A variant of van Hoeij's algorithm to compute hypergeometric term solutions of holonomic recurrence equations

Bertrand Teguia Tabuguia

University of Kassel, Heinrich-Plett-Str. 40. 34132 Kassel, Germany

Abstract

Linear and homogeneous recurrence equations having polynomial coefficients are said to be holonomic. These equations are useful for proving and discovering combinatorial and hypergeometric identities. Given a field $\mathbb{K}$ of characteristic zero, a_n is a hypergeometric term with respect to $\mathbb{K}$, if the ratio a_{n+1}/a_n is a rational function over $\mathbb{K}$. Two algorithms by Marko Petkovšek (1993) and Mark van Hoeij (1999) were proposed to compute hypergeometric term solutions of holonomic recurrence equations. The latter algorithm is more efficient and was implemented by its author in the Computer Algebra System (CAS) Maple through the command `LREtools[hypergeomsols]`.
We describe a variant of van Hoeij's algorithm that performs with the same efficiency without considering certain recommendations of the original version. We implemented our algorithm in the CASes Maxima and Maple. It also appears for some particular cases that our code finds results where `LREtools[hypergeomsols]` fails.
Our implementation is part of the `FPS` software which can be downloaded at `http://www.mathematik.uni-kassel.de/~bteguia/FPS_webpage/FPS.htm`. The command is `HypervanHoeij` for Maxima 5.44 and `rectohyperterm` for Maple 2021.

Keywords: Holonomic recurrence equations, hypergeometric terms, van Hoeij's algorithm, Petkovšek's algorithm, finite singularities, Fuchs relations, local types at infinity.
AMS subject Classification: 33F10, 39A06, 33C20, 68W30.

Email address: `bteguia@mathematik.uni-kassel.de` (Bertrand Teguia Tabuguia)
URL: `https://www.bertrandteguia.com` (Bertrand Teguia Tabuguia)

1. Introduction

Let $\mathbb{K}$ be a field of characteristic zero. $\mathbb{K}$ is mostly a finite extension field of the rationals. A hypergeometric term can always be written in the form

$$C^n \cdot R(n) \cdot h(n), \tag{1}$$

where $C \in \mathbb{K}$, $R(n) \in \mathbb{K}(n)$, and $h(n)$ is a hypergeometric term expressed in terms of factorials and shifted factorials (Pochhammer symbols[1]) such that $h(n+1)/h(n) \in \mathbb{K}(n)$ is monic (see [5], [15, Chapter 6]). Notice that the representation (1) is unique if we choose to write Pochhammer terms as $(p)_n$ with the real part of p in a fixed half-open real interval of unit amplitude. This rewriting creates a multiplicative rational function that is taken into account when computing $R(n)$. We will consider the interval $\mathcal{I} = (0, 1]$ in our algorithm, and say that Pochhammer parts, corresponding to $h(n)$, is taken modulo the integers ($\mathbb{Z}$) with respect to $\mathcal{I}$.

Example 1. $3^n n!$ *and* $7^n \frac{n^2+1}{n+2} \frac{(1/3)_n}{(3/4)_n}$ *are two hypergeometric terms of the form* (1).

We consider recurrence equations of the form

$$\sum_{i=0}^{d} P_i(n) \cdot a_{n+i} = 0, d \in \mathbb{N}, \tag{2}$$

for the indeterminate term a_n, and the coefficients $P_i(n) \in \mathbb{K}[n], i = 0, \ldots, d$. Two main algorithms were proposed to find all hypergeometric term solutions of (2).
Petkovšek's approach presented in [12], focuses on the computation of ratios[2] of hypergeometric term solutions and look for formulas afterward. However, his algorithm has an exponential worst-case complexity on the degree of P_0 and P_d. Thus this approach could not be considered as conclusive for such computations. The commands *solve_rec* of the CAS Maxima and originally *RSolve* of Mathematica implement Petkovšek's algorithm.
Van Hoeij's approach first described in [19] is much more efficient, and finds, moreover, a basis of hypergeometric term solutions of (2). Note that the final output in Petkovšek's algorithm is not necessarily a basis. Computational details of van Hoeij's algorithm were more explained and complemented in [5]. Another important point to notice in this algorithm is how unnecessary splitting fields that increase the running time during computations are avoided.

[1] For a given constant p, the Pochhammer symbol $(p)_n$ is 1 if $n = 0$ and $p \cdot (p+1) \cdots (p+n-1)$ if n is a positive integer.

[2] a_{n+1}/a_n for a hypergeometric term a_n.

Definition 2. *(see [19, Definition 9], [5, Definition 8]) A point $p+\mathbb{Z}, p \in \mathbb{K}$ is called finite singularity of (2) if there exists $\tau \in \mathbb{Z}$ such that $p+\tau$ is a root of $P_d(n-d) \cdot P_0(n)$.*

From this definition where invariance modulo the integers is put forward, one can see the connection between finite singularities and our rewriting of hypergeometric term Pochhammer parts.
The power of van Hoeij's algorithm comes from the following main concepts:

(1) local types[3] at infinity of hypergeometric term solutions of (2),

(2) local types or valuation growths of hypergeometric term solutions at finite singularities of (2).

The computations of (1) and (2) constitute the key steps of van Hoeij's algorithm and that is where our approach proceeds differently (see [15, Chapter 6]).

