*z*dz^3+9432*dz^2]

F :
x^3+y^4+z^2
A :
-11/12
[-16*y^2*dx^2-9*x*dy^2-36*y^2*x*dz^2,

64*y^2*dx^3-27*y*dy^3+(-108*z*dz-171)*dy^2-216*z*y^2*dz^3-360*y^2*dz^2,

(16*y^2*dy-32*z*y*dz)*dx^2+9*x*dy^3-72*z*y*x*dz^3-108*y*x*dz^2,

(-64*y^2*dy+64*z*y*dz-64*y)*dx^3+27*y*dy^4+(81*z*dz+171)*dy^3+18*y^2*dz^2*dy-216*z^2*y*dz^4-1008*z*y*dz^3-666*y*dz^2,

(16*y^2*dy^2+(-32*z*y*dz+32*y)*dy+64*z^2*dz^2+192*z*dz+96)*dx^2+9*x*dy^4+144*z^2*x*dz^4+648*z*x*dz^3+432*x*dz^2,
(-576*y^2*dy^2+(768*z*y*dz-1536*y)*dy-768*z^2*dz^2-1792*z*dz-1408)*dx^3+243*y*dy^5+(648*z*dz+1701)*dy^4+(-648*z*y*dz^3-1224*y*dz^2)*dy+
2592*z^3*dz^5+23328*z^2*dz^4+47448*z*dz^3+17208*dz^2]

F :
y*x^3+y^3*x+z^2
A :
0
[-dz,dy,dx]

F :
y*x^3+y^3*x+z^2
A :
-1/4
[(-4*y*dy-6*z*dz-3)*dx^4+(-28*z*dz+2)*dy^2*dx^2+(32*z^2*y*dz^4+48*z*y*dz^3)*dx+4*y*dy^5+(2*z*dz+17)*dy^4+(32*z^2*x*dz^4+48*z*x*dz^3)*dy,

(-12*y^2*dy^2+(-10*z*y*dz-37*y)*dy+12*z^2*dz^2-6*z*dz-12)*dx^3+((-52*z*y*dz+38*y)*dy^3+(-24*z^2*dz^2-132*z*dz+96)*dy^2)*dx-12*y*x*dy^5+
(-6*z*x*dz-51*x)*dy^4-64*z^3*y*dz^5-192*z^2*y*dz^4-48*z*y*dz^3,

(-12*y^2*dy-18*z*y*dz-9*y)*dx^4+((-52*z*y*dz+38*y)*dy^2+(-80*z^2*dz^2-104*z*dz+56)*dy)*dx^2+12*y^2*dy^5+(38*z*y*dz+83*y)*dy^4+
(16*z^2*dz^2+136*z*dz+104)*dy^3+64*z^3*x*dz^5+192*z^2*x*dz^4+48*z*x*dz^3,

(12*y^2*dy+18*z*y*dz+9*y)*dx^5+(-12*y^2*dy^3+(42*z*y*dz-99*y)*dy^2+(92*z^2*dz^2+88*z*dz-105)*dy)*dx^3+(-12*y^2*dy^5+(-90*z*y*dz-45*y)*dy^4+
(-40*z^2*dz^2-320*z*dz+30)*dy^3)*dx-12*y*x*dy^6+(-6*z*x*dz-63*x)*dy^5+128*z^4*dz^6+960*z^3*dz^5+1440*z^2*dz^4+240*z*dz^3]

F :
y*x^3+y^3*x+z^2
A :
-1/2
[-4*dy*dx^3-3*y^2*dz^2*dx^2-4*dy^3*dx+(-3*x^2+6*y^2)*dz^2*dy^2+(12*z*y*dz^3+42*y*dz^2)*dy-4*z^2*dz^4-6*z*dz^3+18*dz^2,

(-2*y*dy-3*z*dz-3)*dx^3+(-2*y*dy^3+(-9*z*dz-9)*dy^2)*dx+((-6*z*x^2+6*z*y^2)*dz^3+(-3*x^2+3*y^2)*dz^2)*dy+16*z^2*y*dz^4+60*z*y*dz^3+18*y*dz^2,

