Computing Hypergeometric Solutions of Linear Recurrence Equations

Thomas Cluzeau*
LACO, University of Limoges
123 avenue Albert Thomas
87060 Limoges cedex, France
thomas.cluzeau@unilim.fr

Mark van Hoeij†
Department of mathematics
Florida State University
Tallahassee, FL 32306, USA
hoeij@math.fsu.edu

June 15, 2005

Abstract

We describe a complete algorithm to compute the hypergeometric solutions of linear recurrence relations with rational function coefficients. We use the notion of finite singularities and avoid computations in splitting fields. An implementation is available in Maple 9.

Introduction

Let C be a field of characteristic zero and $\overline{C}$ its algebraic closure. In many examples, $C = \mathbb{Q}$, but C can also be a number field and may also have transcendental degree > 0 over $\mathbb{Q}$, as long as the standard algorithms (such as zero-test in C, factoring in $C[x]$, etc.) are available.

A *linear difference operator*

$$L = a_n\,\tau^n + a_{n-1}\,\tau^{n-1} + \cdots + a_0\,\tau^0, \tag{1}$$

with $a_i = a_i(x) \in \overline{C}(x)$ is an operator that acts on functions $u = u(x)$ in the following way:

$$L(u) = a_n(x)\,u(x+n) + a_{n-1}(x)\,u(x+n-1) + \cdots + a_0(x)\,u(x).$$

The set of all such operators is denoted by $\overline{C}(x)[\tau]$. This is a non-commutative ring; multiplication is given by composition. We will always assume that the *leading coefficient* a_n and the *trailing coefficient* a_0 are both non-zero. In this case, n is called the *order* of L. The *field of definition of* L is the smallest field $C' \subseteq \overline{C}$ such that $L \in C'(x)[\tau]$.

The linear difference operator L in (1) corresponds to the *linear recurrence equation (or relation)*

$$L(u) = 0, \quad \text{i.e.,} \quad a_n(x)\,u(x+n) + a_{n-1}(x)\,u(x+n-1) + \cdots + a_0(x)\,u(x) = 0. \tag{2}$$

*work initiated while being a member of Laboratoire STIX, École polytechnique, 91128 Palaiseau cedex, France

†Supported by NSF grant 0098034

A *solution of* L is a function u for which $L(u) = 0$. A *hypergeometric solution* is a solution u for which $r := u(x+1)/u(x)$ is a rational function, *i.e.*, $r \in \overline{C}(x)$. Computing a hypergeometric solution u of L is equivalent to computing a first order right hand factor $\tau - r$ of L in $\overline{C}(x)[\tau]$.

An algorithm for computing hypergeometric solutions of linear recurrence relations with polynomial coefficients was first given by Petkovšek in [Pe92]. This algorithm is very useful and has interesting applications. Unfortunately, in practice it is limited to the case where a_0 and a_n do not have complicated roots because the algorithm computes the splitting field of $a_0\, a_n$ and this can easily run out of control. Even when this splitting field is not too complicated, the algorithm is still not optimal because it searches through more combinations than necessary (see [Ho99, Example 1.5] and also [We01, Section 6.3]). To address these problems, the concept of *finite singularities* was introduced in [Ho99]. This notion and the associated Fuchs' relations allow to reduce the number of combinations to check and to mimic Beke's algorithm to find exponential solutions of linear differential equations (see [Be94] or [CH04]). For $n < 4$, [Ho99, Theorems 2 and 3] show how to avoid splitting fields. In this paper we extend this to order $\geqslant 4$. We also give a number of additional improvements. We detail the steps of the algorithm and we show how they can be done efficiently. The complete HypSols algorithm is described in Section 9 and uses a similar approach as Algorithm ExpSols in [CH04] which computes the exponential solutions of linear differential equations. We will show in Section 10 how modular computations can be used to improve several steps in the algorithm. Finally, we will build the necessary tools to construct an analogue algorithm for q-difference equations when q is not a root of unity. Note that a q-analogue of Petkovšek's algorithm is given in [APP98].

To summarize, the main contributions of the paper are:

- More details on how to implement the algorithm in [Ho99] efficiently.
- How to avoid splitting fields for arbitrary order (in [Ho99] this was done for order $\leqslant 3$).
- Modular improvements in Section 10.
- Introduction of local types and Fuchs' relations for q-difference equations in Section 12, so that an algorithm along the same lines can be given for the q-difference case as well.

The paper is organized as follows. Section 1 recalls the vocabulary concerning hypergeometric terms and their local types. Section 2 shows how the symmetric product is used in the algorithm. Sections 3 and 4 deal with the finite singularities: we detail how candidate local types at these singularities can be computed efficiently. In Section 5 the same problem concerning the point at infinity is addressed. Section 6 discusses issues about field extensions. In Section 7 we give Algorithm ESols to compute the "easy" solutions and in Section 8 we describe Algorithm HSol that computes a "hard" hypergeometric solution, addressing the field problem. The complete Algorithm HypSols is given in Section 9 which also lists a number of additional improvements. Section 10 details how modular computations can improve the algorithm. Section 11 gives some timings. Finally, in the last section, we introduce the notions needed to construct an analogue algorithm for q-difference equations when q is not a root of unity.

1 Hypergeometric terms

Let C be a field of characteristic zero and $\overline{C}$ be its algebraic closure. In [PS97, Section 6.2], the authors prove the existence and uniqueness (up to difference isomorphisms) of a universal extension

$\mathcal{V}$ for difference equations with coefficients in $\overline{C}(x)$. Consequently, given a difference operator L, we can view the solutions of $L(y) = 0$ as elements of this universal extension $\mathcal{V}$. The dimension of the solution space (the kernel of $L : \mathcal{V} \to \mathcal{V}$) is n.

One can view $\mathcal{V}$ as a subring of the ring $\mathcal{S}$ which is defined as follows

$$\mathcal{S} = \{[f] \mid f \in \overline{C}^{\mathbb{N}}\},$$

where $[f]$ is the equivalence class of all $\tilde{f} \in \overline{C}^{\mathbb{N}}$ for which $f - \tilde{f}$ has finite support. If f is a function that is defined on all but finitely many elements of $\mathbb{N}$, then its image $[f] \in \mathcal{S}$ is well defined. This way $\overline{C}(x)$ can be viewed as a subring of $\mathcal{S}$.

Definition 1. *A non-zero expression $u(x) \in \mathcal{V}$ is called a* hypergeometric term *if it satisfies $u(x+1)/u(x) \in \overline{C}(x)$. The rational function $r(x) = u(x+1)/u(x) \in \overline{C}(x)$ is then called the* certificate *of $u(x)$. It is the analogue of the* logarithmic derivative *from the differential case.*

Let $u(x)$ be a hypergeometric term, and view $u(x)$ as an element of $\mathcal{S}$. If $C = \mathbb{C}$ then we can use Gamma functions to represent $u(x)$ by a function, defined for sufficiently large integers, as follows:

$$c^x \, R(x) \prod_{i=1}^{m} \Gamma(x - \alpha_i)^{e_i},$$

where $c \in \mathbb{C}^*$, $R(x) \in \mathbb{C}(x)$ and $\forall\, i \in \{1, \ldots, m\}$, $\alpha_i \in \mathbb{C}$ and $e_i \in \mathbb{Z}$.

In this paper, we do not need to distinguish a hypergeometric term $u(x)$ from a constant times $u(x)$. Thus we can represent $u(x)$ by its certificate, which explains the following definition.

Definition 2. *Let C be a field of characteristic zero. A hypergeometric term $u(x)$ is said to be* defined over C *if $u(x+1)/u(x) \in C(x)$.*

Definition 3. *Let $u_1(x)$, $u_2(x)$ be hypergeometric terms, and $r_1(x) = u_1(x+1)/u_1(x)$, $r_2(x) = u_2(x+1)/u_2(x)$ be their certificates. Then $u_1(x)$ and $u_2(x)$ are said to be of* the same type *(or* similar*) if $u_1(x)/u_2(x) \in \overline{C}(x)$, or equivalently, if $r_1(x)/r_2(x)$ is the certificate of a rational function.*

Remark 1. *The following property (see [PWZ96, Proposition 5.6.2]) shows that the notion of type is very natural:* u_1, u_2 are of the same type $\Leftrightarrow$ $u_1 + u_2$ is either a hypergeometric term or zero.

In fact, if $u_1, \ldots, u_k$ are hypergeometric terms of distinct types then they are $\overline{C}(x)$-linearly independent. Let $u(x)$ be a hypergeometric term defined over C and let $r(x) \in C(x)$ be its certificate. We can factor $r(x)$ over C as follows. Note that in general C will not be algebraically closed, so the factors need not be linear.

$$r(x) = c\, \phi_1(x)^{e_1} \cdots \phi_m(x)^{e_m}, \tag{3}$$

with $c \in C^*$, $e_i \in \mathbb{Z}$, and $\phi_i(x) = x^{\deg(\phi_i)} + \phi_{i,1} x^{\deg(\phi_i)-1} + \cdots + \phi_{i,\deg(\phi_i)} x^0$, where $\phi_{i,j} \in C$, and $\phi_i \in C[x]$ is monic and irreducible over C. Because $r(x)$ is a rational function, the multiplicities e_i can be negative.

Denote $t := 1/x$. Now $r(x) = r(1/t)$ can be written as

$$c\, t^v \, (1 + d\, t + \mathcal{O}(t^2)), \tag{4}$$

where $c \in C^*$, $v \in \mathbb{Z}$ and $d \in C$. Equations (3) and (4) are related in the following way: the number c is the same, and furthermore:

$$v = -\sum_{i=1}^{m} e_i \deg(\phi_i) \qquad \text{and} \qquad d = \sum_{i=1}^{m} e_i \, \phi_{i,1}.$$

If we factor $r(x)$ over $\overline{C}$, so $r(x) = c \prod_i (x-\alpha_i)^{e_i}$, then we can write $v = -\sum_i e_i$ and $d = -\sum_i e_i \, \alpha_i$.

Definition 4. *With the notations above, the* local type *of the hypergeometric term $u(x)$ at $x = \infty$ is defined as*

$$g_\infty(u(x)) := (c, v, d + \mathbb{Z}) \in C^* \times \mathbb{Z} \times C/\mathbb{Z}.$$

If $g_\infty(u_1) = (c_1, v_1, d_1 + \mathbb{Z})$ and $g_\infty(u_2) = (c_2, v_2, d_2 + \mathbb{Z})$, then $g_\infty(u_1 \, u_2) = (c_1 \, c_2, v_1 + v_2, d_1 + d_2 + \mathbb{Z})$.

Example 1. *Let*

$$u(x) = 7^x \frac{\Gamma(x + \frac{1}{3})\Gamma(x + \frac{6}{5})^3}{\Gamma(x - \frac{2}{3})\Gamma(x - \frac{4}{5})} \tag{5}$$

be a hypergeometric term defined over $\mathbb{Q}$. Then

$$r(x) = \frac{u(x+1)}{u(x)} = 7 \left(x + \frac{1}{3}\right)^1 \left(x - \frac{2}{3}\right)^{-1} \left(x + \frac{6}{5}\right)^3 \left(x - \frac{4}{5}\right)^{-1},$$

so that

$$c = 7, \quad v = -(1 \cdot 1 - 1 \cdot 1 + 3 \cdot 1 - 1 \cdot 1) = -2,$$

and

$$d = 1 \cdot \frac{1}{3} - 1 \cdot \left(-\frac{2}{3}\right) + 3 \cdot \frac{6}{5} - 1 \cdot \left(-\frac{4}{5}\right) = \frac{27}{5}.$$

Consequently,

$$g_\infty(u(x)) = \left(7, -2, \frac{27}{5} + \mathbb{Z}\right).$$

Entries of $g_\infty(u(x))$ correspond to terms in the expansion of $r(x)$ at $x = \infty$ (at $t = 0$):

$$r(x) = r\left(\frac{1}{t}\right) = 7\, t^{-2} \left(1 + \frac{27}{5}\, t + \mathcal{O}(t^2)\right).$$

Definition 5. *Let $u(x)$ be a hypergeometric term defined over a field C of characteristic zero and let $r(x) = u(x+1)/u(x) \in C(x)$ be as in Equation (3). Let $q \in \overline{C}$ and let $p = q + \mathbb{Z}$ be the image of q in $\overline{C}/\mathbb{Z}$. Then the* local type *or* valuation growth *of $u(x)$ at the "point" p is defined as:*

$$g_p(u(x)) = \sum_i e_i \text{ where the sum is taken over all } i \text{ for which } \phi_i \text{ has a root in } p.$$

That ϕ_i has a root in p is equivalent to saying that $\phi_i(q + l) = 0$ for some $l \in \mathbb{Z}$.

Example 2. *Let $u(x)$ be as in Example 1.*
For $p = 2/3 + \mathbb{Z}$, we have $g_p(u(x)) = 1 - 1 = 0$ and for $p = 4/5 + \mathbb{Z}$, $g_p(u(x)) = 3 - 1 = 2$.
If p is any other element of $\overline{\mathbb{Q}}/\mathbb{Z}$, then $g_p(u(x)) = 0$.
So for this example we have calculated the local types of $u(x)$ for all $p \in \overline{\mathbb{Q}}/\mathbb{Z} \cup \{\infty\}$. We can verify that the so-called Fuchs' relations *(see [Ho99, Equation 3]) hold:*

$$v + \sum_{p \in \overline{\mathbb{Q}}/\mathbb{Z}} g_p(u(x)) = 0 \qquad \text{and} \qquad d + \sum_{p \in \overline{\mathbb{Q}}/\mathbb{Z}} g_p(u(x))\, p \equiv 0 \bmod \mathbb{Z}. \tag{6}$$

The first one is $-2 + 0 + 2 = 0$, and the second is $\frac{27}{5} + \left(0 \cdot \frac{2}{3} + 2 \cdot \frac{4}{5}\right) \equiv 0 \bmod \mathbb{Z}$.

The following statement is part of [Ho99, Theorem 1].

Theorem 1. *Let $u_1(x)$ and $u_2(x)$ be hypergeometric terms. Then $u_1(x)$ and $u_2(x)$ are of the same type if and only if they have the same local type at every $p \in \overline{C}/\mathbb{Z} \cup \{\infty\}$.*

This theorem is central to the algorithm. The way the algorithm uses it is as follows: from the definition of *type*, if we know the type of $u(x)$, then we know $u(x)$ up to some unknown factor in $\overline{C}(x)$. The theorem says that we know the type of $u(x)$ if and only if we know the local type of $u(x)$ at all "points" $p \in \overline{C}/\mathbb{Z} \cup \{\infty\}$. Hence, in order to find a hypergeometric solution $u(x)$ of a linear recurrence equation, we first have to find the type of $u(x)$, and in order to do that we first have to find the local type of $u(x)$ at each $p \in \overline{C}/\mathbb{Z} \cup \{\infty\}$. In Section 4 we compute for $p \in \overline{C}/\mathbb{Z}$ a finite set $\overline{g}_p(L)$ of *candidate local types of L at p*. This means that for every hypergeometric solution u of L, the local type $g_p(u)$ of u at p is an element of $\overline{g}_p(L)$. In Section 5 we do the same for $p = \infty$.