- For (1), van Hoeij's algorithm uses the Newton polygon algorithm whereas we use a method based on asymptotic expansion inspired by Petkovšek's algorithm Poly (see [12]).
- Computing (2) is inherent in van Hoeij's algorithm, but in our approach, this is automatically considered in the way we construct $h(n)$ in (1) by taking monic factors modulo the integers of P_0 and $P_d(n-d)$. This consideration is valid thanks to Petkovšek's approach.

Apart from these essential differences, it is not trivial to notice that both algorithms do the same thing, because their step orderings do not coincide either.
Hypergeometric terms are usually defined for evaluations at non-negative integers; for instance, combinatorics and power series coefficients (see [9], [10, Section 10.26], [15], [17]). This is settled in our approach by considering Pochhammer parts modulo the integers with respect to $\mathcal{I}$. Contrary to the current Maple (Maple 2021) `LREtools[hypergeomsols]`, this choice is independent of the recurrence equation considered. Sometimes `LREtools[hypergeomsols]` uses different Gamma representations for each hypergeometric term solution and this prevents the obtained basis to be used as a whole. One should also notice that the Pochhammer or factorial notations are closer to the combinatorial meaning than the Gamma notation which may give inconvenient results with evaluation like $\Gamma(1/2) = \sqrt{\pi}$. We will use the terminology "simple" formula for $h(n)$ to denote the representation in terms of factorials or Pochhammer symbols modulo the integers with respect to $\mathcal{I}$.
Let us give an illustrative example.

[3] This notion was introduced to study the local properties of difference operators at infinity (see [4, 6])

Example 3. *Consider the following holonomic recurrence equation*

$$\begin{aligned}
&81\,n^3(-2+n)(2592\,n^{15}+56592\,n^{14}+566784\,n^{13}+3438888\,n^{12}+14040866\\
&n^{11}+40413165\,n^{10}+83014167\,n^9+118689722\,n^8+105269208\,n^7+24761376\,n^6\\
&-78424336\,n^5-131026944\,n^4-108917280\,n^3-54383616\,n^2-15593472\,n-1990656)\\
&(-1+n)^3(1+2\,n)^5a_n-(-1+n)(6718464\,n^{24}+165722112\,n^{23}+1895913216\,n^{22}\\
&+13287379968\,n^{21}+63281637504\,n^{20}+213327813888\,n^{19}+505402785504\,n^{18}\\
&+757111794432\,n^{17}+271146179476\,n^{16}-2121306037512\,n^{15}-7223796390373\,n^{14}\\
&-14217526943124\,n^{13}-20381899157262\,n^{12}-22697247078996\,n^{11}\\
&-20140632084597\,n^{10}-14388789455784\,n^9-8294073141060\,n^8-3843447511168\,n^7\\
&-1418994576624\,n^6-411122122112\,n^5-91298680512\,n^4-14978958336\,n^3\\
&-1708259328\,n^2-120766464\,n-3981312)(n+1)^3a_{n+1}+32\,(2592\,n^{15}\\
&+17712\,n^{14}+46656\,n^{13}+41208\,n^{12}-78046\,n^{11}-305161\,n^{10}\\
&-498877\,n^9-523438\,n^8-374752\,n^7-212350\,n^6-77798\,n^5\\
&-23024\,n^4-4682\,n^3-641\,n^2-53\,n-2)(n+2)^3(3\,n+4)^4a_{n+2}=0 \quad (3)
\end{aligned}$$

Our Maple and Maxima implementation finds the following output with CPU times 0.172 *and* 0.328 *second, respectively.*

$$\left\{\frac{n\,!^3}{\left(\frac{1}{3}\right)_n^{\,4}(n-1)^3\,n^6},\frac{n\,(2n)\,!^5}{(n-2)\;(n-1)\;4^{5n}\,n\,!^4}\right\}. \quad (4)$$

Observe that all Pochhammer parts can be evaluated at non-negative integers.
Maple 2021 `LREtools[hypergeomsols]` *finds*

$$\left[\Gamma(n-2)\left(\Gamma(n+1/2)\right)^5n^2,\frac{\left(\Gamma(n-1)\right)^2\Gamma(n-2)\,(n-2)}{\left(\Gamma(n+1/3)\right)^4n^3}\right], \quad (5)$$

with CPU time 0.375 *second. In this output the Gamma terms* $\Gamma(n-2)$ *and* $\Gamma(n-1)$ *cannot be evaluated at* 0*. Moreover their arguments differs by* 1*, which shows that different integer shifts most be considered before initialization. However, we mention that* (4) *and* (5) *are equivalent.*
The Maxima (version 5.44) command `solve_rec` *finds*

$$a_n=\frac{\Gamma\left(\frac{1}{3}\right)^4\%k_1\,(n-2)\,!^3\,3^{-4n-8}\,81^n}{n^3\,\Gamma\left(\frac{3n+1}{3}\right)^4}+\frac{\%k_2\;(n-3)\,!\,n^2\,2^{5n+15}\,\Gamma\left(\frac{2n+1}{2}\right)^5}{\pi^{\frac{5}{2}}\,32^n}, \quad (6)$$

with CPU time 70.266 *seconds. This result arises the issue of finding closed-form of hypergeometric terms from their ratios computed with Petkovšek's algorithm. Nevertheless, the large computation time here is due to the degrees of the polynomial coefficients of* (3).