(2*y*dy^2+(-5*z*dz+1)*dy)*dx^2+(-6*z*y^2*dz^3-3*y^2*dz^2)*dx+2*y*dy^4+(z*dz+7)*dy^3+(-6*z*y*x*dz^3-3*y*x*dz^2)*dy+4*z^2*x*dz^4+6*z*x*dz^3,

(2*y*dy+3*z*dz+3)*dx^4+(14*z*dz+6)*dy^2*dx^2+(-16*z^2*y*dz^4-48*z*y*dz^3-12*y*dz^2)*dx-2*y*dy^5+(-z*dz-9)*dy^4+(-16*z^2*x*dz^4-48*z*x*dz^3-12*x*dz^2)*dy,

(-2*y^2*dy-3*z*y*dz-3*y)*dx^3+(4*y^2*dy^3+(-12*z*y*dz+24*y)*dy^2+(-6*z^2*dz^2-24*z*dz+24)*dy)*dx-6*y*x*dy^4+(-3*z*x*dz-21*x)*dy^3+16*z^2*y^2*dz^4+24*z*y^2*dz^3,

(-2*y^2*dy^2+(-z*y*dz-7*y)*dy+3*z^2*dz^2+3*z*dz-3)*dx^2-2*y^2*dy^4+(-7*z*y*dz-13*y)*dy^3+(-3*z^2*dz^2-21*z*dz-15)*dy^2+8*z^2*y*x*dz^4+12*z*y*x*dz^3,

(-6*y^2*dy-9*z*y*dz-9*y)*dx^3+(-4*y^2*dy^3+(-28*z*y*dz-16*y)*dy^2+(-34*z^2*dz^2-88*z*dz-8)*dy)*dx-2*y*x*dy^4+(-z*x*dz-7*x)*dy^3-16*z^2*x^2*dz^4-24*z*x^2*dz^3,

(6*y^2*dy^2+(5*z*y*dz+21*y)*dy-6*z^2*dz^2-3*z*dz+9)*dx^3+((26*z*y*dz-6*y)*dy^3+(12*z^2*dz^2+78*z*dz-18)*dy^2)*dx+6*y*x*dy^5+(3*z*x*dz+27*x)*dy^4+
32*z^3*y*dz^5+160*z^2*y*dz^4+120*z*y*dz^3,

(-6*y^2*dy-9*z*y*dz-9*y)*dx^4+((-26*z*y*dz+6*y)*dy^2+(-40*z^2*dz^2-92*z*dz+12)*dy)*dx^2+6*y^2*dy^5+(19*z*y*dz+51*y)*dy^4+(8*z^2*dz^2+76*z*dz+84)*dy^3+
32*z^3*x*dz^5+160*z^2*x*dz^4+120*z*x*dz^3,

(-6*y^2*dy-9*z*y*dz-9*y)*dx^5+(6*y^2*dy^3+(-21*z*y*dz+39*y)*dy^2+(-46*z^2*dz^2-90*z*dz+42)*dy)*dx^3+(6*y^2*dy^5+(45*z*y*dz+45*y)*dy^4+
(20*z^2*dz^2+180*z*dz+60)*dy^3)*dx+6*y*x*dy^6+(3*z*x*dz+33*x)*dy^5-64*z^4*dz^6-640*z^3*dz^5-1520*z^2*dz^4-720*z*dz^3]

F :
y*x^3+y^3*x+z^2
A :
-3/4
[(-4*y*dy-6*z*dz-9)*dx^2-4*y*dy^3+(-2*z*dz-11)*dy^2-16*y^2*x*dz^2*dy-16*z*y*x*dz^3-56*y*x*dz^2,

(4*y*dy+6*z*dz+9)*dx^3+(4*y*dy^3+(18*z*dz+27)*dy^2)*dx+((12*z*x^2-12*z*y^2)*dz^3+(18*x^2-18*y^2)*dz^2)*dy-32*z^2*y*dz^4-168*z*y*dz^3-132*y*dz^2,

(-4*y*dy^2+(10*z*dz+3)*dy)*dx^2+(12*z*y^2*dz^3+18*y^2*dz^2)*dx-4*y*dy^4+(-2*z*dz-15)*dy^3+(12*z*y*x*dz^3+18*y*x*dz^2)*dy-8*z^2*x*dz^4-24*z*x*dz^3-6*x*dz^2,