2 The symmetric product

Definition 6. *Let L_1, $L_2 \in \overline{C}(x)[\tau]$. The* symmetric product *$L_1 \circledS L_2$ of L_1 and L_2 is defined as the monic operator $L \in \overline{C}(x)[\tau]$ of smallest order such that $L(u_1\, u_2) = 0$ for all u_1, $u_2 \in \mathcal{V}$ with $L_1(u_1) = 0$ and $L_2(u_2) = 0$.*

In this paper, we only use this when the order of L_2 is 1, and give a formula for this case. Let $L = \sum_{i=0}^{n} a_i \tau^i \in \overline{C}(x)[\tau]$ be a linear difference operator with $a_i \in \overline{C}(x)$ and let $r \in \overline{C}(x)$. Then

$$L \circledS (\tau - r) = \frac{1}{b_n} \sum_{i=0}^{n} b_i \tau^i \quad \text{where} \quad b_n = a_n \quad \text{and} \quad \forall\, i \in \{0, \ldots, n-1\},\ b_i(x) = a_i(x) \prod_{j=i}^{n-1} r(x+j).$$

We made the result monic (dividing by the leading coefficient b_n) in order to make $L \circledS (\tau - r)$ uniquely defined. However, in the implementation, uniqueness is not necessary here, and we multiply $L \circledS (\tau - r)$ by a polynomial in order to avoid computing with fractions.

One way that the symmetric product is used in the algorithm is as follows: suppose we have chosen a *candidate*, that is, for each point p we have chosen a candidate local type of L at p (*i.e.*, an element of $\overline{g}_p(L)$). If this candidate does not satisfy Fuchs' relations given in Equation (6) then the candidate does not correspond to the type of any hypergeometric term $u(x)$ (see [Ho99, Theorem 1]), and thus it can be discarded. If it does satisfy these relations, then we have a *candidate type*. Then we apply a few tests (see Remark 3 in Section 4) in order to quickly discard those types for which it is easy to decide that L has no hypergeometric solution of that type. If the candidate

type passes these tests, then we want to compute a basis $u_1, \dots, u_m$ of all hypergeometric solutions of L of that type (if such solutions exist). We construct a function $r(x) \in \overline{C}(x)$ that represents this candidate type. This means that a non-zero solution u of $\tau - r$ has this type. So $u_1/u, \dots, u_m/u$ must be in $\overline{C}(x)$. Now $1/u$ is a solution of $\tau - 1/r$ and so from Definition 6 it follows that

$$\frac{u_1}{u}, \dots, \frac{u_m}{u} \text{ is a basis of rational solutions of } L_{\circledS}(\tau - \frac{1}{r}).$$

So finding $u_1, \dots, u_m$ reduces to computing *rational solutions*. However, at no additional computational effort (see Example 5 and Remark 3 in Section 4) we can choose r in such a way that $u_1/u, \dots, u_m/u$ must be in $\overline{C}[x]$. This way we only need to compute *polynomial solutions*.

3 What does it mean to be a finite singularity?

Definition 7 (Definition 8 in [Ho99])**.** *Let $L = a_n\tau^n + \cdots + a_0\tau^0 \in \overline{C}(x)[\tau]$ with $a_n \neq 0$ and $a_0 \neq 0$. After multiplying L on the left by a suitable element of $\overline{C}(x)$, we may assume that the coefficients a_i are in $\overline{C}[x]$ and $\gcd(a_0, \dots, a_n) = 1$. Then $q \in \overline{C}$ is called a* problem point *of L if q is a root of the polynomial $a_0(x)\, a_n(x-n)$.*

We want a notion of singularities that is invariant under the shift operation $p \mapsto p + 1$. Now p is invariant under the shift if $p = \infty$ (Section 5 discusses the singularity $p = \infty$) but not invariant if $p \in \overline{C}$. Thus, we will deviate from other definitions in the literature (our *problem points* are called *singularities* in [No29]) and define finite singularities as follows:

Definition 8. *$p \in \overline{C}/\mathbb{Z}$ is called a* finite singularity *of L if L has a problem point in p.*

Example 3. *Let*

$$L = x\left(x - \frac{1}{3}\right)\left(x + \frac{1}{4}\right)\tau^2 - \tau + x(x-3).$$

The order is $n = 2$. The roots of the trailing coefficient give the problem points 0 and 3, and the roots of the leading coefficient give the problem points $0 + 2$, $1/3 + 2$, $-1/4 + 2$. If $u(x) \in \mathcal{V}$ is a solution of L and if $q \in \overline{\mathbb{Q}}$ then

$$q\left(q - \frac{1}{3}\right)\left(q + \frac{1}{4}\right)u(q+2) - u(q+1) + q(q-3)u(q) = 0,$$

whenever this is defined. Solving for $u(q+2)$ gives

$$u(q+2) = -\frac{-u(q+1) + q(q-3)u(q)}{q(q-\frac{1}{3})(q+\frac{1}{4})}, \quad \text{in general} \quad u(q+n) = -\sum_{i=0}^{n-1} \frac{a_i(q)}{a_n(q)} u(q+i), \tag{7}$$

and solving for $u(q)$ leads to

$$u(q) = -\frac{q(q-\frac{1}{3})(q+\frac{1}{4})u(q+2) - u(q+1)}{q(q-3)}, \quad \text{in general} \quad u(q) = -\sum_{i=1}^{n} \frac{a_i(q)}{a_0(q)} u(q+i). \tag{8}$$

Formula (7) can be used to compute $u(q+2)$ from previous u-values if $q+2$ is not a problem point whereas Formula (8) can be used to compute $u(q)$ from next u-values if q is not a problem point.

Consequently the problem points of L are $\{0, 1/3, -1/4\} + 2 = \{2, 7/3, 7/4\}$ and $\{0, 3\}$. This means that the finite singularities are

$$\left\{2 + \mathbb{Z}, \frac{7}{3} + \mathbb{Z}, \frac{7}{4} + \mathbb{Z}, 0 + \mathbb{Z}, 3 + \mathbb{Z}\right\} = \left\{\mathbb{Z}, \frac{1}{3} + \mathbb{Z}, \frac{3}{4} + \mathbb{Z}\right\},$$

so there are three finite singularities.

Consider $p = \mathbb{Z}$. If for example the values of $u(-5), u(-4)$ are given, then we can use the relations (7,8) to calculate $u(q)$ for $q \in \mathbb{Z}$ with $q < 2$. But $u(2)$ can not be computed because taking $q = 0$ in equation (7) leads to a division by zero. If $u(8), u(9)$ were given, then $u(q)$ could be computed for all $q \in \mathbb{Z}$ for which $q > 3$. We can not reach $u(3)$ from $u(8), u(9)$ because taking $q = 3$ in equation (8) results again in a division by zero.

The point $p = 1/3 + \mathbb{Z}$ is a finite singularity. If $u(5 + 1/3), u(6 + 1/3)$ were given, then $u(q)$ can be determined with equations (7,8) for all $q \in p$. However, if $u(-5 + 1/3), u(-4 + 1/3)$ were given then not all $u(q)$, $q \in p$ can be determined. So p is singular, but one only notices its singular behavior when going from the left to the right (with Equation (7)), and not when going from the right to the left (with Equation (8)). The point $p = 3/4 + \mathbb{Z}$ is a finite singularity for the same reason.

Any other element of $\overline{\mathbb{Q}}/\mathbb{Z}$ is not a singularity. For example, $p = 1/2 + \mathbb{Z}$ is not a finite singularity of L, because whenever we know any two consecutive values of u on p then the two formulas (7,8) allow us to go freely to the left and right without ever bumping into a division by zero; there are no problem points in $p = 1/2 + \mathbb{Z}$.

In general, if p is a finite singularity, if n consecutive values $u(q+1), u(q+2), \ldots, u(q+n)$ are given, and if $q+1, \ldots, q+n$ are *on the left of the problem points*, then $u(q+l)$ can be computed for all $l \leqslant n$ (with $l \in \mathbb{Z}$) but not always for $l > n$ because that would require passing through problem points. This is the essence of the concept of *left solutions of L at p* introduced in [Ho99]: they are solutions defined on a left half line. Similarly, if $q+1, \ldots, q+n$ were *on the right of the problem points*, then $u(q+l)$ could be determined for positive integers l, which means we have *right solutions of L at p*, they are defined on a right half line.

The set of all left (resp. right) solutions at p is denoted by $V_{p,l}(L)$ (resp. $V_{p,r}(L)$). For a precise definition of these sets, see [Ho99, Section 2]. Both are n-dimensional vector spaces, where n is the order of L (we always assume that a_n and a_0 are non-zero). So at a finite singularity p we distinguish two kinds of solutions, left and right solutions, which are separated by the problem points. If p is non-singular then $V_{p,l}(L)$ and $V_{p,r}(L)$ are not separated and so they can be identified.

Deforming the singularity p is a method to bypass the problem points, in order to go from the left to the right solutions, as well as from the right to the left solutions. The deformed operator L_ϵ is obtained by substituting $x \mapsto x + \epsilon$ in L. This deformation moves the singularity p to $p - \epsilon$, and turns p into a non-singular point: dividing by zero is now replaced by dividing by ϵ. Here ϵ is a new constant; deforming increases our field of constants from $\overline{C}$ to $\overline{C}(\epsilon)$. Because of this change of constants, elements of the two solution spaces (left and right) of L at p can not be identified in a unique way with solutions of the deformed operator L_ϵ, but we will see in Section 9.1 (see also [Ho99, Section 4.2]) that it is still possible to define useful maps between these solution spaces.

The main information we gather by going from the left to the right and from the right to the left in the deformed operator L_ϵ is the multiplicity of the factor ϵ that will appear in the denominator (or in the numerator). In the next section, the set $\overline{g}_p(L)$ is constructed from this data.

4 How to compute $\overline{g}_p(L)$ for finite p.

Let C be a field of characteristic zero and let $L \in C(x)[\tau]$ be a linear difference operator. Let $p \in \overline{C}/\mathbb{Z}$. In [Ho99] a procedure was given to calculate a finite set denoted by $\overline{g}_p(L)$ that has the property:

$$u(x) \text{ hypergeometric solution of } L \implies g_p(u(x)) \in \overline{g}_p(L). \tag{9}$$

Note: the converse does not hold in general.
We shall show how this can be done in practice in an efficient way. If $p = q + \mathbb{Z} \in \overline{C}/\mathbb{Z}$ where $q \in \overline{C}$ then computing with the singularity p of L is equivalent to computing with the singularity $0 + \mathbb{Z}$ of the operator L_q where L_q is the operator obtained from L by substituting $x + q$ for x. So without loss of generality we may assume that the singularity p to be considered is $p = \mathbb{Z}$. Note though that (see the discussion in Section 6), given $L \in C(x)[\tau]$ for some field C of characteristic zero, if $q \notin C$ then we need to temporarily (as long as we are computing with this singularity) replace the field C by $C' = C(q)$ because $L_q \in C'(x)[\tau]$.

Now we will assume that $L \in C'[x][\tau]$ has order n and that $p = \mathbb{Z}$ is the singularity to be studied. Then the problem points corresponding to p are the integer roots of $a_0(x)\, a_n(x-n)$. We take the largest possible $p_l \in p$ such that for all integers $q \leqslant p_l$, the two consecutive sequences $u(p_l - n), \ldots, u(p_l - 1)$ and $u(q-n), \ldots, u(q-1)$ can be computed from each other via the two equations (7,8) without bumping into divisions by zero. Then we can identify every *left solution* of L at p with a vector $(u(p_l - n), \ldots, u(p_l - 1))$ with entries in $\overline{C}$, because given any n consecutive values of such a left solution, we can determine $(u(p_l - n), \ldots, u(p_l - 1))$ by repeated use of Equation (7), and furthermore, given $(u(p_l - n), \ldots, u(p_l - 1))$ we can determine $u(q)$ for all integers $q < p_l$ by repeated use of Equation (8). Similarly, we can take the smallest integer $p_r \in p$ such that we can go back and forth between $(u(p_r + 1), \ldots, u(p_r + n))$ and $(u(q+1), \ldots, u(q+n))$ for all $q \geqslant p_r$. This way each *right solution* of L at p can be identified with a vector $(u(p_r + 1), \ldots, u(p_r + n))$ with entries in $\overline{C}$.

Remark 2. *The condition back* and *forth is important in order for such smallest $p_r \in p$ to exist. For example if $L = \tau^2 - x^2$ and $p = \mathbb{Z}$ then starting with $(u(p_r + 1), u(p_r + 2))$ and say $p_r < 0$ we can go arbitrarily far to the right, but say we go to $(u(1), u(2))$, then going from there to $(u(p_r + 1), u(p_r + 2))$ without dividing by zero is not possible when $p_r < 0$. So for $L = \tau^2 - x^2$ we can not take $p_r < 0$. We see that $p_r = 0$. For $L = \tau^2 - x^2$ and $p = \mathbb{Z}$ we have $p_l = 2$. Notice that $p_l - n$ is always smaller than $p_r + 1$, but that $\{p_l - n, \ldots, p_l - 1\}$ and $\{p_r + 1, \ldots, p_r + n\}$ are not necessarily disjoint.*

Example 4. *Back to Example 3, recall that concerning the finite singularity $p = \mathbb{Z}$ we have seen that we can not obtain $u(2)$ from previous u-values nor can we obtain $u(0)$ or $u(3)$ from next u-values. One finds $p_l = 2$ and $p_r = 3$.*

Since L_ϵ, the operator obtained by substituting $x + \epsilon$ for x in L, has no singularity at $p = \mathbb{Z}$, its solution space at p, $V_p(L_\epsilon) := \{\tilde{u} : p \to \overline{C}(\epsilon) \mid L_\epsilon(\tilde{u}) = 0\}$ is an n-dimensional $\overline{C}(\epsilon)$-vector space. We now specify two bases $\tilde{u}_1, \ldots, \tilde{u}_n$ and $\tilde{v}_1, \ldots, \tilde{v}_n$ of $V_p(L_\epsilon)$ in the following way:

$$\tilde{u}_i(p_l - j) = \delta_{i,j} \quad \text{and} \quad \tilde{v}_i(p_r + j) = \delta_{i,j} \quad \text{for} \quad i, j \in \{1, \ldots, n\},$$

where $\delta_{i,j}$ is 1 if $i = j$ and 0 otherwise. If we substitute $\epsilon = 0$ in $\tilde{u}_1, \ldots, \tilde{u}_n$ then one may have divisions by 0, but these do not occur at $q < p_l$. This way we obtain a basis $\{u_1, \ldots, u_n\} =$

$\{\tilde{u}_1, \ldots, \tilde{u}_n\}_{\epsilon=0}$ of left solutions of L at p. Likewise, $\{\tilde{v}_1, \ldots, \tilde{v}_n\}_{\epsilon=0}$ is defined at all $q > p_r$, and this results in a basis of right solutions of L at p.