This paper goes as follows. In the next section, we derive an algorithm to compute holonomic recurrence equations satisfied by a list of linearly independent hypergeometric terms. This algorithm is useful to generate examples and observe some properties of hypergeometric terms by doing forward and backward computations. This can also be done using the Maple package `gfun` (see [13]), but with a slightly different strategy.
In Section 3 we give more details on how we normalize the Pochhammer parts of hypergeometric terms.
Section 4 describes our variant of van Hoeij's algorithm which efficiently computes a basis of hypergeometric term solutions of (2), using the representation (1). The paper ends with some comparisons with existing implementations.

2. Hypergeometric terms to holonomic recurrence equations

It is well known that linear combinations of holonomic functions are holonomic (see [14]). Since hypergeometric terms are holonomic, there exist algorithms to compute a holonomic recurrence equation of least order satisfied by a given linear combination[4] of hypergeometric terms. Throughout this section we assume there exists an algorithm for finding the rational function defined by the ratio of a hypergeometric term (see [9, Algorithm 2.2]). The algorithm of this section is a generalization of the case of two given linearly independent hypergeometric terms. Thus, we treat this particular case and by simple analogy, we give the general approach for a given list of linearly independent hypergeometric terms.

2.1. Case of two linearly independent hypergeometric terms

Let a_n and b_n be two linearly independent hypergeometric terms over $\mathbb{K}$ such that

$$a_{n+1} = r_1(n)a_n \text{ and } b_{n+1} = r_2(n)b_n, \tag{7}$$

where r_1 and r_2 are rational functions in $\mathbb{K}(n)$. As we consider two terms, the order of the recurrence equation sought is 2, so we are looking for a recurrence equation of the form

$$P_2(n)s_{n+2} + P_1(n)s_{n+1} + P_0(n)s_n = 0, \tag{8}$$

[4]Note that this generally reduces to a sum of hypergeometric terms. Therefore the main information here is that the given hypergeometric terms are distinct.

where P_0, P_1, P_2 are polynomials over $\mathbb{K}$, satisfied by a_n and b_n. We must assume that $P_0 \cdot P_2 \neq 0$, otherwise the recurrence equation can be reduced to a first order recurrence relation. Thus finding (8) is equivalent to searching for rational functions R_2 and R_1 such that

$$R_2(n)s_{n+2} + R_1(n)s_{n+1} + s_n = 0. \tag{9}$$

Using (7), we have

$$a_{n+2} = r_1(n+1)a_{n+1} \text{ and } b_{n+2} = r_2(n+1)b_{n+1}. \tag{10}$$

By substitution, a_n and b_n satisfy (9) if and only if

$$\begin{cases} r_1(n+1)R_2 + R_1 = -\frac{1}{r_1(n)} \\ r_2(n+1)R_2 + R_1 = -\frac{1}{r_2(n)} \end{cases}, \tag{11}$$

which is a linear system of two equations with two unknowns in $\mathbb{K}(n)$. Furthermore, a solution exists and is unique since the determinant of the system

$$r_a(n+1) - r_b(n+1) \neq 0 \tag{12}$$

by assumption. As a linear system of two equations, the exact solution is easy to compute, that is

$$R_1(n) = \frac{r_2(n+1)r_2(n) - r_1(n+1)r_1(n)}{r_1(n)r_2(n)(r_1(n+1) - r_2(n+1))}, \tag{13}$$

$$R_2(n) = \frac{r_1(n) - r_2(n)}{r_1(n)r_2(n)(r_1(n+1) - r_2(n+1))}. \tag{14}$$

Finally, the holonomic recurrence equation sought is found by multiplying the equation (9) by the common denominator of $R_1(n)$ and $R_2(n)$.

2.2. General case

Now we want to generalize the above approach for finitely many linearly independent hypergeometric terms. Let $a_n^{[i]}, i = 1, \ldots, d$ $(d \geqslant 1)$ be d given linearly independent hypergeometric terms over $\mathbb{K}$ such that

$$a_{n+1}^{[i]} = r_i(n)a_n^{[i]},\ i = 1, \ldots, d, \tag{15}$$

for some rational functions r_i. The vector $(R_1(n), R_2(n), \ldots, R_d(n))^T \in \mathbb{K}(n)^d$ of rational coefficients of the recurrence equation

$$R_d(n)s_{n+d} + R_{d-1}(n)s_{n+d-1} + \ldots + R_1(n)s_{n+1} + s_n = 0 \tag{16}$$

satisfied by each hypergeometric term $a_n^{[i]}$, is the unique vector solution $v \in \mathbb{K}(n)^d$ of the matrix system

$$\left[\prod_{k=1}^{j-1} r_i(n+k)\right]_{i,j=1,\ldots,d} \cdot v = -\left(\frac{1}{r_i(n)}\right)^T_{i=1,\ldots,d}. \tag{17}$$

In case there are linearly dependent hypergeometric terms, one can still use this process by replacing the arbitrary constants appearing in the solution of (17) by zero. This is how we implemented this method. The steps of the algorithm can be summarized as follows.

Algorithm 1 Compute the holonomic recurrence equation of least order for the sum of hypergeometric terms

---

Input: A list $L := [h_1, \ldots, h_d]$ of hypergeometric terms in the variable n and a symbol a.