(12*y^2*dy^2+(-6*z*y*dz+39*y)*dy-4*z^2*dz^2-10*z*dz+16)*dx-12*y*x*dy^3+(-6*z*x*dz-33*x)*dy^2-16*z*y^3*dz^3-8*y^3*dz^2,

(4*y^2*dy+6*z*y*dz+9*y)*dx^2+4*y^2*dy^3+(18*z*y*dz+27*y)*dy^2+(8*z^2*dz^2+44*z*dz+28)*dy+16*z*y^2*x*dz^3+8*y^2*x*dz^2,

(-4*y*dy-6*z*dz-9)*dx^4+(-28*z*dz-26)*dy^2*dx^2+(32*z^2*y*dz^4+144*z*y*dz^3+96*y*dz^2)*dx+4*y*dy^5+(2*z*dz+19)*dy^4+(32*z^2*x*dz^4+144*z*x*dz^3+96*x*dz^2)*dy,

(-4*y^2*dy-6*z*y*dz-9*y)*dx^3+(8*y^2*dy^3+(-24*z*y*dz+36*y)*dy^2+(-12*z^2*dz^2-60*z*dz+27)*dy)*dx-12*y*x*dy^4+(-6*z*x*dz-45*x)*dy^3+32*z^2*y^2*dz^4+
96*z*y^2*dz^3+24*y^2*dz^2,

(8*y^2*dy^2+(4*z*y*dz+30*y)*dy-12*z^2*dz^2-24*z*dz+9)*dx^2+8*y^2*dy^4+(28*z*y*dz+66*y)*dy^3+(12*z^2*dz^2+96*z*dz+99)*dy^2-32*z^2*y*x*dz^4-96*z*y*x*dz^3-24*y*x*dz^2,

(12*y^2*dy+18*z*y*dz+27*y)*dx^3+(8*y^2*dy^3+(56*z*y*dz+60*y)*dy^2+(68*z^2*dz^2+244*z*dz+87)*dy)*dx+4*y*x*dy^4+(2*z*x*dz+15*x)*dy^3+
32*z^2*x^2*dz^4+96*z*x^2*dz^3+24*x^2*dz^2,

(-12*y^2*dy^2+(-10*z*y*dz-47*y)*dy+12*z^2*dz^2+18*z*dz-18)*dx^3+((-52*z*y*dz-14*y)*dy^3+(-24*z^2*dz^2-180*z*dz-36)*dy^2)*dx-12*y*x*dy^5+(-6*z*x*dz-57*x)*dy^4-
64*z^3*y*dz^5-448*z^2*y*dz^4-624*z*y*dz^3-96*y*dz^2,

(12*y^2*dy+18*z*y*dz+27*y)*dx^4+((52*z*y*dz+14*y)*dy^2+(80*z^2*dz^2+264*z*dz+48)*dy)*dx^2-12*y^2*dy^5+(-38*z*y*dz-121*y)*dy^4+(-16*z^2*dz^2-168*z*dz-240)*dy^3-
64*z^3*x*dz^5-448*z^2*x*dz^4-624*z*x*dz^3-96*x*dz^2,

(12*y^2*dy+18*z*y*dz+27*y)*dx^5+(-12*y^2*dy^3+(42*z*y*dz-57*y)*dy^2+(92*z^2*dz^2+272*z*dz-17)*dy)*dx^3+(-12*y^2*dy^5+(-90*z*y*dz-135*y)*dy^4+
(-40*z^2*dz^2-400*z*dz-290)*dy^3)*dx-12*y*x*dy^6+(-6*z*x*dz-69*x)*dy^5+128*z^4*dz^6+1600*z^3*dz^5+5280*z^2*dz^4+4560*z*dz^3+480*dz^2]

F :
y*x^3+y^3*x+z^2
A :
-1
[(6*y^2*dy+z*y*dz+10*y)*dx-6*y*x*dy^2+(-3*z*x*dz-12*x)*dy+4*y^4*dz^2,
(6*y^2*dy^2+(-3*z*y*dz+18*y)*dy-2*z^2*dz^2-7*z*dz+6)*dx-6*y*x*dy^3+(-3*z*x*dz-18*x)*dy^2-8*z*y^3*dz^3-12*y^3*dz^2,