Definition 9. *Let $a \in \overline{C}(\epsilon)$. The* valuation $v_\epsilon(a)$ of a at $\epsilon = 0$ *is the element of $\mathbb{Z} \cup \{\infty\}$ defined as follows: if $a \neq 0$ then $v_\epsilon(a)$ is the largest integer $m \in \mathbb{Z}$ such that $a/\epsilon^m \in \overline{C}[[\epsilon]]$, and $v_\epsilon(0) = \infty$.*

So

$$v_\epsilon(a) > 0 \iff a_{\epsilon=0} = 0 \quad \text{and} \quad v_\epsilon(a) < 0 \iff a \text{ has a pole at } \epsilon = 0.$$

Now, for $p = \mathbb{Z}$, we want to compute a set $\overline{g}_p(L)$ satisfying the specification of (9). First, for $q \in \{p_l - n, \ldots, p_r + n\}$, we consider the following integers:

$$B^l(q) = \min_{1 \leqslant i \leqslant n} v_\epsilon(\tilde{u}_i(q)) \text{ and } B^r(q) = \min_{1 \leqslant i \leqslant n} v_\epsilon(\tilde{v}_i(q)). \tag{10}$$

These integers are very useful since they also lead to bounds for the denominators of rational solutions in the same way as in [Ho98].

Definition 10. *The* minimum valuation growth $g_{p,r}(L)$ *and the* maximum valuation growth $g_{p,l}(L)$ *of L at p are defined as*

$$g_{p,r}(L) = \min\{B^l(q) \,|\, q > p_r\}, \quad g_{p,l}(L) = -\min\{B^r(q) \,|\, q < p_l\}, \tag{11}$$

and the set of valuation growths*, or* candidate local types*, denoted $\overline{g}_p(L)$, is then*

$$\overline{g}_p(L) := \{m \in \mathbb{Z} \,|\, g_{p,r}(L) \leqslant m \leqslant g_{p,l}(L)\}.$$

Note on difference modules: If $p \in \overline{C}/Z$ and if two operators L_1, L_2 define isomorphic difference modules (for a definition see [PS97]) then $\overline{g}_p(L_1) = \overline{g}_p(L_2)$, and thus $\overline{g}_p$ can be defined for difference modules. This $\overline{g}_p$ behaves in a predictable way with respect to tensor products, direct sums, etc. Moreover, if $\overline{g}_p(L_1) = \{0\}$ then there exists an operator L_2 defining an isomorphic difference module, for which p is not a singularity. We do not detail this because difference modules are not needed for this paper.

If $u = u(x)$ is a hypergeometric solution of L, then its *local type* $g_p(u)$ at p, also called its *valuation growth*, must be an element of this set $\overline{g}_p(L)$, see [Ho99]. If $\overline{g}_p(L)$ has only one element, then that leaves only one choice for $g_p(u)$. In that case p is called a *semi-apparent* singularity. If 0 is the only element of $\overline{g}_p(L)$ then p is called an *apparent* singularity[1]. The singularities that are most significant in the algorithm are the ones where there is more than one choice for $g_p(u)$, *i.e.*, the singularities p where $\overline{g}_p(L)$ has more than one element. Such singularities are called *essential* singularities.

If L has a basis of rational solutions, then all singularities are apparent. More generally, if L has n linearly independent hypergeometric solutions, then $g_{p,r}(L)$ is precisely the smallest $g_p(u)$ of all hypergeometric solutions u of L, and $g_{p,l}(L)$ is the largest $g_p(u)$. If L has fewer than n linearly independent hypergeometric solutions then $g_{p,r}(L)$ and $g_{p,l}(L)$ are just bounds for $g_p(u)$ and are often no longer sharp. If $L = L_1 L_2$ then $\overline{g}_p(L_2) \subseteq \overline{g}_p(L)$ always holds, but it is easy to find examples where $\overline{g}_p(L_2) \neq \overline{g}_p(L)$, even when L does not have more hypergeometric solutions than

[1] our terminology deviates slightly from [ABH05] where trailing (resp. leading) singularities are called apparent if the numbers $B^r(q)$ (resp. $B^l(q)$) are non-negative.

L_2. This already occurs when $L = (\tau - x)(\tau - x) = \tau^2 - (2x+1)\tau + x^2$. Fortunately, this does not imply that Algorithm Search in Section 7 will have to try many types in Steps **2a**–**2e** because Fuchs' relations (Equation (6)) are usually quite selective.

Now, in order to calculate the *minimum valuation growth* $g_{p,r}(L)$ from the formulas (10) and (11), we have to start from the $\tilde{u}_i$ (which take values 0 and 1 at $p_l - n, \ldots, p_l - 1$) and evaluate them at $p_r + 1, \ldots, p_r + n$. This computation starts with values at $p_l - n, \ldots, p_l - 1$, and then goes to the right using Equation (7).

Similarly, to compute the *maximum valuation growth* $g_{p,l}(L)$, we start with values 0 and 1 for $\tilde{v}_i$ at $p_r + 1, \ldots, p_r + n$, and then we proceed to the left (with Equation (8)) in order to evaluate $\tilde{v}_i$ at $p_l - n, \ldots, p_l - 1$.

Truncated power series:
Suppose we want to calculate $B^l(q)$ for $q \in \{p_l - n, \ldots, p_r + n\}$. We need to compute the $\tilde{u}_i(q)$ for $i \in \{1, \ldots, n\}$ and $q \in \{p_l, \ldots, p_r + n\}$. For efficiency reasons, we want to prevent computing in our new field of constants $C'(\epsilon)$. Recall that C' is the field extension of C given by the singularity $p \in \overline{C}/\mathbb{Z}$ that we shifted to $p = \mathbb{Z}$. Note that computing the sets $\overline{g}_p(L)$ using direct computations in $C'(\epsilon)$ hurts the running time of the algorithm: this is remarked in [We01, Section 6.3].
As we are merely interested in the ϵ-valuations of the $\tilde{u}_i(q)$, these $\tilde{u}_i(q) \in C'(\epsilon)$ can be replaced by truncated power series in ϵ, *i.e.*, power series of the form $\sum_{i<a} c_i \epsilon^i + \mathcal{O}(\epsilon^a)$ where the sum is finite and $a \in \mathbb{Z}$. This a is called *the accuracy* of the truncated power series. If $a_0(x), \ldots, a_n(x)$ are the coefficients of L then $a_0(x+\epsilon), \ldots, a_n(x+\epsilon)$ are the coefficients of L_ϵ. Let N_0 (resp. N_n) be the number of roots (counted with multiplicity) of $a_0(x)$ (resp. $a_n(x)$) in p. When we compute the next term with Equation (7), the accuracy decreases by the ϵ-valuation of $a_n(q+\epsilon)$, which is the same as the order of $a_n(x)$ at $x = q$. Combined, going from the left to the right, the accuracy decreases by N_n. We can recover the ϵ-valuation of an element of $C'(\epsilon)$ from a truncated power series when the accuracy is greater than the ϵ-valuation. We start computing with $\tilde{u}(p_l - n), \ldots, \tilde{u}(p_l - 1)$ with initial accuracy $a = N_0 + N_n + 1$. Going to the right with Equation (7), the accuracy has decreased by N_n when we reach $\tilde{u}_i(q)$ with $q > p_r$, so the remaining accuracy is $a - N_n$. Considering the fact that one can go back with Equation (8), it is not difficult to show that N_0 is an a priori upper bound for $g_{p,r}(L)$, hence the remaining accuracy $a - N_n = N_0 + 1$ suffices to determine $g_{p,r}(L)$. The same initial accuracy $N_0 + N_n + 1$ suffices to compute $g_{p,l}(L)$ as well.

Example 5. *Let* $L = (x-1)(x+1)\tau^2 - x(x^2+x-1)\tau + x^2(x-1)$ *and* $p = \mathbb{Z}$. *The problem points are* 1, 3 *and* 0, 1. *Here* 1, 3 *are the solutions of* $a_n(x-n) = (x-1-2)(x+1-2) = 0$, *and* 0, 1 *are the roots of* a_0. *Now* $p_l - n, \ldots, p_l - 1$ *(the starting points for the* $\tilde{u}_i$*) is* $-1, 0$ *because this is the furthest to the right for which the following is true: for any* q, $q+1 \in p$ *on the left of* $-1, 0$ *one can go back and forth between* q, $q+1$ *and* $-1, 0$ *without dividing by zero. Likewise,* $p_r + 1, \ldots, p_r + n$ *(the starting points for the* $\tilde{v}_i$*) will be* 2, 3.

Replacing x *by* $x + \epsilon$ *to determine* L_ϵ, *and solving* $\tilde{u}(x+2)$ *from* $L_\epsilon(\tilde{u}) = 0$ *like in Equation (7) we get*

$$\tilde{u}(x+2) = -\frac{-(x+\epsilon)\left((x+\epsilon)^2 + (x+\epsilon) - 1\right)\tilde{u}(x+1) + (x+\epsilon)^2(x+\epsilon-1)\tilde{u}(x)}{(x+\epsilon-1)(x+\epsilon+1)}.$$

$N_0 + N_n + 1 = 6$ *so we start with:* $\tilde{u}_1(-1) = 0 + \mathcal{O}(\epsilon^6)$, $\tilde{u}_1(0) = 1 + \mathcal{O}(\epsilon^6)$. *Likewise, take* $\tilde{u}_2(-1) = 1 + \mathcal{O}(\epsilon^6)$, $\tilde{u}_2(0) = 0 + \mathcal{O}(\epsilon^6)$. *Then substitute* $x = -1, 0, 1$ *in the above equation to find:*

$\tilde{u}_1(1) = -\frac{1}{2\epsilon} - \frac{1}{4} + \frac{7\epsilon}{8} - \frac{\epsilon^2}{16} - \frac{\epsilon^3}{32} - \frac{\epsilon^4}{64} + \mathcal{O}\left(\epsilon^5\right)$, $\tilde{u}_2(1) = -\frac{1}{\epsilon} + 2 - \epsilon + \mathcal{O}\left(\epsilon^5\right)$,
$\tilde{u}_1(2) = -\frac{1}{2} + \frac{\epsilon}{4} + \frac{\epsilon^2}{8} + \frac{9}{16}\epsilon^3 - \frac{23}{32}\epsilon^4 + \mathcal{O}\left(\epsilon^5\right)$, $\tilde{u}_2(2) = -1 + 3\,\epsilon - 3\,\epsilon^2 + 2\,\epsilon^3 - 2\,\epsilon^4 + \mathcal{O}\left(\epsilon^5\right)$,
$\tilde{u}_1(3) = -\frac{1}{4} - \frac{1}{4}\epsilon + \frac{3}{16}\epsilon^2 + \frac{11}{16}\epsilon^3 + \mathcal{O}\left(\epsilon^4\right)$, $\tilde{u}_2(3) = -\frac{1}{2} + \frac{7}{4}\epsilon - \frac{3}{8}\epsilon^2 - \frac{13}{16}\epsilon^3 + \mathcal{O}\left(\epsilon^4\right)$.

The accuracy, indicated by $\mathcal{O}(\epsilon^a)$, drops when we divide by ϵ. Now $B^l(q) = \min\{v_\epsilon(\tilde{u}_1(q)), v_\epsilon(\tilde{u}_2(q))\}$ and $g_{p,r}(L) = \min\{B^l(2), B^l(3)\}$ so we have $B^l(1) = -1$, $B^l(2) = 0$, $B^l(3) = 0$, and $g_{p,r}(L) = 0$. Now use $L_\epsilon(\tilde{v}) = 0$ to write $\tilde{v}(x)$ as a linear combination of $\tilde{v}(x+1)$ and $\tilde{v}(x+2)$. Then start with $\tilde{v}_1(2) = 1 + \mathcal{O}(\epsilon^6)$, $\tilde{v}_1(3) = 0 + \mathcal{O}(\epsilon^6)$ and $\tilde{v}_2(2) = 0 + \mathcal{O}(\epsilon^6)$, $\tilde{v}_2(3) = 1 + \mathcal{O}(\epsilon^6)$ and compute for $i = 1, 2$ the values of $\tilde{v}_i(x)$ at $x = 1, 0, -1$. At $x = 1$ we divide by $a_0(1+\epsilon)$ and the accuracy drops by 1. Then at $x = 0$ we divide by $a_0(0+\epsilon)$ and the accuracy drops by 2. So for $i = 1, 2$ we find $\tilde{v}_i(1)$ with accuracy 5, and find $\tilde{v}_i(0)$ and $\tilde{v}_i(-1)$ with accuracy 3 which is more than enough to determine $g_{p,l}(L)$. We find $B^r(1) = -1$, $B^r(0) = -1$, $B^r(-1) = -1$ and $g_{p,l}(L) = -\min\{B^r(0), B^r(-1)\} = 1$. Finally, we get

$$\overline{g}_p(L) = \{m \in \mathbb{Z} \,|\, g_{p,r}(L) \leqslant m \leqslant g_{p,l}(L)\} = \{0, 1\}.$$

Our algorithm for computing hypergeometric solutions will try two candidate types *(see Section 2) for this example. In one of these candidate types, we have the element 0 from $\overline{g}_p(L)$ and the number c from Section 1 (computed in Section 5) is 1. For this candidate type we need to find rational solutions of L. In [Ho98] the numbers $\max\{B^l(q), B^r(q)\}$ were used to give a denominator bound for rational solutions. We can do the same here. The number $\max\{B^l(q), B^r(q)\}$ is a lower bound for the valuation of a rational solution at $x = q$. This bound is -1 at $x = 1$ and zero elsewhere, thus, the rational solution must be of the form $(x-1)^{-1}$ times a polynomial. The degree bound for this polynomial is obtained using the information at $p = \infty$ computed in the next section. This degree bound is zero, and thus what remains to be done for this candidate type is to check if $(x-1)^{-1}$ is a solution, which it is.*

The next candidate type has the element 1 from $\overline{g}_p(L)$ and the number c is again 1. So we have to compute the rational solutions of $L_{\circledS}(\tau - 1/r)$ where $\tau - r$ should have the desired candidate type, which implies that $r = x\,\frac{\tau(f)}{f}$ for some rational function f. Now the $B^l(q)$ number for $L_{\circledS}(\tau - 1/r)$ is the $B^l(q)$ number for L plus the $B^l(q)$ number for $\tau - 1/r$. The same holds for $B^r(q)$. We can choose f in $r = x\,\frac{\tau(f)}{f}$ in such a way that $L_{\circledS}(\tau - 1/r)$ will have $\max\{B^l(q), B^r(q)\} = 0$ for all $q \in \overline{C}$. This way, the rational solutions of $L_{\circledS}(\tau - 1/r)$ must be polynomials. The r obtained this way is $r = x - 1$. The degree bound for the polynomial solutions is zero, thus, the algorithm checks if the solution $\Gamma(x-1)$ of $\tau - r$ is a solution of L, which it is.