Output: A holonomic recurrence equation in a_n of least order satisfied by the elements in L.

1. Let $R := [r_i(n), \ldots, r_d(n)]$ be the ratios of the elements in L.
2. If there are some irrational functions in R then stop and return FALSE. No holonomic recurrence equation can be found.
3. Let
$$M := \left[\prod_{k=1}^{j-1} r_i(n+k)\right]_{i,j=1,\ldots,d}.$$
4. Let
$$b := \left(\frac{1}{r_i(n)}\right)^T_{i=1,\ldots,d}.$$
5. Let V be the solution of the matrix system $M \cdot v = b$.
6. If there are arbitrary constants in V then substitute those by zero.
7. Let $RE := a_n + \sum_{i=1}^{d} V[i] \cdot a_{n+i}$, where $V[i]$ denotes the i^{th} component in V.
8. Multiply RE by the common denominator of the components of V and return the result with equality to 0, after factoring the coefficients.

Example 4. *We implemented this algorithm as sumhyperRE. Let us consider Example 4.1 in [12] and make a backward computation to find the recurrence equation for*

$$\frac{1}{(n+1)(n+2)}, \quad \textit{and} \quad \frac{(-1)^n(2n+3)}{(n+1)(n+2)}.$$

Our Maxima code gets

(% i1) *sumhyperRE([1/((n+1)*(n+2)), (-1)^n*(2*n+3)/((n+1)*(n+2))],a[n]);*

$$-(n+4)\ a_{n+2} - a_{n+1} + (n+1)\ a_n = 0, \qquad (\%\ \text{o1})$$

which is the expected result. Next, we recover the Fibonacci recurrence from the golden number and its conjugate.

(% i2) *sumhyperRE([(1-sqrt(5))^n/2^n, (1+sqrt(5))^n/2^n], a[n]);*

$$-a_{n+2} + a_{n+1} + a_n = 0 \qquad (\%\ \text{o2})$$

The latter example illustrates an important point of the algorithm. Indeed, when considering extension fields to determine hypergeometric term solutions, the conjugates of algebraic numbers involved are also part of the solution basis. We will give more details about this in Section 4.

3. "Simple" formulas for hypergeometric terms

Let a_n be a hypergeometric term over a field $\mathbb{K}$ of characteristic zero. Then by definition $r(n) := a_{n+1}/a_n \in \mathbb{K}(n)$. $\mathbb{K}$ is taken as the minimal extension field of $\mathbb{Q}$ where the numerator and the denominator of $r(n)$ split. We wish to write the formula of a_n using only numbers appearing in the splitting field of $r(n)$ with factorials, and when not trivially possible, Pochhammer symbols, thus allowing evaluations at non-negative integers. This is what we call a "simple" formula. One could say that a formula is considered to be "simple" when it presents more familiar objects from mathematical dictionaries in a reduced form. In the sense of computing formulas of hypergeometric terms, this consists of simplifying as much as possible, Pochhammer symbols to rational multiples of factorials with positive integer-linear arguments. In this section, we present preliminary steps to recover the representation (1) and gather some classical rules as an algorithm to simplify its Pochhammer part. Similar computations can be found in [8]; what is worth noticing is the consideration we make to get "simple" formulas in Section 4.
Consider a rational function

$$r(k) := \frac{P(k)}{Q(k)},\ P(k), Q(k) \in \mathbb{K}[k],\ Q(k) \neq 0 \ \text{ for integers } k \geqslant 0 \qquad (18)$$

such that P and Q do not have non-negative integer roots. We also assume that roots of P and Q are all distinct. We will see in the next section how the computations are done to satisfy these assumptions. For example, a hypergeometric term ratio $r(k)$ with

non-negative integer zeros and poles would implicitly be replaced by $r(k+m)$, where $m = \max\{j \in \mathbb{N}_{\geqslant 0} \,:\, Q(j) \cdot P(j) = 0\}$[5].
We consider a hypergeometric term defined with the property

$$a_{k+1} = r(k)a_k, \quad \text{for integer } k \geqslant 0. \tag{19}$$

Computing a "simple" formula of such a term is to find its general expression a_n for a positive integer n provided that the corresponding initial values a_0 is given. That is the result of the product

$$\prod_{k=0}^{n-1} r(k). \tag{20}$$

For that purpose, the first step is to split the polynomials of r as follows

$$r(k) = C\frac{(k+a_1)(k+a_2)\cdots(k+a_p)}{(k+b_1)(k+b_2)\cdots(k+b_q)}, \tag{21}$$

where p and q are, respectively, the degrees of P and Q; C is a constant representing the ratio of the leading coefficients of P and Q; and $-a_i$'s[6], $0 \leqslant i \leqslant p$, and $-b_j$'s, $0 \leqslant j \leqslant q$ are the zeros and poles of r, respectively. From the Pochhammer symbol definition, using (21) it follows that

$$\prod_{k=0}^{n-1} r(k) = C^n \frac{(a_1)_n (a_2)_n \cdots (a_p)_n}{(b_1)_n (b_2)_n \cdots (b_q)_n}. \tag{22}$$

Thus we are called to try simplifications of ratios and products of Pochhammer symbols and some isolated ones. Many such computations can be found in books or undergraduate courses, see for example [9, Exercises 1.1 - 1.5]. Bellow, we recall some classical ones. x and y denote some numbers, and j an integer.