(-2*y^2*dy-3*z*y*dz-6*y)*dx^2-2*y^2*dy^3+(-9*z*y*dz-18*y)*dy^2+(-4*z^2*dz^2-26*z*dz-24)*dy-8*z*y^2*x*dz^3-12*y^2*x*dz^2,

(-2*y^2*dy-3*z*y*dz-6*y)*dx^3+(4*y^2*dy^3+(-12*z*y*dz+12*y)*dy^2+(-6*z^2*dz^2-36*z*dz)*dy)*dx-6*y*x*dy^4+(-3*z*x*dz-24*x)*dy^3+16*z^2*y^2*dz^4+72*z*y^2*dz^3+48*y^2*dz^2,

(-2*y^2*dy^2+(-z*y*dz-8*y)*dy+3*z^2*dz^2+9*z*dz)*dx^2-2*y^2*dy^4+(-7*z*y*dz-20*y)*dy^3+(-3*z^2*dz^2-27*z*dz-36)*dy^2+8*z^2*y*x*dz^4+36*z*y*x*dz^3+24*y*x*dz^2,

(6*y^2*dy+9*z*y*dz+18*y)*dx^3+(4*y^2*dy^3+(28*z*y*dz+44*y)*dy^2+(34*z^2*dz^2+156*z*dz+96)*dy)*dx+2*y*x*dy^4+(z*x*dz+8*x)*dy^3+16*z^2*x^2*dz^4+72*z*x^2*dz^3+48*x^2*dz^2,

(6*y^2*dy^2+(5*z*y*dz+26*y)*dy-6*z^2*dz^2-15*z*dz+6)*dx^3+((26*z*y*dz+20*y)*dy^3+(12*z^2*dz^2+102*z*dz+60)*dy^2)*dx+6*y*x*dy^5+
(3*z*x*dz+30*x)*dy^4+32*z^3*y*dz^5+288*z^2*y*dz^4+600*z*y*dz^3+240*y*dz^2,

(-6*y^2*dy-9*z*y*dz-18*y)*dx^4+((-26*z*y*dz-20*y)*dy^2+(-40*z^2*dz^2-172*z*dz-80)*dy)*dx^2+6*y^2*dy^5+(19*z*y*dz+70*y)*dy^4+
(8*z^2*dz^2+92*z*dz+160)*dy^3+32*z^3*x*dz^5+288*z^2*x*dz^4+600*z*x*dz^3+240*x*dz^2,

(-6*y^2*dy-9*z*y*dz-18*y)*dx^5+(6*y^2*dy^3+(-21*z*y*dz+18*y)*dy^2+(-46*z^2*dz^2-182*z*dz-48)*dy)*dx^3+(6*y^2*dy^5+(45*z*y*dz+90*y)*dy^4+
(20*z^2*dz^2+220*z*dz+240)*dy^3)*dx+6*y*x*dy^6+(3*z*x*dz+36*x)*dy^5-64*z^4*dz^6-960*z^3*dz^5-4080*z^2*dz^4-5280*z*dz^3-1440*dz^2]

F :
x^3+y^5+z^2
A :
0
[-dz,-dy,-dx]

F :
x^3+y^5+z^2
A :
-1/30
[-1000000*z^2*dx^6+(288000*y^3*dy^3+2916000*y^2*dy^2+(3600000*z^2*y*dz^2-1140000*z*y*dz+6708000*y)*dy+2250000*z^3*dz^3+1125000*z^2*dz^2-3205000*z*dz+2717000)*dx^3-
62208*y*dy^6+(-194400*z*dz-152928)*dy^5+(2430000*z^2*y*dz^4+2646000*z*y*dz^3-111600*y*dz^2)*dy+3796875*z^4*dz^6+24806250*z^3*dz^5+37749375*z^2*dz^4+
11557125*z*dz^3-238725*dz^2]

F :
x^3+y^5+z^2
A :
-7/30
[(36000*y^3*dy^2+(-60000*z*y^2*dz+228000*y^2)*dy+75000*z^2*y*dz^2-25000*z*y*dz+257000*y)*dx^3-7776*y*dy^5+(-25920*z*dz-20736)*dy^4+
(-81000*z*y^2*dz^3-19800*y^2*dz^2)*dy-253125*z^3*y*dz^5-911250*z^2*y*dz^4-646875*z*y*dz^3-58725*y*dz^2,