Example 6. *Let us see briefly what we obtain for the operator given in Example 3. In Example 4, we found $p_l = 2$ and $p_r = 3$, so we start with values of $\tilde{u}_i$ at 0, 1 and go to the right. We find $B^l(2) = B^l(3) = B^l(4) = B^l(5) = -1$ so that $g_{p,r}(L) = -1$. Then we start with values of $\tilde{v}_i$ at 4, 5 and go the the left, and we find $B^r(3) = B^r(2) = B^r(1) = -1$, $B^r(0) = -2$. So $g_{p,l}(L) = -\min\{-1, -2\} = 2$ and finally, $\overline{g}_p(L) = \{-1, 0, 1, 2\}$.*

Modular computations:

We can further improve this by computing modulo prime numbers. In the Maple implementation, this is done as follows: take three "big" primes p_1, p_2 and p_3, consider the coefficients of the truncated power series as elements of $\overline{\mathbb{F}}_{p_i}$ (details concerning the reduction of elements in C' modulo p are included in [CH04, Sections 1.3 and 7.1]) and make the computations modulo each p_i. If these three modular computations lead to the same result, then return it and stop, otherwise take new primes.

Remark 3. *With these two tricks, reducing rational functions in ϵ to truncated power series, and reducing the elements of C' to a finite field, computing the sets $\overline{g}_p(L)$ for $p \in \overline{C}/\mathbb{Z}$ is quite fast and does generally not dominate the running time of our algorithm.*
Moreover, by storing for each p the numbers $B^l(q)$ and $B^r(q)$ that were computed to obtain $\overline{g}_p(L)$, we can compute r in such a way that all rational solutions of $L_{\circledS}(\tau - 1/r)$ are already polynomials. A degree bound for these polynomial solutions can be obtained from the $B^l(q), B^r(q)$ numbers (these numbers determine r) and the local information at $p = \infty$ computed in the next section. We compute this degree bound before *computing $L_{\circledS}(\tau - 1/r)$. This saves computation time, because if for example the degree bound is negative then we do not need to compute $L_{\circledS}(\tau - 1/r)$ (this is one of the tests we use to quickly discard candidate types that do not correspond to hypergeometric solutions, other tests are based on modular computations, see Section 10). If the degree bound is 0 then we take the sum of the coefficients of $L_{\circledS}(\tau - 1/r)$ and do a zero-test (first a fast probabilistic test, then if necessary a full test). If the degree-bound is positive but not high, then we proceed by computing and solving necessary linear equations for the coefficients of the polynomial solutions. Since we do not want to solve systems of equations with many variables, we should use a different method if the degree-bound is high (see [ABP95, Ba97, BCS05]).*

5 How to compute $\overline{g}_\infty(L)$.

As with finite points, given $L \in C(x)[\tau]$, one can exhibit a finite set $\overline{g}_\infty(L)$ such that:

$$u(x) \text{ hypergeometric solution of } L \implies g_\infty(u(x)) \in \overline{g}_\infty(L).$$

We detail here how we can compute this set with the classical techniques (see [Ad28, Bi30, No29]) used to compute the formal solutions at infinity of linear difference operators. Our approach is based on the following easy lemma which follows directly from the definition of the local type at infinity and the exposition in [Ad28] or [Bi30].

Lemma 1. *Let $L \in C[x][\tau]$. If L has a hypergeometric solution $u = u(x)$ with local type $(c, v, d+\mathbb{Z})$ at infinity, then L has a formal solution at infinity of the form $\Gamma(x)^{-v}\, c^x\, x^d\, F(1/x)$ where $F(1/x)$ is a formal power series in $1/x$.*

The possible couples $(c, v) \in \overline{C} \times \mathbb{Z}$ appearing in the formal solutions at infinity can be read from the so-called *Newton τ-polygon* (see [BD94], note that similar polygons already appeared in [Ad28, Bi30]). More precisely $-v$ is a (integer) slope of this polygon whereas c is a root of the *Newton τ-polynomial* (or characteristic equation) associated with the slope $-v$.

Definition 11. *Let $L = \sum_{i=0}^{n} a_i\, \tau^i \in C[x][\tau]$. For $(u, v) \in \mathbb{R}^2$, let $\mathcal{L}(u, v) := \{(x, y) \in \mathbb{R}^2 \,|\, x = u,\, y \geqslant v\}$. Let $\mathcal{M}(L)$ be the union of the $\mathcal{L}(i, -\deg(a_i, x))$ for $i \in \{0, \ldots, n\}$ with $a_i \neq 0$. The* Newton τ-polygon of L *is the convex hull of $\mathcal{M}(L)$.*

We give now an algorithm that computes only relevant (*i.e.*, integer) slopes of the Newton τ-polygon with their associated Newton τ-polynomial.

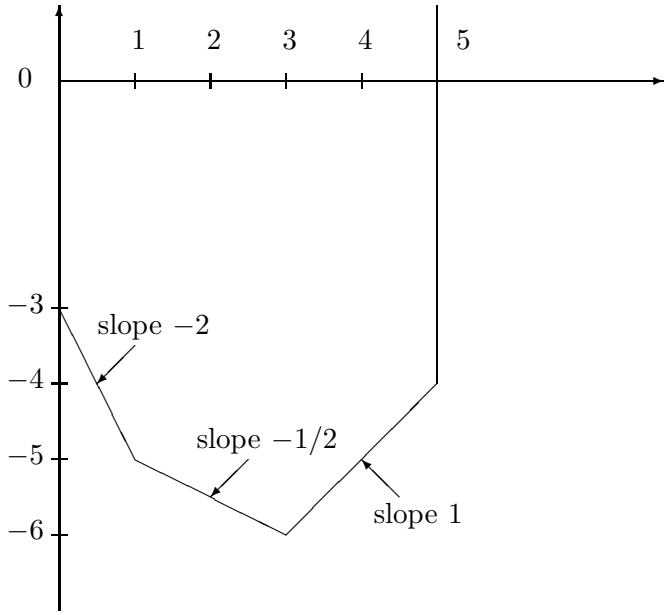
Algorithm τPolygon**:**
Input: $L := a_n\tau^n + \cdots + a_0\tau^0$ with $a_i \in C[x]$, a_0, $a_n \neq 0$.
Output: The integer slopes of the Newton τ-polygon and corresponding Newton τ-polynomials.

1. If $n = 0$, then return $\emptyset$ and stop.
2. For $i \in \{0, \ldots, n\}$, let $v_i := -\deg(a_i, x)$ (if $a_i = 0$ then $v_i = \infty$).
3. $s := \max((v_n - v_i)/(n-i), i = 0, \ldots, n-1)$.
4. $m := \min(i < n \,|\, (v_n - v_i)/(n-i) = s)$.
5. $R := \emptyset$.
6. If $s \in \mathbb{Z}$, then
  6a. $P := \sum_{i=0}^{n-m} \text{coefficient}(a_{n-i}, x^{s\,i-v_n})\, x^{n-m-i}$,
  6b. $R := \{[s, P]\}$.
7. Return $R \cup \tau\text{Polygon}(\ a_m\tau^m + \cdots + a_0\tau^0\ )$ and stop.

Example 7. *Let*

$$L := x^4\,\tau^5 + (5\,x^5 - 12\,x^3 - 3\,x)\,\tau^4 - (x^6 + x + 7)\,\tau^3 - (140\,x^3 + 1)\,\tau^2 + 10\,x^5\,\tau - 8\,x^3.$$

The Newton τ-polygon of L is:



The slopes are 1, $-1/2$, -2. *Algorithm* τPolygon *first finds the slope* $s = 1$ *and computes the associated Newton* τ*-polynomial* $P = x^2 + 5\,x - 1$. *Then, in the recursive call, considering only the terms of* L *of order* $\leqslant 3$ *(since* $m = 3$*), it finds* $s = -1/2$ *(with* $m = 1$*) which is not an integer. Thus, it continues with only the terms of* L *of order* $\leqslant 1$ *and finds the slope* $s = -2$ *(and* $m = 0$*): it computes the associated Newton* τ*-polynomial* $P = 10\,x - 8$ *and terminates. Consequently, Algorithm* τPolygon *returns:* $\{[1,\ x^2 + 5x - 1],\ [-2,\ 10x - 8]\}$.

The above algorithm works more generally for $L \in C((t))[\tau]$ where $t = 1/x$ if we replace $-\deg(a_i, x)$ by $v_t(a_i)$ where v_t denotes the t-adic valuation.

If $\tau - r$ is a right hand factor of L then $\tau - r/(c\,t^v) = \tau - (1 + d\,t + \mathcal{O}(t^2))$ is a right hand factor of $L_{c,v}$ which is defined as (note: $1/t^v$ is just x^v)

$$L_{\circledS}(\tau - 1/(c\,t^v)),$$

multiplied on the left by a polynomial in order to avoid fractions. Now write $\delta = \tau - 1$ so $\tau = \delta + 1$ and $\tau - (1 + d\,t + \mathcal{O}(t^2)) = \delta - d\,t + \mathcal{O}(t^2)$. Let

$$L^{\delta}_{c,v} := \text{substitute}(\tau = \delta + 1,\ L_{c,v}) \in C(c)[x][\delta].$$

The transformation we have made to obtain $L_{c,v}$ amounts to dividing the formal solution in Lemma 1 by the term $\Gamma(x)^{-v}\,c^x$ so that the first order right hand factor $\delta - d\,t + \mathcal{O}(t^2)$ we are searching for corresponds to a formal solution of the form $t^{-d}\,F(t)$ where $F(t)$ is a formal power series in t with non-zero constant term.

Definition 12. *Let $L \in C((t))[\delta]$ and e be a variable. Let $L_e := L(t^e)/t^e \in C[e]((t))$. Then, we define $v_1(L)$ as the t-adic valuation of L_e. The* indicial equation of L *is the element of $C[e]$ defined as the coefficient of $t^{v_1(L)}$ in L_e.*

This definition is the one used in [AB98]. In [BD94, Section 2.1] a slightly different *indicial equation* is defined from another polygon we can associate to a difference operator: the *Newton δ-polygon*. Our indicial equation can be computed in the same way.

If $f \in C((t))^*$ has t-adic valuation $v_t(f)$, then the t-adic valuation of $L(f)$ is $v_t(f) + v_1(L)$ if $v_t(f)$ is not a root of the indicial equation and is $> v_t(f) + v_1(L)$ otherwise. Consequently, $v_1(L)$ can be viewed as the amount the t-adic valuation changes if we apply L to a "generic element" of $C((t))$. It follows that if $L_1,\, L_2 \in C((t))[\delta]$, then $v_1(L_1\,L_2) = v_1(L_1) + v_1(L_2)$ so that v_1 is a valuation on $C((t))[\delta]$.

Note that $v_1(t^i\,\delta^j) = i + j$, and that if $L = \sum_{i,j} a_{i,j}\,t^i\,\delta^j$ then $v_1(L) = \min(i + j \,|\, a_{i,j} \neq 0)$. So the only coefficients that contribute to the indicial equation are the $a_{i,j}$ for which $i + j = v_1(L)$. Set $e_j = (-1)^j\,e\,(e+1)\,(e+2)\cdots(e+j-1) \in C[e]$ then $\delta^j(t^e)/t^e = e_j\,t^j + \mathcal{O}(t^{j+1})$ so the indicial equation of $L = \sum_{i,j} a_{i,j}\,t^i\,\delta^j$ is the sum of the $a_{i,j}\,e_j$ taken over all i, j for which $i + j = v_1(L)$. From [Ad28, Bi30] we know that:

Lemma 2. *If $L \in \overline{C}((t))[\delta]$ has a solution of the form $t^e\,F(t)$ where $F(t) \in \overline{C}[[t]]$ is a formal power series with non-zero constant term and $e \in \overline{C}$, then e is a root of the indicial equation of L.*

The roots of the indicial equation are called the *exponents* of L. The numbers d we are looking for are *negated exponents* $d = -e$. For the local type we only need d modulo $\mathbb{Z}$. So if some roots e_1, e_2 of the same indicial equation differ by an integer in Step **3a(v)** below, say $e_2 - e_1 \in \mathbb{Z}$, $e_2 - e_1 > 0$, then we skip e_2. We also work up to conjugation over $C(c)$, meaning that we consider only one root of each irreducible factor of the indicial equation over $C(c)$.

Algorithm LocalTypeAt∞**:**
Input: $L \in C(x)[\tau]$ and the field C.
Output: the set $\overline{g}_{\infty}(L)$, up to conjugation over C.

1. $\mathcal{A} := \emptyset$.
2. $\mathcal{P} := \tau$Polygon(L). If $\emptyset$, then return $\emptyset$.
3. For each $[s, P] \in \mathcal{P}$, set $v = -s$ and do:

3a. For each (up to conjugation over C) root c of P, do:
(i) $L_{c,v} := L_{\circledS}(\tau - x^v/c)$,
(ii) Clear denominators in $L_{c,v}$,
(iii) $L^{\delta}_{c,v} :=$ substitute$(\tau = \delta + 1,\ L_{c,v})$,
(iv) Compute the indicial equation of $L^{\delta}_{c,v}$,
(v) For each (mod $\mathbb{Z}$, up to conjugation over $C(c)$) root e of this indicial equation, do:
- Set $d := -e$ and add (c, v, d) to $\mathcal{A}$.

4. Return $\mathcal{A}$.

Remark 4. *For efficiency reasons, step* **3a** *is implemented slightly differently: take* $L_v := L_{\circledS}(\tau - x^v)$ *and let* $\tilde{L} = \sum a_{i,j} x^{-i} \delta^j = \sum a_{i,j} t^i \delta^j$ *be the operator obtained from* L_v *by substituting* $\tau = \delta + c$. *We do not actually calculate* $\tilde{L}$ *completely, rather, we compute only those terms* $a_{i,j} t^i \delta^j$ *for which* $i + j = v_1(\tilde{L})$. *Then, we sum the* $a_{i,j}\, c^j\, (-1)^j\, e\,(e+1)\cdots(e+j-1)$ *over those* i, j *to get the indicial equation. The factor* c^j *corrects for the fact that we did not include* c *in the computation of* L_v *for better efficiency (this helps in particular when* c *involves an algebraic extension). This way, the indicial equation generally takes very little CPU time to calculate.*

Example 8. *We continue with example 7. The output of Algorithm* τPolygon *was* $\{[1,\ x^2 + 5x - 1],\ [-2,\ 10x - 8]\}$. *The first root we consider is* $c = \alpha$ *where* $\alpha = \mathrm{RootOf}(x^2 + 5\,x - 1)$. *The indicial equation in Step* **3a(v)** *has degree one, and has as root* $e = -d = -5/29\,\alpha + 89/29$. *The second possible* c *is* $4/5$ *which leads to* $-d = 0$. *Consequently, we have:*

$$\overline{g}_{\infty}(L) = \{(\alpha,\ -1,\ 5/29\,\alpha - 89/29),\ (4/5,\ 2,\ 0)\}.$$

Definition 13. *The* roots-number *of* $(c, v, d + \mathbb{Z}) \in \overline{g}_{\infty}(L)$ *is the number of distinct roots of the indicial equation of* $L^{\delta}_{c,v}$ *in* $e + \mathbb{Z}$ *where* $e = -d$. *The* formal-sol-dimension *of* $(c, v, d + \mathbb{Z})$ *is the dimension of the formal solutions of* $L^{\delta}_{c,v}$ *in* $t^e\, \overline{C}((t))$.