Isolated Rule Assume x is rational, then one can simplify $(x)_n$, according to the following cases.

1. if $x \in \mathbb{N}$, then

$$(x)_n = x \cdot (x+1) \cdots (x+n-1) = \frac{(x+n-1)!}{(x-1)!} \tag{23}$$

[5] $\mathbb{N}_{\geqslant 0} = \{0, 1, 2, \ldots\}$
[6] To not confuse with a_k in (19).

2. Else if x has a denominator equal to 2, then let $s \in \mathbb{N}$ such that $x = \frac{s}{2}$. s is necessarily an odd integer since $x \notin \mathbb{N}$. We set $s = 2t+1$, $t \in \mathbb{N}_{\geqslant 0}$, then it follows that

$$\begin{aligned}(x)_n &= \frac{s}{2}\cdot\left(\frac{s}{2}+1\right)\cdots\left(\frac{s}{2}+n-1\right)\\ &= \frac{(2t+1)\cdot(2(t+1)+1)\cdots(2(t+n-1)+1)}{2^n}\\ &= \frac{(2\,(t+n))\,!}{(2t)!\cdot(2t+2)\cdots(2(t+n-1)+2)\cdot 2^n}\\ &= \frac{(2(t+n))\,!}{(2t)!\,4^n\binom{t+n}{n}n!}.\end{aligned} \tag{24}$$

3. Otherwise no simplification is done for $(x)_n$.

Ratio Rule Assume $x - y = j > 0$. Then we have

$$\frac{(y)_n}{(x)_n} = \frac{(y)_j\cdot(y+j)\cdots(y+n-1)}{(y+j)\cdots(y+j+n-1)} = \frac{(y)_j}{(y+n)_j}. \tag{25}$$

Therefore for $x - y = j \in \mathbb{Z}$,

$$\frac{(y)_n}{(x)_n} = \begin{cases}\frac{(y)_j}{(y+n)_j} & \text{if } j>0\\ \frac{(x+n)_{-j}}{(x)_{-j}} & \text{if } j<0\end{cases}. \tag{26}$$

This shows that differences between zeros and poles of r in (21) should be checked before applying the **Isolated Rule** in order to apply (25) which can simplify two Pochhammer symbols at the same time. Fortunately, these nice computations can be done in Maxima by combining `makegamma()`, `makefact()`, `minfactorial()` and `factor()` as below.

```
(% i1) r:pochhammer(7/3,n)/pochhammer(1/3,n);
```

$$\frac{\left(\frac{7}{3}\right)_n}{\left(\frac{1}{3}\right)_n} \tag{\% o1}$$

```
(% i2) factor(minfactorial(makefact(makegamma(r))));
```

$$\frac{(3n+1)\ (3n+4)}{4} \tag{\% o2}$$

Product Rule We consider the following two rules to simplify $(x)_n(y)_n$.

- Assume $y-x=1/2$, then multiplying $(x)_n$ and $(y)_n=(x+1/2)_n$ by 2^n leads to the relation
$$(x)_n \cdot \left(x+\frac{1}{2}\right)_n = \frac{(2x)_{2n}}{4^n}. \tag{27}$$
- Assume $y-x=j>0$, it is easy to see that
$$(x)_n \cdot (y)_n = (x)_n \cdot (x+j)_n = \frac{(x)_{n+j}^2}{(x)_j (x+n)_j}. \tag{28}$$

More generally, one can find a "simple" formula of a hypergeometric term Pochhammer part with ratio having non-negative integer zeros and poles by applying the following algorithm.

Algorithm 2 Compute $\prod_{k=1}^{n-1} r(k)$

Input: A rational function $r := r(n)$ and a variable n.
Output: A formula of $\prod_{k=1}^{n-1} r(k)$ in terms of factorial and Pochhammer symbols.

1. Factorize r and write it in terms of linear factors and set
$$h := r = C\frac{(n+a_1)(n+a_2)\cdots(n+a_p)}{(n+b_1)(n+b_2)\cdots(n+b_q)}.$$
2. Substitute C by C^n in h.
3. For each a_i, $i=1,\ldots,p$ do
   (a) if there is b_j in h such that $a_i-b_j\in\mathbb{Z}$ then substitute $\frac{n+a_i}{n+b_i}$ by applying the **Ratio Rule** accordingly.
4. For the remaining a_i's (resp. b_j's) do
   (a) if there is $a_{i'}$ (resp. $b_{j'}$) such that $a_i-a_{i'}=\pm 1/2$ or $a_i-a_{i'}\in\mathbb{Z}$ (resp. $a_j-a_{j'}=\pm 1/2$ or $a_j-a_{j'}\in\mathbb{Z}$) then substitute $(n+a_i)(n+a_{i'})$ (resp. $(n+b_j)(n+b_{j'})$) by applying the **Product Rule** accordingly.
5. Substitute the remaining $n+a_i$'s and $n+b_j$'s by the result of the **Isolated Rule** applied to $(a_i)_n$ and $(b_j)_n$ respectively.
6. Return h.

For a "fair" comparison of efficiency between our implementation and the current Maple `LREtools[hypergeomsols]`, we did not considered the **Product Rule** in our implementation since `LREtools[hypergeomsols]` does not apply simplifications as we

do. Indeed, these computations are supplementary steps that we use in the algorithm of Section 4. However, for power series computations (see [17]), all rules are applied to get nice power series formulas. Algorithm 2 is implemented in our Maxima package as `pochfactorsimp(r,n)`.