-1000000*z^2*dx^6+(288000*y^3*dy^3+2628000*y^2*dy^2+(3600000*z^2*y*dz^2+1020000*z*y*dz+5484000*y)*dy+2250000*z^3*dz^3+1125000*z^2*dz^2-5545000*z*dz+1601000)*dx^3-
62208*y*dy^6+(-194400*z*dz-215136)*dy^5+(4860000*z^2*y*dz^4+8964000*z*y*dz^3+655200*y*dz^2)*dy+3796875*z^4*dz^6+36956250*z^3*dz^5+92019375*z^2*dz^4+
53785125*z*dz^3+1885275*dz^2]

F :
x^3+y^5+z^2
A :
-11/30
[100000*z^2*dx^5+(-21600*y^3*dy^3-205200*y^2*dy^2+(-405000*z^2*y*dz^2-171000*z*y*dz-426600*y)*dy+247500*z^2*dz^2+355500*z*dz-153900)*dx^2-7776*x*dy^5+
759375*z^3*x*dz^5+2733750*z^2*x*dz^4+1184625*z*x*dz^3-18225*x*dz^2,

1000000*z^2*dx^6+(-288000*y^3*dy^3-2916000*y^2*dy^2+(-3600000*z^2*y*dz^2-60000*z*y*dz-6108000*y)*dy-2250000*z^3*dz^3-10125000*z^2*dz^2-7895000*z*dz-3017000)*dx^3+
62208*y*dy^6+(194400*z*dz+360288)*dy^5+(9720000*z^2*y*dz^4+19224000*z*y*dz^3+2145600*y*dz^2)*dy-3796875*z^4*dz^6-14681250*z^3*dz^5+47300625*z^2*dz^4+
93157875*z*dz^3+7843725*dz^2]

F :
x^3+y^5+z^2
A :
-13/30
[(4000*y^3*dy-5000*z*y^2*dz+11000*y^2)*dx^3-864*y*dy^4+(-3240*z*dz-2664)*dy^3+1800*y^3*dz^2*dy+16875*z^2*y^2*dz^4+25875*z*y^2*dz^3+6525*y^2*dz^2,

(-36000*y^3*dy^2+(60000*z*y^2*dz-192000*y^2)*dy-75000*z^2*y*dz^2-95000*z*y*dz-233000*y)*dx^3+7776*y*dy^5+(25920*z*dz+28512)*dy^4+
(202500*z*y^2*dz^3+125100*y^2*dz^2)*dy+253125*z^3*y*dz^5+1518750*z^2*y*dz^4+2145375*z*y*dz^3+475425*y*dz^2,

-1000000*z^2*dx^6+(288000*y^3*dy^3+2340000*y^2*dy^2+(3600000*z^2*y*dz^2+3180000*z*y*dz+4836000*y)*dy+2250000*z^3*dz^3+1125000*z^2*dz^2-10045000*z*dz-451000)*dx^3-
62208*y*dy^6+(-194400*z*dz-277344)*dy^5+(7290000*z^2*y*dz^4+21114000*z*y*dz^3+5698800*y*dz^2)*dy+3796875*z^4*dz^6+49106250*z^3*dz^5+173019375*z^2*dz^4+
165511125*z*dz^3+21775275*dz^2]

F :
x^3+y^5+z^2
A :
-17/30
[(-400*y^3*dy^2+(1000*z*y^2*dz-2200*y^2)*dy-2500*z^2*y*dz^2-2500*z*y*dz-2300*y)*dx^2-144*x*dy^4-5625*z^2*y*x*dz^4-10125*z*y*x*dz^3-675*y*x*dz^2,

(36000*y^3*dy^2+(-60000*z*y^2*dz+228000*y^2)*dy+75000*z^2*y*dz^2-125000*z*y*dz+157000*y)*dx^3-7776*y*dy^5+(-25920*z*dz-46656)*dy^4+
(324000*z*y^2*dz^3+223200*y^2*dz^2)*dy-253125*z^3*y*dz^5-911250*z^2*y*dz^4+973125*z*y*dz^3+913275*y*dz^2,