The formal-sol-dimension is less than or equal to the roots-number. Each of these two numbers can be used (see Proposition 1 in Section 8) to bound the degree of a field extension of $C(c, d)$ needed to find hypergeometric solutions. In this paper we use the roots-number since it is less work to implement. However, instead one may also use the formal-sol-dimension which leads to a better bound. The same issue for the differential case is discussed in the comments on logarithms in [CH04, Section 7.1].

6 Representing the finite singularities

Let $L = a_n\,\tau^n + \cdots + a_0\,\tau^0$ where the a_i are polynomials in x with coefficients in a field C of characteristic zero. In our algorithm we will use algebraic extensions C' of C but not splitting fields. Hence, we can not work with all roots of $a_0(x)\,a_n(x - n)$ simultaneously. This restriction is necessary because the construction of a splitting field can be prohibitively costly. This does not imply that there are any roots of $a_0(x)\,a_n(x-n)$ that we can not compute with, we can compute with each root, but we should compute with only one root at a time. This also means that throughout the computation we always need to keep track of what the base field C is and what the current extension field C' is, in particular, when factoring a polynomial it is essential (for efficiency as well as correctness of the algorithm) to specify the correct field. The algorithm SingOver$C$ below is essentially trivial, but we include it to emphasize this field issue. First another definition:

Definition 14. *Let $f(x)$, $g(x) \in \overline{C}[x]$. If there exists an integer m such that $f(x) = g(x+m)$ then $f(x)$ and $g(x)$ are said* shift equivalent.

Algorithm SingOver$C$**:**
Input: $L \in C(x)[\tau]$ and the field C.
Output: the set of finite singularities of L over C, up to conjugation over C.

1. Sing := $\emptyset$.
2. Let $\mathcal{S}$ be the set of monic irreducible factors of $a_n(x)\, a_0(x)$ in $C[x]$.
3. Now take a subset $\mathcal{T} \subseteq \mathcal{S}$ such that precisely one polynomial has been chosen in each shift equivalence class.
4. For all $P \in \mathcal{T}$, do:
  4a. Take a root x_P of P (see comments below),
  4b. Sing := Sing $\cup \{[P, x_P]\}$.
5. Return Sing.

The output Sing is given as couples $[P, x_P]$ where $P \in C[x]$ is irreducible over C and $x_P = \mathrm{RootOf}(P, x)$, which denotes the image of x in $C' = C[x]/(P)$. So the set Sing represents all finite singularities "up to conjugation over C" (see [CH04, 5.1] for details). Note that no calculation takes place in Step **4a**. Instead, this step involves the construction of a data structure: one needs to construct a field C' generated by one root x_P of the polynomial P.

After the set Sing has been computed, our base field is still C in many places in the algorithm. An exception is of course when we compute with a singularity $p = x_P + \mathbb{Z}$ in which case the base field is $C' = C(x_P)$.

Given Sing, if we want to determine the singularities over a field extension of C then we take each $[P, x_P]$ in Sing, we factor P over the field extension, and replace $[P, x_P]$ by the set of all $[P', x_{P'}]$ for all irreducible factors P' of P.

7 Computing e-solutions

Definition 15. *Let C be a field of characteristic zero. Let $u(x)$ be a hypergeometric term and let $r(x) = u(x+1)/u(x) \in \overline{C}(x)$. The* field of definition of $u(x)$ over C *is the smallest field $C' \supseteq C$ for which $r(x) \in C'(x)$. The* cvd-field of $u(x)$ *is the field $C(c, d)$ where c, d are as in Equation (4). For $L \in C(x)[\tau]$, the* cvd-fields of L *are the fields $C(c, d)$ for $(c, v, d + \mathbb{Z}) \in \overline{g}_\infty(L)$.*

The cvd-field of a hypergeometric term u is a subfield of the field of definition of u.

Definition 16. *Let C be a field of characteristic zero and $L \in C(x)[\tau]$. An* easy hypergeometric solution*, or* e-solution *for short, is a hypergeometric solution of L defined over its cvd-field.*

Definition 17. *For $r \in \overline{C}(x)$, let* $\mathrm{hyp}(r) \in \mathcal{V}$ *denote a non-zero solution of $\tau - r$.*

Note that $\mathrm{hyp}(r)$ is not uniquely defined, but only defined up to a constant factor. However, that will not play a role in this paper.

Example 9. *Let $L \in \mathbb{Q}(x)[\tau]$ and assume that L has both $u_1 := \mathrm{hyp}(x^2+\sqrt{2})$ and $u_2 := \mathrm{hyp}(x+\sqrt{2})$ as hypergeometric solutions. By definition, u_1 is a hypergeometric solution of $\tau - (x^2 + \sqrt{2})$ so its local type at infinity is $(1, -2, \mathbb{Z})$ since $x^2 + \sqrt{2} = 1\, t^{-2}\ (1 + O(t^2))$ with $t = 1/x$, and so its cvd-field*

is $\mathbb{Q}$: it follows that this is not an e-solution. Now, $x+\sqrt{2} = 1\,t^{-1}\,(1+\sqrt{2}\,t^1+O(t^2))$ thus the local type at infinity of u_2 is $(1,-1,\sqrt{2}+\mathbb{Z})$ and its cvd-field is $\mathbb{Q}(\sqrt{2})$ so u_2 is an e-solution of L.

The following algorithm computes a basis of all e-solutions of $L \in C(x)[\tau]$ with a given local type $(c,v,d) \in \overline{g}_\infty(L)$.

Algorithm ESols**:**
Input: $L \in C(x)[\tau]$, the field C and (c,v,d).
Output: Sol, a basis of all e-solutions of L with local type $(c,v,d+\mathbb{Z})$ at ∞.
1. Let $\mathcal{S} :=$ SingOverC(L,C).
2. For $[P_i,x_i] \in \mathcal{S}$, compute $\overline{g}_{x_i}(L)$.
3. Let $C' := C(c,d)$ be the cvd-field.
4. For $[P_i,x_i] \in \mathcal{S}$, factor P_i over C': let $P_i := P_{i,1}\cdots P_{i,m_i}$.
5. Sol := Search(L, C', (c,v,d), $\{\overline{g}_{x_i}(L)\}_i$, $\{P_{i,j}\}_{i,j}$).
6. Return Sol.

We store the result from Steps **1** and **2** so that the computed information can be re-used later, for example, when ESols is called again with the same L but a new (c,v,d).

Algorithm ESols first computes the finite singularities $[P_1,x_1],\ldots,[P_s,x_s]$ over C. For each x_i it computes the set $\overline{g}_{x_i}(L)$ of candidate local types. It then factors the P_i over the cvd-field $C(c,d)$ and calls Algorithm Search below, which computes hypergeometric solutions defined over C'. By taking $C' := C(c,d)$ we find e-solutions (to search for other hypergeometric solutions, Algorithm HSol in the next section calls Search with a larger field). The factorization $P_{i,1}\cdots P_{i,m_i}$ of P_i over C' is part of the input of Algorithm Search below. We do not need to compute the set $\overline{g}_{x_{i,j}}(L)$ (where $x_{i,j}$ denotes a root of $P_{i,j}$) because $x_{i,j}$ is also a root of P_i and so $\overline{g}_{x_{i,j}}(L) = \overline{g}_{x_i}(L)$.

Algorithm Search**:**
Input: $L \in C(x)[\tau]$, the field C', $(c,v,d) \in \overline{g}_\infty(L)$, the $\overline{g}_{x_i}(L)$ and the $P_{i,j}$.
Output: Sol, a basis of all hypergeometric solutions of L defined over C' with local type $(c,v,d+\mathbb{Z})$ at ∞.
1. Let Sol $:= \emptyset$.
2. For each combination $(l_{1,1},\ldots,l_{1,m_1},\ldots,l_{s,m_s})$ with $l_{i,j} \in \overline{g}_{x_i}(L)$ that satisfies the two Fuchs' relations (Equation (6)), do:
2a. Let $\tilde{r} := c\prod_{i,j} P_{i,j}^{l_{i,j}}$,
2b. Compute $R \in C'(x)$ (see comments below) and let $r := \tilde{r}\,\frac{\tau(R)}{R}$,
2c. Compute $\tilde{L} := L_{\circledS}(\tau - 1/r)$,
2d. Compute a basis $Q_1,\ldots,Q_w$ of polynomial solutions in $C'[x]$ for $\tilde{L}$,
2e. If $w>0$, then add the hyp$(r)\,Q_i$ to Sol.
3. Return Sol.

In Section 4, for each singularity x_i, to produce $\overline{g}_{x_i}(L)$ we compute the numbers $B^l(q)$, $B^r(q)$ defined in Equation (10). From these numbers and r, it is easy to deduce the corresponding numbers for the operator $\tilde{L}$ in Step **2c**. Let $\tilde{B}^l(q)$, $\tilde{B}^r(q)$ denote those numbers. We choose R in Step **2b** in such a way that $\max\{\tilde{B}^l(q),\tilde{B}^r(q)\} = 0$ for all q. Then any rational solution of $\tilde{L}$ is a polynomial. From the numbers $l_{i,j}$, $B^l(q)$, $B^r(q)$ and d (here we need d instead of $d+\mathbb{Z}$), we compute a degree

bound for these polynomial solutions *before* computing $\tilde{r}$, R and $\tilde{L}$. As mentioned in Remark 3 we do this to save computation time whenever the degree bound is negative.

To use Algorithm Search in the recursive algorithm HSol in Section 8, we need to allow one extra input, namely an option called "just one". If this option is given, then we only need to return one (if it exists) hypergeometric solution over C' but not a totally arbitrary one: we return a $\mathrm{hyp}(r)\,Q$ with Q of minimal possible degree.

8 Computing a hard solution

Let C be a field of characteristic zero and $L \in C(x)[\tau]$. To find a basis of all hypergeometric solutions of L, we may have to search for hypergeometric solutions defined over algebraic extensions of the cvd-fields of L. We first bound the degrees of such extensions. As in the differential case (see [CH04, Section 6.1]), we define the notion *the minimal algebraic degree*:

Definition 18. *Let C be a field of characteristic zero and u be a hypergeometric term. Let C' be the field of definition of u over C (see Definition 15). Then the* algebraic degree *of u over C is $[C' : C]$. We say that u is of* minimal algebraic degree m over C *if $[C' : C] = m$ and there exist no hypergeometric term, of the same type as u, having smaller algebraic degree over C.*

Then, adapting the exposition in [CH04, Section 6.1] from the differential case, we can prove the following ([CH04, Lemma 6.3]):

Lemma 3. *If u is a hypergeometric solution of L defined over some algebraic extension C' of C, then its conjugates over C are also solutions of L.*
Given L and u, we can compute $u_1, \ldots, u_k$ of minimal algebraic degree over C that form a basis of all hypergeometric solutions of L of the same type as u.
If u is of minimal algebraic degree over C, then distinct conjugates of u over C are of distinct type and hence linearly independent.

This implies that a hypergeometric solution of L of minimal algebraic degree m corresponds to m linearly independent hypergeometric solutions of L. In Algorithm HypSols in Section 9, we only return hypergeometric solutions up to conjugation over C. Therefore to make it easier to count the number of linearly independent solutions represented by this output, we want HypSols to only return hypergeometric solutions of minimal algebraic degree.

Proposition 1. *Let u be a hypergeometric solution of L with local type $(c, v, d + \mathbb{Z})$ at infinity. Let $B_1(c, v, d)$ be the roots-number of $(c, v, d + \mathbb{Z})$ (see Definition 13). If u has minimal algebraic degree m over its cvd-field $C(c, d)$, then $m \leqslant B_1(c, v, d)$.*

Proof. From Lemma 3, the conjugates of u form m linearly independent hypergeometric solutions of L, all of which have local type $(c, n, d + \mathbb{Z})$ at infinity. Now (see Section 5), these hypergeometric solutions correspond to formal solutions at infinity of $L^{\delta}_{c,v}$ that can be written $t^{-d}\,F_i(t)$, $i \in \{1, \ldots, m\}$, where $F_i(t) \in \overline{C}[[t]]$. After Gaussian elimination on the vectors of coefficients of the $F_i(t)$, we obtain m formal solutions $t^{-d}\,\tilde{F}_i(t)$ of $L^{\delta}_{c,v}$ with distinct exponents. Hence, we can conclude $m \leqslant B_1(c, v, d)$ by Definition 13. □

The final Algorithm HypSols in Section 9 first computes the e-solutions and uses them to reduce the order. After this, L will no longer have e-solutions. Then, we want to compute (if it exists) a

hypergeometric solution, with local type $(c, v, d+\mathbb{Z})$ at infinity, defined over an algebraic extension of its cvd-field $C(c, d)$. Algorithm HSol ("hard" hypergeometric solution) below introduces algebraic extensions of this field and searches for a hypergeometric solution. We stop Algorithm HSol as soon as one solution is found, so that Algorithm HypSols can use this solution to reduce the order. Since Algorithm HSol is only called after the e-solutions have been computed, the finite singularities $\text{Sing} = \{[P_1, x_1], \ldots, [P_s, x_s]\}$ over C and the $\overline{g}_{x_i}(L)$ are already known. Algorithm HypSols calls Algorithm HSol with arguments: L, a cvd-field $C(c, d)$ of L, (c, v, d), a bound B, the set Sing, and the empty set for $\mathcal{F}$. So the field that is denoted as C in Algorithm HSol is initially a cvd-field $C(c, d)$, and so in Algorithm HSol we always have c, $d \in C$.