Example 5.

(% i1) pochfactorsimp(-1/(2(n+1)*(2*n+1)),n);*

$$\frac{(-1)^n}{(2n)\,!} \qquad (\%\ \text{o1})$$

*(% i2) pochfactorsimp((2*n+3)^2/((n+1)*(2*n+1)),n);*

$$\frac{(2n+1)\ 2^{n-1}\ (2\,(n+1))\,!}{(n+1)\ 4^n\ n\,!^2} \qquad (\%\ \text{o2})$$

4. Basis of hypergeometric term solutions

Let us rewrite (2) as follows.

$$P_d(n)a_{n+d} + P_{d-1}(n)a_{n+d-1} + \cdots P_1(n)a_{n+1} + P_0(n)a_n = 0, \tag{29}$$

with polynomials $P_i(n) \in \mathbb{K}[n], i = 0, \ldots, d$ such that $P_0(n) \cdot P_d(n) \neq 0$. $\mathbb{K}$ is a field of characteristic zero.
We have seen how to compute a holonomic recurrence equation of lowest order satisfied by a given number of linearly independent hypergeometric terms. Any computed hypergeometric term solution of such a holonomic recurrence equation is a linear combination of these linearly independent terms. The algorithm of this section is a kind of reverse process which for a given holonomic recurrence equation (29) computes a basis of at most d hypergeometric term solutions of (29).
In the first place, we establish (1) to see hypergeometric terms in normal forms (see [7, Chapter 3]). Let $a_n, n \in \mathbb{N}_{\geqslant 0}$, be a hypergeometric sequence such that $r(n) = a_{n+1}/a_n \in \mathbb{K}(n)$. Then we have

$$\frac{a_1}{a_0} = r(0),\ \frac{a_2}{a_1} = r(1),\ \ldots,\ \frac{a_n}{a_{n-1}} = r(n-1),\ \ n \geqslant 1,$$

and therefore

$$\frac{a_n}{a_0} = \prod_{k=0}^{n-1} \frac{a_{k+1}}{a_k} = \prod_{k=0}^{n-1} r(k) \Rightarrow a_n = a_0 \prod_{k=0}^{n-1} r(k). \tag{30}$$

Factorizing $r(n)$ over $\mathbb{K}$ gives

$$r(n) = C\frac{\prod_{i=1}^{I}(n-\alpha_i)}{\prod_{j=1}^{J}(n-\beta_j)}, \tag{31}$$

where C is a constant. Note that contrary to (21), in (31) $r(n)$ is considered in a more general setting; α_i and β_j are not uniquely determined and may have negative or positive real parts, which is more general than avoiding non-negative integer values.
Combining (30) and (31) leads to

$$a_n = a_0 \cdot C^n \cdot \frac{(-\alpha_1)_n \cdots (-\alpha_I)_n}{(-\beta_1)_n \cdots (-\beta_J)_n}. \tag{32}$$

Now we want to write each Pochhammer symbol modulo $\mathbb{Z}$ in a certain real interval, i.e., the real parts of the arguments of Pochhammer terms can be chosen to belong to an interval of amplitude 1. This is an interesting observation made by van Hoeij. In our case, we choose to rewrite the Pochhammer symbols modulo $\mathbb{Z}$ so that $\alpha_i, \beta_j \in \mathcal{I} := [-1, 0)$. Each Pochhammer symbol is then substituted by the product of a polynomial and another Pochhammer term whose argument differs by an integer u. Precisely, let y be a real number (for the case of complex numbers, the computations are applied on their real parts), then its corresponding value in $\mathcal{I}$ is $u = y - \lfloor y \rfloor - 1$ and we have

$$\begin{aligned}
(y)_n &= \frac{(u)_n \cdot (u+n) \cdots (y+n-1)}{u \cdot (u+1) \cdots (y-1)} \\
&= (u)_n \cdot \frac{(u+n)_{y-u}}{(u)_{y-u}} &(33)\\
&= (y - \lfloor y \rfloor - 1)_n \cdot \frac{(n + y - \lfloor y \rfloor - 1)_{\lfloor y \rfloor + 1}}{(y - \lfloor y \rfloor - 1)_{\lfloor y \rfloor + 1}}. &(34)
\end{aligned}$$

After applying (34) to each Pochhammer symbol in (32), the remaining expression will have Pochhammer terms having arguments with real parts in $(0, 1]$. These terms may have more coincidence than the $(-\alpha_i)_n$ and $(-\beta_j)_n$ in (32) since all Pochhammer terms in (32) whose arguments differ by an integer give the same Pochhammer term modulo $\mathbb{Z}$ after substitution. Therefore there exists a rational function $R(n) \in \mathbb{K}(n)$ and some constant numbers $\tilde{\alpha_1}, \ldots, \tilde{\alpha_I}, \tilde{\beta}, \ldots, \tilde{\beta_J}$, with real parts in $\mathcal{I}$, such that

$$a_n = R(n) \cdot C^n \cdot \frac{(-\tilde{\alpha_1})_n \cdots (-\tilde{\alpha_I})_n}{(-\tilde{\beta_1})_n \cdots (-\tilde{\beta_J})_n}. \tag{35}$$

The constant a_0 is neglected by linearity since we will look for a basis of hypergeometric term solutions of (29).