-100000*z^2*dx^5+(21600*y^3*dy^3+183600*y^2*dy^2+(405000*z^2*y*dz^2+387000*z*y*dz+361800*y)*dy-382500*z^2*dz^2-724500*z*dz+83700)*dx^2+
7776*x*dy^5-759375*z^3*x*dz^5-3948750*z^2*x*dz^4-3371625*z*x*dz^3-127575*x*dz^2,

1000000*z^2*dx^6+(-288000*y^3*dy^3-2628000*y^2*dy^2+(-3600000*z^2*y*dz^2-2220000*z*y*dz-4884000*y)*dy-2250000*z^3*dz^3-10125000*z^2*dz^2-
9155000*z*dz-3701000)*dx^3+62208*y*dy^6+(194400*z*dz+422496)*dy^5+(7290000*z^2*y*dz^4+22626000*z*y*dz^3+7210800*y*dz^2)*dy-3796875*z^4*dz^6-
26831250*z^3*dz^5-2919375*z^2*dz^4+97099875*z*dz^3+30109725*dz^2]

F :
x^3+y^5+z^2
A :
-19/30
[1000*y^3*dx^3-216*y*dy^3+(-1080*z*dz-936)*dy^2-3375*z*y^3*dz^3-1125*y^3*dz^2,

(-4000*y^3*dy+5000*z*y^2*dz-7000*y^2)*dx^3+864*y*dy^4+(3240*z*dz+3528)*dy^3-7200*y^3*dz^2*dy-16875*z^2*y^2*dz^4-52875*z*y^2*dz^3-38925*y^2*dz^2,
(-36000*y^3*dy^2+(60000*z*y^2*dz-156000*y^2)*dy-75000*z^2*y*dz^2-215000*z*y*dz-281000*y)*dx^3+7776*y*dy^5+(25920*z*dz+36288)*dy^4+
(324000*z*y^2*dz^3+424800*y^2*dz^2)*dy+253125*z^3*y*dz^5+2126250*z^2*y*dz^4+4615875*z*y*dz^3+2123325*y*dz^2,

-1000000*z^2*dx^6+(288000*y^3*dy^3+2052000*y^2*dy^2+(3600000*z^2*y*dz^2+5340000*z*y*dz+4764000*y)*dy+2250000*z^3*dz^3+1125000*z^2*dz^2-16705000*z*dz-5167000)*dx^3-
62208*y*dy^6+(-194400*z*dz-339552)*dy^5+(9720000*z^2*y*dz^4+39096000*z*y*dz^3+22017600*y*dz^2)*dy+3796875*z^4*dz^6+61256250*z^3*dz^5+280749375*z^2*dz^4+
387559125*z*dz^3+108031275*dz^2]

F :
x^3+y^5+z^2
A :
-23/30
[(200*y^3*dy-500*z*y^2*dz+300*y^2)*dx^2+72*x*dy^3-1125*z*y^2*x*dz^3-675*y^2*x*dz^2,

(4000*y^3*dy-5000*z*y^2*dz+11000*y^2)*dx^3-864*y*dy^4+(-3240*z*dz-5544)*dy^3-7200*y^3*dz^2*dy+16875*z^2*y^2*dz^4+48375*z*y^2*dz^3-15975*y^2*dz^2,

(400*y^3*dy^2+(-1000*z*y^2*dz+1800*y^2)*dy+2500*z^2*y*dz^2+4500*z*y*dz+2700*y)*dx^2+144*x*dy^4+5625*z^2*y*x*dz^4+16875*z*y*x*dz^3+4725*y*x*dz^2,

(36000*y^3*dy^2+(-60000*z*y^2*dz+192000*y^2)*dy+75000*z^2*y*dz^2-5000*z*y*dz+133000*y)*dx^3-7776*y*dy^5+(-25920*z*dz-54432)*dy^4+
(202500*z*y^2*dz^3+279900*y^2*dz^2)*dy-253125*z^3*y*dz^5-1518750*z^2*y*dz^4-930375*z*y*dz^3+901575*y*dz^2,

100000*z^2*dx^5+(-21600*y^3*dy^3-162000*y^2*dy^2+(-405000*z^2*y*dz^2-603000*z*y*dz-340200*y)*dy+517500*z^2*dz^2+1309500*z*dz+137700)*dx^2-
7776*x*dy^5+759375*z^3*x*dz^5+5163750*z^2*x*dz^4+7016625*z*x*dz^3+1148175*x*dz^2,