Algorithm HSol**:**
Input: $L \in C(x)[\tau]$, a field C, (c, v, d), $B \in \mathbb{N}$, a set Sing and a set $\mathcal{F}$.
Output: a hypergeometric solution of L with local type $(c, v, d+\mathbb{Z})$ at ∞ or $\emptyset$.
1. If $B = 0$, then return $\emptyset$ and stop.
2. If option "no extension", then $\mathcal{S} :=$ Sing, otherwise:
  2a. For all $[P_i, x_i] \in$ Sing, do:
    2a1. If $[P_i, x_i] \in \mathcal{F}$, then $P_{i,1} := P_i$, otherwise factor P_i over C, $P_i = P_{i,1} \cdots P_{i,m_i}$,
  2b. If Search(L, C, (c, v, d), $\{\overline{g}_{x_i}(L)\}_i$, $\{P_{i,j}\}_{i,j}$, "just one") $\neq \emptyset$, then return it and stop.
  2c. If $B = 1$, then return $\emptyset$ and stop.
  2d. Let $\mathcal{S}$ be the set of all $[P_{i,j}, x_{i,j}]$ and let $\overline{g}_{x_{i,j}}(L) := \overline{g}_{x_i}(L)$ for all i, j.
3. If $\mathcal{S} \setminus \mathcal{F}$ contains a $[P_i, x_i]$ with x_i an essential singularity and $[C(x_i) : C] > 1$, then:
  3a. If HSol(L,C,(c, v, d),B,$\mathcal{S}$, $\mathcal{F} \cup \{[P_i, x_i]\}$, "no extension") $\neq \emptyset$ then return it and stop.
  3b: If HSol(L, $C(x_i)$, (c, v, d), $\lfloor B \frac{\lfloor \deg(P_i)/2 \rfloor}{\deg(P_i)} \rfloor$, $\mathcal{S}$, $\mathcal{F}$) $\neq \emptyset$ then return it and stop.
4: Return $\emptyset$.

Algorithm specification: if there exists a hypergeometric solution u that satisfies the information encoded by $\mathcal{F}$ (meaning that none of the minimal polynomials in $\mathcal{F}$ becomes reducible over the field of definition of u) and that has minimal algebraic degree $\leqslant B$ over C, then HSol will return some hypergeometric solution of L (but not necessarily of degree $\leqslant B$ over C).

Proof. Suppose that a hypergeometric solution u exists, satisfies the information encoded by $\mathcal{F}$ and has degree $\leqslant B$ over C. Let C' be the field of definition of this u over C. Consider the $[P_i, x_i] \in \mathcal{S}$ for which x_i is an essential singularity. If none of these P_i becomes reducible over C', then there are no more combinations over C' than there are over C. Thus, if there are no hypergeometric solutions defined over C but there is a hypergeometric solution u defined over C' then at least one P_i must factor over C'. In Step **3** we pick a $[P_i, x_i]$, which may have been a wrong choice (P_i stays irreducible over C'), in which case Step **3a** will work, but it may also have been a right choice (meaning that P_i is reducible over C'), in which case Step **3b** will work. We then have to prove that in this case the bound in Step **3b** is correct. Suppose that P_i has degree d_i. P_i is reducible over C' and thus one of its irreducible factors will have degree $d_i' \leqslant \lfloor d_i/2 \rfloor$. By abuse of notation, we also denote x_i as a root of that factor. So then $[C'(x_i) : C'] = d_i'$, $[C' : C] \leqslant B$, so $[C'(x_i) : C] \leqslant d_i' B$. Now $C(x_i)$ is a field between $C'(x_i)$ and C having degree d_i over C, so $[C'(x_i) : C(x_i)] = [C'(x_i) : C]/d_i \leqslant d_i' B/d_i \leqslant \lfloor d_i/2 \rfloor B/d_i$. Now u is defined over C', hence also over $C'(x_i)$, which is an extension of degree $\leqslant \lfloor d_i/2 \rfloor B/d_i$ over our new field $C(x_i)$. Hence the degree bound used in Step **3b** is correct. □

9 An algorithm to find all hypergeometric solutions

We now present an algorithm to compute a basis of all hypergeometric solutions (up to conjugation over C) of a difference operator $L \in C(x)[\tau]$ where C is a field of characteristic 0. It uses the same strategy and satisfies the same specifications as Algorithm ExpSols in [CH04, Section 7].

Algorithm HypSols**:**
Input: a linear difference operator $L \in C(x)[\tau]$ and the field C.
Output: Sol, a basis of hypergeometric solutions of L up to conjugation over C.
1: Sol $:= \emptyset$.
2: $\mathcal{A} :=$ LocalTypeAt$\infty(L,\, C)$.
3: For $(c, v, d) \in \mathcal{A}$, do:
3a: $\mathrm{V}_{(\mathrm{c,v,d})} :=$ ESols$(L,\, C,\, (c, v, d))$,
3b: Sol $:=$ Sol $\cup\, \mathrm{V}_{(\mathrm{c,v,d})}$,
3c: Let $B_{(c,v,d)}$ be a bound for the number of linearly independent hypergeometric solutions with local type $(c, v, d + \mathbb{Z})$ at infinity,
3d: $B_{(c,v,d)} := B_{(c,v,d)}$ - Cardinality$(\mathrm{V}_{(\mathrm{c,v,d})})$,
3e: If $B_{(c,v,d)} \leqslant 1$, then $\mathcal{A} := \mathcal{A} \setminus \{(c, v, d)\}$.
4: If $\mathcal{A} = \emptyset$, then return Sol and stop.
5: If Sol $\neq \emptyset$, then
5a: Write $L = \tilde{L}$ LCLM$(\tau - \tau(s)/s$ and conjugates over $C \,|\, s \in$ Sol$)$. Run HypSols$(\tilde{L}, C)$,
5b: Of the solutions in **5a**, keep only the types not defined over their cvd-field,
5c: For each remaining type represented by t, do:
– Compute a basis $R_1, \ldots, R_s$ of rational solutions of $L\circledS(\tau - 1/t)$,
– Sol $:=$ Sol $\cup\, \{\mathrm{hyp}(t)R_1, \ldots, \mathrm{hyp}(t)R_s\}$.
5d: Return Sol and stop.
6: If the order is 2 then go to **10**.
7: Make L monic and let L^* be the adjoint of L.
8: Let S be the union of ESols$(L^*,\, C,\, (c, v, d))$ for all $(c, v, d) \in \overline{g}_\infty(L^*)$.
9: If $S \neq \emptyset$, then
9a: Write $L =$ LCLM$(\tau - \tau(s)/s$ and conjugates over $C \,|\, s \in S)^*\, \tilde{L}$,
9b: Return HypSols$(\tilde{L},\, C)$ and stop.
10: For $(c, v, d) \in \mathcal{A}$, do:
10a: HSol$(L,\, C(c, d),\, (c, v, d),\, B_{(c,v,d)},$ Sing, $\emptyset)$,
10b: If it finds a solution, then
– Optimize the solution found, denote it as u, and let $r = \tau(u)/u$,
– Compute a basis $R_1, \ldots, R_s$ of rational solutions of $L\circledS(\tau - 1/r)$,
– Sol $:= \{u\, R_1, \ldots, u\, R_s\}$,
– Write $L = \tilde{L}$ LCLM$(\tau - \tau(s)/s$ and conjugates over $C \,|\, s \in$ Sol$)$,
– Remove recursively from $\tilde{L}$ the solutions of the same type as conjugates of u,
– Run HypSols$(\tilde{L},\, C)$ and of its output, keep only the types that are not defined over their cvd-field. Then run Steps **5c, 5d**.
11: Return $\emptyset$ and stop.

We explain some points in the algorithm. Some remarks have already been made in [CH04].
- The bound $B_{(c,v,d)}$ used in Step **3c** is usually the $B_1(c, v, d)$ of Proposition 1. However, other

bounds can be used: the formal-sol-dimension of $(c, v, d + \mathbb{Z})$ (see Definition 13) or the bound coming from computations modulo a prime p (see Proposition 3 in Section 10.3).
- When we apply recursion on a left factor $\tilde{L}$ of L (Steps **5** or **10b**), the hypergeometric solutions of $\tilde{L}$ are in general not hypergeometric solutions of L. We will only use the types of the hypergeometric solutions of $\tilde{L}$, not the hypergeometric solutions themselves. We skip types defined over their cvd-field because all e-solutions have already been found.
- We will apply HSol only when all easy factors have been removed both on the left and on the right. Removal of easy factors, left or right, does not cause solutions to be lost because we only do this after all e-solutions have already been computed and stored in the set Sol, in Steps **1**,**2**,**3** of the algorithm. To find the easy left factors, in Step **8**, we apply ESols to the adjoint operator L^* (see for example [We01, Definition 3.2.1]) to find the e-solutions of L^* which correspond to easy left factors of L. Then we apply recursion on the remaining right hand factor.
- When we reach the point (Step **10**) where there is nothing left to do than entering the "hard case", we try to find a solution over an extension of a cvd-field with HSol. An *a priori* first bound is $B_1(c, v, d)$ (see Proposition 1). The variable Sing in Step **10a** is calculated during the first call to ESols in Step **3a**.
- Let u be a hypergeometric solution. We can write $\tau(u)/u = P_1/P_2$ where P_1 and P_2 are polynomials with gcd 1 and P_2 is monic. The field of definition of u is then the field generated over C by the coefficients of P_1 and P_2. In Step **10b**, by "optimizing the solution" we mean two things: (1). Using this solution u to find a solution of minimal algebraic degree over C. And (2). Making sure that the field that the algorithm gives for u (this field contains the field of definition of u) is actually equal to the field of definition of u. Both (1) and (2) are important. We want u to be of minimal algebraic degree so that we know that its conjugates are linearly independent. But the way we count the number of conjugates of u is not by looking at u, but by looking at the field given for u. The field provided for u by Algorithm HSol contains, but need not be equal to, the field of definition of u. So to optimize u, we take the field given for u, and then determine the subfield generated over C by the coefficients of P_1 and P_2. Then we find defining equations (*i.e.*, new RootOf's) for this subfield, and use them to rewrite the coefficients of P_1 and P_2. This then takes care of (2). To do (1), we could use the approach in Lemma 3, however, this is not necessary because the special choice that Algorithm ESols makes when the option "just one" is given causes (1) to be automatically satisfied. The polynomial Q is of minimal degree with this option given, which leads to uniqueness of $\tau(u)/u$ which in turn causes u to already be of minimal algebraic degree over C.
- When we "remove recursively from $\tilde{L}$ the solutions of the same type as u" (Step **10b**), we also remove solutions whose type is conjugated over C to the type of u. We will remove such solutions recursively because if we do not then it would be non-trivial to ensure that the "same" solution is not returned more than once. So before we call HypSols on the remaining left factor $\tilde{L}$ of L, we first remove from $\tilde{L}$ all hypergeometric solutions of the same type as (conjugates of) u, and we keep repeating this until $\tilde{L}$ no longer has solutions of this type. A recursive procedure to achieve that is the following: compute the rational solutions R_i of $\tilde{L}_{\circledS}(\tau - 1/r)$ where $r = \tau(u)/u$ is the certificate of u. Each R_i gives a first order right hand factor $\tau - r_i$ of $\tilde{L}$ where $r_i = r\tau(R_i)/R_i$. If there are none, then return $\tilde{L}$ else write $\tilde{L} = \tilde{L}_1$ LCLM$(\tau - r_1, \ldots, \tau - r_s,$ "and conjugates over C") and apply recursion on $\tilde{L}_1$.

9.1 Some possible improvements

We shall give two possible improvements that can be added to Algorithm HypSols. The first one is included in the Maple implementation.

Maps between $V_{p,l}(L)$ and $V_{p,r}(L)$, and possible degrees of field extensions: we show how the ranks of the maps between left and right solution spaces $V_{p,l}(L)$ and $V_{p,r}(L)$ at finite singularities p can be used to discard certain cases during the search phase of the algorithm.

Let p be a finite singularity of L. Let $g_{p,r} = g_{p,r}(L)$ be the smallest and $g_{p,l} = g_{p,l}(L)$ the largest element of $\overline{g}_p(L)$. In [Ho99, Section 4.2] it is shown how to construct two maps $E_{p,r} : V_{p,l}(L) \to V_{p,r}(L)$ and $E_{p,l} : V_{p,r}(L) \to V_{p,l}(L)$ by defining their action on explicit bases. In our implementation, we only use the rank of these maps. If rk_r is the rank of $E_{p,r}$, and rk_l is the rank of $E_{p,l}$, then this gives us the following information:

- there can be at most rk_r linearly independent hypergeometric solutions u with $g_p(u) = g_{p,r}$ (minimal valuation growth),
- at most $n - \mathrm{rk}_r$ with $g_p(u) > g_{p,r}$ (more than minimal valuation growth),
- at most rk_l with $g_p(u) = g_{p,l}$ (maximal valuation growth), and,
- at most $n - \mathrm{rk}_l$ with $g_p(u) < g_{p,l}$ (less than maximal valuation growth).

This can be exploited in several ways:

1. Suppose that there can be at most 2 independent solutions with less than maximal valuation growth, and suppose that the algorithm already found 2 such solutions. Then from that moment on, we no longer need to consider the case $g_p(u) < g_{p,l}$, and so we may assume $g_p(u) = g_{p,l}$.
2. Or suppose that there can be at most 1 linearly independent solution with minimal valuation growth. Then, if there exists such a solution u with $g_p(u)$ minimal, and if $p = q + \mathbb{Z}$ with $q \in C$, then the certificate $r(x) = u(x+1)/u(x)$ of u must be an element of $C(x)$. The reason for that is that if $r(x)$ was defined over an extension C' of C, then by taking conjugates of $r(x)$ over C' we would find more than 1 linearly independent solution with minimal valuation growth at p. More generally, suppose that there are at most s_1 independent solutions with maximal valuation growth, but s_2 have already been found. So there can remain at most $s_1 - s_2$ independent hypergeometric solutions $u(x)$ with maximal valuation growth, and this means that for computing the certificate $r(x) = u(x+1)/u(x)$ of such a solution, it is not necessary to consider field extensions of degree more than $s_1 - s_2$.

The above results generalize [Ho99, Theorems 2 and 3] and can be proven in a similar way.

Sorting the local types at infinity: it can be useful to sort the local types at infinity in such a way that we compute the solutions that have the smallest fields of definition first. The reason is that each time a solution is found, the number of combinations to be considered may be reduced following the items above, so we want to compute first those solutions that are the least expensive to calculate.

10 Modular improvements

We shall propose a number of improvements we can add to Algorithm HypSols using p-curvature computations. This section has its own theoretical interest since, as it is done in [CH04] for the differential case, we give links between the local information in characteristic zero and the roots of the characteristic polynomial of the p-curvature matrix.

10.1 Difference operators in characteristic p

In this part, most results are stated without proof since they are just adaptations of results in [CH04, 1.2] from the differential to the difference case. A classification (similar to that for the differential case) of difference modules in characteristic p can be found in [PS97, Chapter 5].