Considering multiplicities e_k over $\mathbb{Z} \setminus \{0\}$ and replacing $-\tilde{\alpha}_i$ and $-\tilde{\beta}_j$, by θ_k, we get the normal form

$$a_n = C^n \cdot R(n) \cdot h(n) := C^n \cdot R(n) \cdot \prod_{k=1}^{K} (\theta_k)_n^{e_k} \ (\theta_k \in \mathbb{K}, \text{ with real part in } \mathcal{I}), \qquad (36)$$

$K \leqslant I + J$. This time all the involved data are uniquely determined. The ratio $r(n)$ can be rewritten as

$$r(n) = \frac{a_{n+1}}{a_n} = \frac{R(n+1)}{R(n)} \cdot C \cdot h(n+1)/h(n) = \frac{R(n+1)}{R(n)} \cdot C \cdot \prod_{k=1}^{K} (n+\theta_k)^{e_k} \ \in \mathbb{K}(n). \quad (37)$$

Mark van Hoeij uses Gamma representations in (36) and denotes it singularity structure of a_n (see [19, 5]). This representation can be seen as the endpoint of our algorithm when it computes an element of the basis of hypergeometric terms sought. In fact, the goal of computing a basis of all hypergeometric term solutions of (29) is equivalent to finding solutions of (29) with the structure (36).

4.1. Monic factors modulo $\mathbb{Z}$

Lemma 6. *([12, Algorithm Hyper], [5, Left and Right solutions]) The ratio of the Pochhammer part $h(n+1)/h(n)$ of hypergeometric term solutions of (29) are built from monic factors of $P_d(n-d)$ for the numerators and $P_0(n-1)$ for the denominators.*

This lemma allows us to apply factorization modulo $\mathbb{Z}$ on $P_0(n)$ and $P_d(n)$. In fact the ratio of the Pochhammer part $h(n+1)/h(n)$ in (37) is obtained from factorization of $P_0(n)$ and $P_d(n)$ modulo $\mathbb{Z}$. Let us generate a recurrence equation that will be used while describing the steps of our algorithm.

```
(% i1) RE:sumhyperRE([binomial(n+3,n),1/n!,(-1)^n/n,
(-1)^n/pochhammer(1/2,n)^2],a[n])$
```

We do not display the output to save space. We will refer to this recurrence equation as (RE). The leading term is

```
(% i2) first(lhs(RE));
```

$$(n+2)\,(n+3)\,(n+4)\,(2n+7)^2\,(\,64n^{11}+1536n^{10}+16176n^9+98080n^8+377372n^7 +955200n^6+1584741n^5+1631354n^4+852544n^3-25229n^2-264212n-94472\,)\,a_{n+4}, \qquad (\%\ 2)$$

and the trailing term

(% i3) last(lhs(RE));

$$4n\ (n+4)\,(\,64n^{11}+2240n^{10}+35056n^9+323344n^8+1949788n^7+8053956n^6 \\ +23188049n^5+46338535n^4+62583534n^3+53821965n^2+26011175n+5133154\,)\ a_n.$$

(% o3)

For more clarity, we will present computations over the rationals. The case of extension fields works similarly, this choice is just to avoid lengthy notations for roots labeling.
For the leading polynomial coefficient, the monic factors to be considered after factorization in $\mathbb{Q}$ modulo $\mathbb{Z}$ with roots real parts in $\mathcal{I}$ are

$$(n+1)^{e_1}\left(n+\frac{1}{2}\right)^{e_2},\ \text{ for } 0\leqslant e_1\leqslant 3,\ 0\leqslant e_2\leqslant 2.$$

For the trailing term we have

$$(n+1)^{e}\ \text{ for } 0\leqslant e\leqslant 2.$$

Therefore ratios of Pochhammer parts of hypergeometric term solutions are among the following

$$1,\frac{1}{(n+1)},\frac{1}{(n+1)^2},\frac{1}{(n+1)^3},\frac{1}{\left(n+\frac{1}{2}\right)},\frac{1}{\left(n+\frac{1}{2}\right)^2},(n+1) \\ \frac{(n+1)}{\left(n+\frac{1}{2}\right)},\frac{(n+1)}{\left(n+\frac{1}{2}\right)^2},\frac{(n+1)^2}{\left(n+\frac{1}{2}\right)},\frac{(n+1)^2}{\left(n+\frac{1}{2}\right)^2} \tag{38}$$

Observe that none of these ratios has a non-negative integer zero or pole, hence the type of rational function that we treat with Algorithm 2.
Moreover, not all ratios in (38) should be considered because the exponents of each linear factor appearing in the possible ratios of hypergeometric term solutions can be bounded from the given holonomic recurrence equation. For this purpose van Hoeij's algorithm uses the notion of valuation growth or local types of difference operators at finite singularities [19, Definition 9]. Such a point is simply a root modulo $\mathbb{Z}$ of the trailing or the leading polynomial coefficient of (29) as we considered.
Since we are already computing ratios of Pochhammer parts of hypergeometric term solutions, we proceed in a slightly different way than what is described in ([19, 5]) to compute exponent bounds at finite singularities.