1000000*z^2*dx^6+(-288000*y^3*dy^3-2340000*y^2*dy^2+(-3600000*z^2*y*dz^2-4380000*z*y*dz-4236000*y)*dy-2250000*z^3*dz^3-10125000*z^2*dz^2-
8255000*z*dz-3449000)*dx^3+62208*y*dy^6+(194400*z*dz+484704)*dy^5+(4860000*z^2*y*dz^4+20196000*z*y*dz^3+11887200*y*dz^2)*dy-3796875*z^4*dz^6-
38981250*z^3*dz^5-79869375*z^2*dz^4+21823875*z*dz^3+44329725*dz^2]

F :
x^3+y^5+z^2
A :
-29/30
[-100*y^3*dx^2-36*x*dy^2-225*y^3*x*dz^2,

1000*y^3*dx^3-216*y*dy^3+(-1080*z*dz-1656)*dy^2-3375*z*y^3*dz^3-5625*y^3*dz^2,

(200*y^3*dy-500*z*y^2*dz+100*y^2)*dx^2+72*x*dy^3-1125*z*y^2*x*dz^3-1575*y^2*x*dz^2,

(-4000*y^3*dy+5000*z*y^2*dz-7000*y^2)*dx^3+864*y*dy^4+(3240*z*dz+6408)*dy^3+1800*y^3*dz^2*dy-16875*z^2*y
^2*dz^4-75375*z*y^2*dz^3-43425*y^2*dz^2,

(400*y^3*dy^2+(-1000*z*y^2*dz+1400*y^2)*dy+2500*z^2*y*dz^2+6500*z*y*dz+3900*y)*dx^2+144*x*dy^4+5625*z^2*y*x*dz^4+23625*z*y*x*dz^3+14175*y*x*dz^2,

(-36000*y^3*dy^2+(60000*z*y^2*dz-156000*y^2)*dy-75000*z^2*y*dz^2-115000*z*y*dz-181000*y)*dx^3+7776*y*dy^5+(25920*z*dz+62208)*dy^4+
(-81000*z*y^2*dz^3-142200*y^2*dz^2)*dy+253125*z^3*y*dz^5+2126250*z^2*y*dz^4+3805875*z*y*dz^3+989325*y*dz^2,

100000*z^2*dx^5+(-21600*y^3*dy^3-140400*y^2*dy^2+(-405000*z^2*y*dz^2-819000*z*y*dz-361800*y)*dy+652500*z^2*dz^2+2110500*z*dz+639900)*dx^2-
7776*x*dy^5+759375*z^3*x*dz^5+6378750*z^2*x*dz^4+12119625*z*x*dz^3+4209975*x*dz^2,