Let $\overline{\mathbb{F}}_p$ denote the algebraic closure of the finite field $\mathbb{F}_p$. We define the ring $\overline{\mathbb{F}}_p(x)[\tau]$ of difference operators in the same way as in characteristic zero. The main difference is that the constant field is then $\overline{\mathbb{F}}_p(x^p - x)$ (see [PS97, Chapter 5] or [GZ03, Theorem 3.1]). To simplify the notations, we set:

$$\lambda := x^p - x.$$

The central tool to study recurrence relations in characteristic p is the p-curvature; we give a definition of the *p-curvature* when the equation is written in matrix form $Y(x+1) = A(x)\,Y(x)$ with $A(x) \in \mathcal{M}_n(\overline{\mathbb{F}}_p(x))$ and for more details we refer to [PS97, Chapter 5] and references therein. Note that a difference operator $L \in \overline{\mathbb{F}}_p(x)[\tau]$ is naturally associated to a linear difference equation in matrix form; take for $A(x)$ the companion matrix of L.

Definition 19. *Let $Y(x+1) = A(x)\,Y(x)$ with $A(x) \in \mathcal{M}_n(\overline{\mathbb{F}}_p(x))$ be a linear difference equation. Its* p-curvature *is defined as the product of matrices $A(x+p-1)\cdots A(x+1)\,A(x)$.*

Consider the following group homomorphisms:

$$\tau_1 : \overline{\mathbb{F}}_p(x)^* \to \overline{\mathbb{F}}_p(\lambda)^*,\ \ u \mapsto u(x+p-1)\cdots u(x+1)\,u(x),$$

and

$$\tau_2 : \overline{\mathbb{F}}_p \to \overline{\mathbb{F}}_p,\ a \mapsto a^p - a,$$

which are related by Equation (12) below.

Lemma 4. **(i):** *The map $\tau_1 : \overline{\mathbb{F}}_p(x)^* \to \overline{\mathbb{F}}_p(\lambda)^*$ is a surjective multiplicative map with kernel $\{\tau(u)/u \,|\, u \in \overline{\mathbb{F}}_p(x)^*\}$.*
(ii): *The map τ_2 is a surjective additive map with kernel $\mathbb{F}_p$.*

Proof. **(ii)** is obvious, and **(i)** follows from **(ii)** and Equation (12) below (one can also follow a similar result for the differential case in [Pu95, Lemma 1.4.2]). □

For $L \in \overline{\mathbb{F}}_p(x)[\tau]$, let $\chi_p(L)$ denote the characteristic polynomial of the p-curvature of L. As we are only interested in first order factors, the only modular information that we can use are the roots in $\overline{\mathbb{F}}_p(\lambda)$ (with multiplicities) of $\chi_p(L)$. We denote $\mathcal{R}(\chi_p(L))$ the set of roots of $\chi_p(L)$ in $\overline{\mathbb{F}}_p(\lambda)$.

Lemma 5. *Let $L \in \overline{\mathbb{F}}_p(x)[\tau]$.*
(i): *If $L = L_1 L_2$, then $\chi_p(L) = \chi_p(L_1)\chi_p(L_2)$ with $\deg(\chi_p(L_i)) = \operatorname{order}(L_i)$.*
(ii): *The map from the set of monic first order right hand factors of L in $\overline{\mathbb{F}}_p(x)[\tau]$ to $\mathcal{R}(\chi_p(L))$ defined by $\tau - r \mapsto \tau_1(r)$ is well defined and surjective.*

As in the differential case, the key ingredient to use modular information for computing hypergeometric solutions of $L \in \overline{C}(x)[\tau]$ is the known fact that, with few assumptions on the prime p to be chosen, a factorization of L in characteristic zero can be reduced mod p.

Definition 20. *For some object f in characteristic zero, we note $f[p]$ its reduction modulo p in the sense of [CH04, Sections 1.3 and 7.1].*

The following proposition and its corollary can be proven as [CH04, Proposition 1.15] and [CH04, Corollary 1.16].

Proposition 2. *Let C be a field of characteristic zero, let $L \in C(x)[\tau]$ and let p be such that L can be reduced mod p. If $L = L_1 L_2$ then, after possibly replacing L_1 and L_2 by cL_1 and $\frac{1}{c}L_2$ for some constant $c \in C$, we have $L[p] = L_1[p]\ L_2[p]$.*

Corollary 1. *Let C be a field of characteristic zero, let $L \in C(x)[\tau]$ and let p be a prime such that L can be reduced mod p. Assume further that the order n of L does not drop after the reduction.*
(i): *If $\chi_p(L)$ is irreducible over $\overline{\mathbb{F}}_p(\lambda)$, then L is irreducible over $\overline{C}(x)$.*
(ii): *If $\chi_p(L)$ has no roots in $\overline{\mathbb{F}}_p(\lambda)$, then L has no hypergeometric solutions.*

10.2 Local types at infinity and $\mathcal{R}(\chi_p(L))$

Definition 21. *If a is an object in characteristic zero, then for $i \in \{1, 2\}$, we set $\tau_i(a) := \tau_i(a[p])$.*

If $d \in \overline{C}$ and $d[p]$ is defined then $\tau_2(d + \mathbb{Z})$ is defined since $\tau_2(i) = 0$ for $i \in \mathbb{Z}$.

Lemma 6. *Let $r(x) = c\,\frac{\tau(Q)}{Q}\,\prod_{i=1}^{m}(x - x_i)^{e_i} \in \overline{C}(x)$ with c, $x_i \in \overline{C}$, $e_i \in \mathbb{Z}$ and $Q \in \overline{C}[x]$. Assume further that the x_i, c, Q can be reduced* mod p *and that $Q[p] \neq 0$. Then*

$$\tau_1(r(x)) = c^p \prod_i (\lambda - \tau_2(x_i))^{e_i}. \tag{12}$$

Furthermore, if $r(x)$ has expansion $c\,t^v\,(1 + d\,t + \mathcal{O}(t^2))$ at infinity, then

$$\tau_1(r(x)) = c^p\,t_\lambda^v\,(1 + \tau_2(d)\,t_\lambda + \mathcal{O}(t_\lambda^2)), \tag{13}$$

where

$$t_\lambda := \frac{1}{\lambda} = \frac{1}{x^p - x}.$$

Proof. We have $\tau_1(r(x)) = \tau_1(c\,\frac{\tau(Q)}{Q}\,\prod_i (x - x_i)^{e_i}) = \tau_1(c)\,\prod_{i=1}^{s}(\tau_1(x - x_i))^{e_i}$ since, from Lemma 4, τ_1 is multiplicative and $\tau(Q)/Q \in \ker(\tau_1)$. Now, clearly $\tau_1(c) = c^p$. Furthermore, we have $\tau_1(x) = (x + p - 1)\cdots(x + 1)\,x = x^p - x$ and substituting $x = x - x_i$ in this equality leads to $\tau_1(x - x_i) = (x - x_i)^p - (x - x_i) = \lambda - \tau_2(x_i)$. Equation (12) is then clear.
For (13), we only need to prove that the coefficient $\tau_2(d)$ of t_λ is correct. Let $\tilde{d}$ denote this coefficient. From (12), we have $\tilde{d} = -\sum_i e_i \tau_2(x_i)$. But we also know from Section 1 that $d = -\sum_i e_i x_i$ so that an easy verification leads to $\tau_2(d) = \tilde{d}$. □

Definition 22. *Let $L = a_n\tau^n + \cdots + a_0\tau^0 \in C(x)[\tau]$ for some field C of characteristic zero and assume that a_n, $a_0 \neq 0$. After multiplying L on the left by a suitable element of $C(x)$, we may assume the coefficients a_i are in $C[x]$ and* $\gcd(a_0, \ldots, a_n) = 1$.

We say that a prime p is a good prime for L *if the two following conditions hold:*
(C1): *the coefficients of L can be reduced* mod p *and $a_0[p] \neq 0$, $a_n[p] \neq 0$.*
(C2): *the finite singularities can be reduced* mod p.

Condition **(C1)** ensures that the order does not decrease after reduction mod p. Condition **(C2)** simply means that the leading coefficient of the polynomials a_0 and a_n do not reduce to zero mod p. We give now a new result linking local types of hypergeometric solutions and eigenvalues of the p-curvature in $\overline{\mathbb{F}}_p(\lambda)$.

Theorem 2. *Let C be a field of characteristic zero, let $L \in C(x)[\tau]$ and let p be a good prime for L. Suppose that $\tau - r$ is a right hand factor of L with $r \in \overline{C}(x)$. Let $s = \tau_1(r) \in \mathcal{R}(\chi_p(L))$. Then there exist $e_i \in \overline{g}_{x_i}(L)$ at each finite singularity x_i of L and $(c, v, d+\mathbb{Z}) \in \overline{g}_\infty(L)$ such that c, d can be reduced mod p and*
(i): *$s = c^p \prod_i (\lambda - \tau_2(x_i))^{e_i}$,*
(ii): *the expansion at infinity of s is $c^p\, t_\lambda^v\, (1 + \tau_2(d)\, t_\lambda + \mathcal{O}(t_\lambda^2))$.*

Proof. One can prove (for example, adapt the proof of [CH04, Lemma 1.8] to the difference case) that if $\tau - r$ is a first order right hand factor of L then there exist a local type $e_i \in \overline{g}_{x_i}(L)$ at each finite singularity x_i of L, a local type $(c, v, d+\mathbb{Z}) \in \overline{g}_\infty(L)$ at infinity and a rational function $R \in \overline{C}(x)$ such that:

$$r = c\,\frac{\tau(R)}{R} \prod_i (x - x_i)^{e_i} = c\, t^v\, (1 + d\, t + \mathcal{O}(t^2)).$$

Since p satisfies condition **(C1)** in Definition 22, r can be reduced mod p (see Proposition 2). Now from condition **(C2)**, $\prod_i (x - x_i)^{e_i}$ can be reduced mod p and is further not zero mod p. Furthermore, R is defined modulo a multiplicative constant which we can choose in such a way that $R[p]$ is defined and not zero. Then $\tau(R)/R$ is also defined and not zero mod p. Consequently, c can be reduced mod p, and $c[p] \neq 0$, because c is the quotient of r and $\frac{\tau(R)}{R} \prod_i (x - x_i)^{e_i}$ which both reduce to something non-zero mod p. Then d can also be reduced mod p. Now the two formulas **(i)** and **(ii)** follow directly from Lemma 6. □

10.3 How to improve HypSols using the p-curvature?

The first improvement we propose for HypSols is a bound unrelated to the one of Proposition 1. Then we use the minimum of these bounds.

Proposition 3. *Let C be a field of characteristic zero, let $L \in C(x)[\partial]$ and let p be a good prime for L. Let $(c, v, d+\mathbb{Z}) \in \overline{g}_\infty(L)$. Let $B_2(c, v, d)$ be the number of roots of $\chi_p(L)$ (counted with multiplicity) that have expansion $c^p\, t_\lambda^v\, (1 + \tau_2(d)\, t_\lambda + \mathcal{O}(t_\lambda^2))$ at infinity. If u is a hypergeometric solution of minimal algebraic degree m over its cvd-field, with local type $(c, v, d+\mathbb{Z})$ at infinity, then $m \leqslant B_2(c, v, d)$.*

Proof. Let C' be the cvd-field. Let $R \in C'(x)[\tau]$ be the right hand factor of L whose solutions are spanned by all hypergeometric solutions of L that have local type $(c, v, d+\mathbb{Z})$ at infinity. Since there are at least m independent such solutions, the order o of R is at least m. All solutions of R have the same local type $(c, v, d+\mathbb{Z})$ at infinity. Now R factors as a product of first order factors $R = R_1 R_2 \cdots R_o$ because R has a basis of hypergeometric solutions. All R_i have the same local

type at infinity, so for each R_i, the local type at infinity must be $(c, v, d + \mathbb{Z})$. Hence the root of $\chi_p(R_i)$ has expansion $c^p\, t_\lambda^v\, (1 + \tau_2(d)\, t_\lambda + \mathcal{O}(t_\lambda^2))$ at infinity (see Theorem 2). The proposition now follows from the fact that R and $R_1, \ldots, R_o$ can be reduced mod p (see Proposition 2) and Lemma 5**(i)**. □

The bound $B_2(c, v, d)$ is in some cases used in the Maple implementation. By computing this bound before doing Step **3a** in Algorithm HypSols we can, if the bound is zero, discard (c, v, d) before calling Algorithm ESols. Some other possible improvements can be deduced from Theorem 2 (these have not been implemented):

(I). Suppose we are searching for the e-solutions of $L \in C(x)[\tau]$ that have local type $(c, v, d + \mathbb{Z})$ at infinity. Let p be a good prime for L. Distinct singularities do not necessarily stay distinct after reduction mod p. However, one can use the following approach to discard some combinations in Step **2** of Algorithm Search. Using the notation of Search: let $\mathcal{S} \subset \mathcal{R}(\chi_p(L))$ be the set of eigenvalues of the p-curvature matrix having expansion $c^p\, t_\lambda^v\, (1 + \tau_2(d)\, t_\lambda + \mathcal{O}(t_\lambda^2))$ at infinity. Now use only combinations $l_{i,j}$ in Step **2** of Algorithm Search that match an element $s \in \mathcal{S}$. Deciding if a combination matches s is done as follows: let $C'[p]$ denote C' reduced mod p as in [CH04]. Factor s in $(C'[p])(\lambda)$, let $s = V_1^{e_1} \cdots V_l^{e_l}$ where $e_i \in \mathbb{Z}$ and V_i irreducible in $(C'[p])[\lambda]$. For each $k \in \{1, \ldots, l\}$, check if the sum of all $m_{i,j,k} l_{i,j}$ adds up to e_k. Here $m_{i,j,k}$ denotes the maximal m for which V_k^m divides $\tau_1(P_{i,j})$.

Remark 5. *This method to discard some combinations is very close to the one used in [CH04, Section 5.2] in the differential case.*

(II). Let $(c, v, d + \mathbb{Z}) \in \overline{g}_\infty(L)$. If there is a good prime p for which c or d can not be reduced mod p, or c reduces to 0 mod p, then this $(c, v, d + \mathbb{Z})$ can be discarded by the proof of Theorem 2.

Example 10. *Taking $p = 29$ and $p = 5$ which are good primes in Example 7 one sees that all elements of $\overline{g}_\infty(L)$ given in Example 8 can be thrown away, so the example has no hypergeometric solutions.*

11 Computation timings

A Maple implementation of HypSols by the second author has been available since 2001 on `http://www.math.fsu.edu/~hoeij/maple.html` and is now included in Maple 9. To use it, do `with(LREtools):` and then call `hypergeomsols`. Older versions of Maple have an implementation of Petkovšek's algorithm (called here PetkoSols). This algorithm can still be called in Maple 9 by setting `_Env_old_hypergeomsols:=true` before calling `hypergeomsols`. A comparison on 36 examples can be found on `http://www.math.fsu.edu/~hoeij/comparison_hypergeomsols`.
The tests show that HypSols is much faster than PetkoSols except on very small examples. We give one additional example:

$$L(u) \;=\; u(x+2) - (x+1)\,(2\,x^2 + 3\,x + 2)\,u(x+1) + (x^6 + 2\,x^5 + x^4 - 2)\,u(x) \;=\; 0.$$

Recall that $\mathrm{hyp}(r)$ is short notation for a solution u of the operator $\tau - r$. The operator L has a basis of two hypergeometric solutions, which are conjugated over $\mathbb{Q}$. If HypSols is given the field $C = \mathbb{Q}$ then the output of HypSols will contain only one solution $u(x) = \mathrm{hyp}(x^3 + x^2 + \sqrt{2})$ because

the output is a basis *up to conjugation* over C. Note that HypSols needed to enter the "hard case" (Algorithm HSol) to find this solution. Since the user will expect `hypergeomsols` to return a basis of solutions, not a basis up to conjugation, Maple will compute all conjugates of $x^3 + x^2 + \sqrt{2}$ over $C = \mathbb{Q}$. The expression $\mathrm{hyp}(r)$ can be rewritten as $\prod_{k=0}^{x-1} r(k)$ but it can also be written as a product of Γ functions (Maple's choice depends on the factorization of r). The output of

```
with(LREtools): L := u(x)*(x^6+2*x^5+x^4-2)-(x+1)*(2*x^2+3*x+2)*u(x+1)+u(x+2);
hypergeomsols(L, u(x), {}, output=basis);
```

in Maple 9 on this example is

$$\prod_{k=0}^{x-1} \left(k^3 + k^2 + \sqrt{2}\right), \quad \prod_{k=0}^{x-1} \left(k^3 + k^2 - \sqrt{2}\right).$$

The computation takes about 1.2 seconds (Pentium 733 MHz). In PetkoSols the example takes about 436 seconds. Mathematica's implementation of Petkovšek's algorithm finds no solutions because it is limited to the case where a_0 and a_n have no irreducible factors of degree > 2, see [PWZ96, Section 8.5].

Another comparison between the algorithm in [Ho99] and Petkovšek's algorithm was done in [We01, Section 6.3]. Here, the author points out the problem with the number of cases Petkovšek's algorithm must consider. He also mentions the high cost of computing the sets $\overline{g}_p(L)$, most likely caused by not using truncated power series as in Section 4. The author also remarks that, unlike [Ho99], Petkovšek's algorithm may return too many (*i.e.*, linearly dependent) solutions.

12 An analogue algorithm for q-difference equations

In this section, we explain briefly how to develop the q-analogue of HypSols, for computing q-hypergeometric solutions of q-difference operators when q is not a root of unity and not zero. The q-analogue of Petkovšek's algorithm was given in [APP98].

We consider a field C of characteristic zero and its algebraic closure $\overline{C}$. To simplify the notations, we set $K := C(x)$.

Definition 23. *Let $q \in \overline{C}$, not zero and not a root of unity. A* q-difference operator $L := \sum_{i=0}^{n} a_i \, \tau^i$ *with $a_i = a_i(x) \in \overline{C}(x)$ is an operator that acts on $u = u(x)$ as follows:*

$$L(u(x)) = \sum_{i=0}^{n} a_i(x) \, u(q^i x).$$

In [PS97, Section 12.1], the authors prove the existence of a universal extension $\mathcal{V}_q$ for q-difference equations with coefficients in $\overline{C}(x)$.

Definition 24. *An expression $u(x) \in \mathcal{V}_q$ is called a* q-hypergeometric term *if $u(qx)/u(x) \in \overline{C}(x)$.*

We will define a notion of local types for first order q-difference operators (or q-hypergeometric terms) and construct sets of candidate local types for possible first order right hand factors of q-difference operators. We shall also exhibit Fuchs' relations and show that the type of a first order operator (see Definition 25) only depends on its local type at each point.

12.1 Local types

While a difference operator has only one special singularity, namely at ∞, a q-difference operator has two special singularities: 0 and ∞. Consider the two following groups with multiplication $\circledast$ where $\circledast$ is defined as in Definition 6:

$$K^*_{\circledast} := \{\tau - r \,|\, r \in K^*\} \quad \text{and} \quad K^*_R := \{\tau - \frac{\tau(r)}{r} \,|\, r \in K^*\}.$$

Note that $K^* \cong K^*_{\circledast}$ (resp. $K^*/C^* \cong K^*_R$) under $r \mapsto \tau - r$ (resp. $r \mapsto \tau - \tau(r)/r$).

Definition 25. *Let r in K^*. The* type of $\tau - r$ *is defined as the image of $\tau - r$ in $K^*_{\circledast}/K^*_R$.*

12.1.1 Special singularities

Here, we will define the local type at 0 and ∞ of $L = \tau - r$ with $r \in K^*$. Let $\langle q \rangle$ denote the multiplicative group generated by q (remember that q is not a root of unity and not zero).

Definition 26. *For $e \in \overline{C}^*$, denote $\overline{e} = \{e\, q^i \,|\, i \in \mathbb{Z}\}$, the image of e in $\overline{C}^*/\langle q \rangle$.*

Let

$$\mathcal{G}_0 = \mathcal{G}_\infty := \overline{C}^*/\langle q \rangle \;\times\; \mathbb{Z}.$$

We will use the additive notation for this group: $(\overline{c_1}, v_1) + (\overline{c_2}, v_2) = (\overline{c_1 c_2}, v_1 + v_2)$. The element $r \in K^*$ can both be written in the form $r = c_0\, x^{v_0}\,(1 + \mathcal{O}(x^1))$ and $r = c_\infty\, t^{v_\infty}\,(1 + \mathcal{O}(t^1))$ where $t := 1/x$, c_0, $c_\infty \in \overline{C}^*$ and v_0, $v_\infty \in \mathbb{Z}$.

Definition 27. *With these notations, the* local types *of $L = \tau - r$ at 0 and ∞ are respectively defined as:*

$$g_0(L) = (\overline{c_0}, v_0) \in \mathcal{G}_0,$$

and

$$g_\infty(L) = (\overline{c_\infty}, v_\infty) \in \mathcal{G}_\infty.$$

Note that g_0 (resp. g_∞) define homomorphisms from $K^*_{\circledast}/K^*_R$ to $\mathcal{G}_0$ (resp. $\mathcal{G}_\infty$). So $g_0(\tau - r)$ and $g_\infty(\tau - r)$ only depend on the type of $\tau - r$.

12.1.2 Other Singularities

Definition 28. *Let C be a field of characteristic zero and $L \in C[x][\tau]$ be a q-difference operator. Then $\alpha \in \overline{C}^*$ is a* problem point of L *if α is a root of $a_0(x)\, a_n(x/q^n)$.*
$\overline{\alpha} \in \overline{C}^*/\langle q \rangle$ is a singularity of L *if L has a problem point in $\overline{\alpha}$.*

Definition 29. *For $e \in \overline{C}^*$ and $r \in K^*$, we define the* valuation $v_e(r)$ of r at e *as the largest integer m such that $r/(x-e)^m \in \overline{C}[[x - e]]$. The valuation of* 0 *is ∞.*

Definition 30. *For $\overline{\alpha} \in \overline{C}^*/\langle q \rangle$, we define the group homomorphism $v_{\overline{\alpha}} : K^* \to \mathbb{Z}$ given by $v_{\overline{\alpha}}(r) = \sum_{e \in \overline{\alpha}} v_e(r)$. The* local type of $L = \tau - r$ at $\overline{\alpha}$ *is defined as*

$$g_{\overline{\alpha}}(L) = v_{\overline{\alpha}}(r) \in \mathbb{Z}.$$

It is easy to show that $g_{\overline{\alpha}}(\tau - r)$ only depends on the type of $\tau - r$.

12.2 Fuchs' relations

Let $\mathcal{G}_+$ be the additive group of functions $C^*/\langle q\rangle \to \mathbb{Z}$ with finite support and let

$$\mathcal{H} := \mathcal{G}_+ \times \mathcal{G}_0 \times \mathcal{G}_\infty.$$

Definition 31. *The* collection of local types *of $L = \tau - r$ is defined as*

$$g(L) := (\overline{\alpha} \mapsto g_{\overline{\alpha}}(L),\ g_0(L),\ g_\infty(L)) \in \mathcal{H}.$$

Lemma 7 (Fuchs' Relations)**.** *Let $L = \tau - r \in K^*_{\circledS}$. If $g(L) := (\overline{\alpha} \mapsto g_{\overline{\alpha}}(L), (\overline{c_0}, v_0), (\overline{c_\infty}, v_\infty)) \in \mathcal{H}$ is the collection of local types of L, then we have*

$$v_0 \ + \ v_\infty \ + \sum_{\overline{\alpha}\in\overline{C}^*/\langle q\rangle} g_{\overline{\alpha}}(L) = 0 \quad \text{and} \quad \overline{c_0} \ = \ \overline{c_\infty} \prod_{\overline{\alpha}\in\overline{C}^*/\langle q\rangle} (\overline{-\alpha})^{g_{\overline{\alpha}}(L)}. \tag{14}$$

Proof. One can check this for generators of the group $K^*_{\circledS}$, *i.e.*, the $\tau - c$ with $c \in \overline{C}^*$ and the $\tau - (x - e)$ with $e \in \overline{C}$. □

Let $\mathcal{H}_F$ be the subset of $\mathcal{H}$ containing all the elements satisfying Fuchs' relations given by Equation (14).

Proposition 4. *g induces a group isomorphism from $K^*_{\circledS}/K^*_R$ to $\mathcal{H}_F$.*

Proof. Let $g : K^*_{\circledS} \to \mathcal{H}$ be as in Definition 31. From the previous lemma, the image of g is contained in $\mathcal{H}_F$. The map is surjective since if $F = (f, (\overline{c_0}, v_0), (\overline{c_\infty}, v_\infty)) \in \mathcal{H}_F$, then $F = g(L)$ where $L = \tau - c_\infty\, x^{v_0}\,(x-\alpha_1)^{f(\alpha_1)} \cdots (x-\alpha_m)^{f(\alpha_m)}$. Here, c_∞ is a representant in $\overline{C}^*$ of $\overline{c_\infty}$ and $\alpha_1, \ldots, \alpha_m$ are representants in $\overline{C}^*$ for the $\overline{\alpha_i} \in \overline{C}^*/\langle q\rangle$ for which $f(\overline{\alpha_i}) \neq 0$, and $\overline{\alpha_i} \neq \overline{\alpha_j}$ if $i \neq j$. Furthermore, it is easy to check that such an L is in K^*_R only if $F = (0, (\overline{1}, 0), (\overline{1}, 0))$ which is the identity in $\mathcal{H}_F$. So the kernel of g is K^*_R. □

Consequently, the type of a q-difference operator of order one is determined by its local types. Now, as in the difference case, given $L \in K[\tau]$, one can effectively construct sets of candidate local types for possible order one right hand factors of L. Then q-HypSols, the q-analogue of HypSols, follows naturally.

Acknowledgments: the authors would like to thank M. A. Barkatou for helpful explanations and references concerning Section 5.

References

[Ab95] S. A. Abramov. Rational solutions of linear difference and q-difference equations with polynomial coefficients. In *(Russian) Programmirovanie*, **6**: 3-11, 1995; translation in *Program. Comput. Software 21*, **6**: 273-278, 1995.

[AB98] S. A. Abramov, M. A. Barkatou. Rational solutions of first order linear difference systems. In *Proceedings of ISSAC'98*, ACM Press, 1998.

[ABH05] S. A. Abramov, M. A. Barkatou, M. van Hoeij. Apparent Singularities of Linear Difference Equations with Polynomial Coefficients. To appear in *AAECC*, 2005.

[ABP95] S. A. Abramov, M. Bronstein, M. Petkovšek. On polynomial solutions of linear operator equations. In *Proceedings of ISSAC'95*, ACM, New York, 1995.

[APP98] S. A. Abramov, P. Paule, M. Petkovšek. q-Hypergeometric solutions of q-difference equations. In *Discrete Math.*, **180**: 3-22,1998.

[Ad28] C. R. Adams. On the irregular cases of the linear ordinary difference equations. In *Trans. Amer. Math. Soc.*, **30**: 507-541, 1928.

[Ba97] M. A. Barkatou. A fast algorithm to compute the rational solutions of systems of linear differential equations. In *Rapport de Recherche IMAG* (Grenoble), **RR 973-M-**, 1997.

[BD94] M. A. Barkatou, A. Duval. Sur les séries formelles solutions d'équations aux différences polynômiales. In *Ann. Inst. Fourier*(Grenoble), **44**(2): 495-524, 1994.

[Be94] E. Beke. Die Irreduzibilität der homogenen linearen differentialgleichungen. In *Math. Ann.*, **45**, 278-294, 1894.

[Bi30] G. D. Birkhoff. Formal theory of irregular linear difference equations. In *Acta Math.*, **54**: 205-246, 1930.

[BCS05] A. Bostan, T. Cluzeau, B. Salvy. Fast algorithms for polynomial solutions of linear differential equations. In *Proceedings of ISSAC'05*, July 24-27, Beijing, China, 2005.

[CH04] T. Cluzeau, M. van Hoeij. A modular algorithm to compute the exponential solutions of a linear differential operator. *Journal of Symbolic Computation*, **38**, 1043-1076, 2004.

[GZ03] M. Giesbrecht, Y. Zhang. Factoring and decomposing ore polynomials over $\mathbb{F}_p(t)$. In *Proceedings of ISSAC'03*, August 3-6, Philadelphia, Pennsylvania: 127-134, 2003.

[Ho98] M. van Hoeij. Rational solutions of linear difference equations. In *Proceedings of ISSAC'98*: 120-123, 1998.

[Ho99] M. van Hoeij. Finite singularities and hypergeometric solutions of linear recurrence equations. In *Journal of Pure and Applied Algebra*, **139**: 109-131, 1999.

[No29] N. E. Nörlund. Leçons sur les équations linéaires aux différences finies. *Gauthiers Villard et Cie*, Paris, 1929.

[Pe92] M. Petkovšek. Hypergeometric solutions of linear recurrences with polynomial coefficients. In *Journal of Symbolic Computation*, **14**(2-3): 243-264, 1992.

[PWZ96] M. Petkovšek, H. S. Wilf, D. Zeilberger. $A = B$. With a foreword by Donald E. Knuth. A. K. Peters, Ltd., Wellesley, MA, 1996.

[Pu95] M. van der Put. Differential equations in characteristic p. In *Compositio Mathematica*, **97**: 227-251, 1995.

[PS97] M. van der Put, M. F. Singer. Galois theory of difference equations. In *Lectures Notes in Mathematics*, vol. **1666** Springer, Berlin, 1997.

[We01] C. Weixlbaumer. Solutions of difference equations with polynomial coefficients. Diplomarbeit, RISC Linz, Johannes Kepler Universität, 2001.