-1000000*z^2*dx^6+(288000*y^3*dy^3+2052000*y^2*dy^2+(3600000*z^2*y*dz^2+6540000*z*y*dz+4164000*y)*dy+2250000*z^3*dz^3+10125000*z^2*dz^2+5195000*z*dz+533000)*dx^3-
62208*y*dy^6+(-194400*z*dz-546912)*dy^5+(-2430000*z^2*y*dz^4-11934000*z*y*dz^3-9176400*y*dz^2)*dy+3796875*z^4*dz^6+51131250*z^3*dz^5+183549375*z^2*dz^4+
173494125*z*dz^3+17536275*dz^2]
```

We have not yet tried of doing a local version of this section with applying our algorithms in the previous sections. It is a next problem as well as proving the conjecture mathematically.

Bibliography

[1] Assi, A., Castro, F., Granger, M. : The standard fan of an analytic $\mathcal{D}$-module, Journal of Pure and Applied Algebra, 164, 3-21, (2001)

[2] Bernstein, I.N. : Modules over a ring of differential operators, Functional Analysis and Its Applications, 5, 89-101, (1971)

[3] Bernstein, I.N. : The analytic continuation of generalized functions with respect to a parameter, Functional Analysis and Its Applications, 6, 273-285, (1972)

[4] Castro, F. : Théoréme de division pur les opérateurs différentials et calcul des multiplicités, PhD thesis, Universite Paris VII, (1984)

[5] Castro, F. : Calculs effectifs pour les idéaux d'opérateurs différentiels, Travaux en Cours 24, 1-19, (1987)

[6] Castro, F., Granger, M. : Explicit calculations in rings of differential operators, Séminaires et Congrès 8, 89-128, (2004)

[7] Gebauer, R., Möller, H.M., On an installation of Buchberger's algorithm, Journal of Symbolic Computation, 3, 275-286, (1989)

[8] Giovini, A., Mora, T., Nielse, G., Robbiano, L., Traverso, C., : "One sugar cube, please" OR Selection strategies in the Buchberger algorithm, Proceeding of ISSAC '91, 49-54, (1991)

[9] Granger, M., Oaku, T. : Mimimal filtered free resolutions and division algorithms for analytic D-modules, Journal of Pure and Applied Algebra, 191, 1-2, 157-180, (2004)

[10] Granger, M., Oaku, T., Takayama, N. : Tangent cone algorithm for homogenized differential operators, Journal of Symbolic Computation, 39, 417-431, (2005)

[11] Greuel, G.-M., Pfister, G. : Advances and improvements in the theory of standard bases and syzygies, Archiv der Mathematik, 66, 163-176, (1996)

[12] Hironaka, H., Urabe, T. : *Analytic space*, Asakura, (in Japanese), (1981)

[13] Kashiwara, M. : B-functions and holonomic systems – Rationality of roots of b-functions, Inventiones Mathematicae, 38 (1976/77), no.1, 33-53

[14] Kashiwara, M. : *D-modules and Microlocal Calculus*, Translations of Mathematical Monographs, 217, AMS, (2003)

[15] Kobayashi, H., Furukawa, A., Sasaki, T. : Gröbner basis of ideal of convergent power series, Sūurikaisekikenkyūsho Kōkyūroku, 581, 7-25, (1986)

[16] Mora, T. : An algorithm to compute the equations of tangent cones, EUROCAM 82, Springer Lecture Notes in Computer Science, 144, 158-165, (1982)

[17] Nakayama, H. : An Algorithm Computing the Local b Funtion by an Approximate Division Algorithm in $\widehat{\mathcal{D}}$, `mathRA/0606437`

[18] Nakayama, H., Sekiguchi, J. : Determination of b-functions of polynomials defining Saito free divisors related with simple curve singularities of types E_6, E_7, E_8, Preprint

[19] Noro, M. : An Efficient Modular Algorithm for Computing the Global b-function , Mathematical Software(Beijing 2002), 147-157, (2002)

[20] Oaku, T., Shimoyama, T. : A Gröbner basis Method for Modules over Rings of Differential Operators, Journal of Symbolic Computation, 18, 223-248, (1994)

[21] Oaku, T. : An algorithm of computing b-function, Duke Mathematical Journal, 87, no.1, 115-132, (1997)

[22] Oaku, T. : Algorithm for the b-function and D-modules asscociated with a polynomial, Journal of Pure and Applied Algebra, 117/118, 495-518, (1997)

[23] Oaku, T. : *Gröbner bases and systems of linear differential equations – An intoduction to computational algebraic analysis*, Sophia University lecture note series, (in Japanese), (1994)

[24] Oaku, T. : *Computatinal algebra and D-modules*, Asakura, (in Japanese), (2002)

[25] Oaku, T. , Takayama, N. : Algorithms for D-modules - restriction, tensor product, localization, and local cohomology groups, Journal of Pure and Applied Algebra, 156, 267-308, (2001)

[26] Saito, M., Sturmfels, B., Takayama, N. : *Groebner Deformations of Hypergeometric Differential Equations*, Springer, (2000)

[27] Sekiguchi, J. : A classification of weighted homogeneours Saito free divisors in three dimensional space, Preprint

[28] Nakayama, H., Kuwahara, K. : Interactive and Graphical User Interfaces for Computer Algebra Systems (Software), `http://www.math.kobe-u.ac.jp/~nakayama/bg.html`

[29] Noro, M. et al : Risa/Asir, `http://www.math.kobe-u.ac.jp/Asir`

[30] Takayama,N. : Kan — A system for doing algebraic analysis by computer, `http://www.math.kobe-u.ac.jp/KAN`

[31] OpenXM, `http://www.math.kobe-u.ac.jp/OpenXM`