Theorem 7. *The exponent bounds at finite singularities result from taking the minimum exponents (or valuations) taken by a factor modulo $\mathbb{Z}$ in the trailing and the leading polynomial coefficients of the initial recurrence equation as lower bounds; this makes the upper bounds to be automatically considered while computing ratios of Pochhammer parts.*

Coming back to our example, it follows that ratios with denominator $(n+1/2)$ should be removed. Therefore the remaining ratios are

$$1, \frac{1}{(n+1)}, \frac{1}{(n+1)^2}, \frac{1}{(n+1)^3}, \frac{1}{\left(n+\frac{1}{2}\right)^2}, (n+1)$$
$$\frac{(n+1)}{\left(n+\frac{1}{2}\right)^2}, \frac{(n+1)^2}{\left(n+\frac{1}{2}\right)^2}. \tag{39}$$

Note that these considerations already present an important gain of efficiency compared to the algorithm in [12].

4.2. Local type at infinity

Without ambiguity, we will more often use the terminology "local type" instead of "local type at infinity" since we only consider computations at infinity. This is about a characteristic property of hypergeometric term solutions of holonomic recurrence equations.
We study the behavior of a hypergeometric term (a_n) ratio $r(n)$ at infinity. Indeed, at ∞ we can write

$$r(n) = c \cdot n^{\nu} \cdot \left(1 + \frac{b}{n} + O\left(\frac{1}{n^2}\right)\right), \tag{40}$$

with the unique triple (ν, c, b) called the local type of a_n at ∞.

Theorem 8 (Fuchs Relations)**.** *Let $R(n) = \frac{N(n)}{U(n)}$ with $N(n), U(n) \in \mathbb{K}[n]$. The following relations between the local type of a hypergeometric term a_n given by (36) hold:*

i. $\nu = \sum_{k=1}^{K} e_k$,

ii. $b = \sum_{k=1}^{K} \theta_k \, e_k \; + \deg(N(n)) - \deg(U(n))$,

iii. $c = C$,

where (ν, c, b) denotes the local type of a_n at ∞.

Proof. From (37) we know that

$$r(n) = \frac{a_{n+1}}{a_n} = C \cdot \left(\frac{R(n+1)}{R(n)} \cdot \prod_{k=1}^{K} (n+\theta_k)^{e_k}\right). \tag{41}$$

We would like to compute a truncated asymptotic expansion of (41). This can be seen as the result of the product of asymptotic expansions of the form (40) of $\frac{R(n+1)}{R(n)}$ and $\prod_{k=1}^{K}(n+$

$\theta_k)^{e_k}$ times C. Since $R(n) = \frac{N(n)}{U(n)}$, the highest degree of n in its asymptotic expansion is $\delta = \deg(N(n)) - \deg(U(n))$. So we can write

$$R(n) = c_R \cdot n^\delta \cdot \left(1 + \frac{b_R}{n} + O\left(\frac{1}{n^2}\right)\right), \tag{42}$$

for some constants c_R, b_R. Let us now deduce a truncated asymptotic expansion of $R(n+1)$.

$$\begin{aligned} R(n+1) &= c_R \cdot (n+1)^\delta \cdot \left(1 + \frac{b_R}{n+1} + O\left(\frac{1}{n^2}\right)\right) \\ &= c_R \cdot n^\delta \left(1 + \frac{1}{n}\right)^\delta \cdot \left(1 + \frac{b_R}{n\left(1 + \frac{1}{n}\right)} + O\left(\frac{1}{n^2}\right)\right) \\ &= c_R \cdot n^\delta \left(1 + \frac{\delta}{n} + \sum_{j=2}^{\delta} \binom{\delta}{j} \left(\frac{1}{n}\right)^j\right) \cdot \left(1 + \frac{b_R}{n} + O\left(\frac{1}{n^2}\right)\right) \\ &= c_R \cdot n^\delta \left(1 + \frac{b_R + \delta}{n} + O\left(\frac{1}{n^2}\right)\right). \end{aligned} \tag{43}$$

Thus from (42) and (43) the first order asymptotic expansion of $\frac{R(n+1)}{R(n)}$ yields

$$\begin{aligned} \frac{R(n+1)}{R(n)} &= \frac{1 + \frac{b_R + \delta}{n} + O\left(\frac{1}{n^2}\right)}{1 + \frac{b_R}{n} + O\left(\frac{1}{n^2}\right)} \\ &= \left(1 + \frac{b_R + \delta}{n} + O\left(\frac{1}{n^2}\right)\right) \cdot \left(1 - \frac{b_R}{n} + O\left(\frac{1}{n^2}\right)\right) \\ &= 1 + \frac{\delta}{n} + O\left(\frac{1}{n^2}\right) = 1 + \frac{\deg(N(n)) - \deg(U(n))}{n} + O\left(\frac{1}{n^2}\right). \end{aligned} \tag{44}$$

On the other hand

$$\begin{aligned} (n + \theta_k)^{e_k} &= n^{e_k} \cdot \left(1 + \frac{\theta_k}{n}\right)^{e_k} \\ &= n^{e_k} \cdot \left(1 + \frac{\theta_k e_k}{n} + \sum_{j=2}^{e_k} \binom{e_k}{j} \left(\frac{\theta_k}{n}\right)^j\right) \\ &= n^{e_k} \cdot \left(1 + \frac{\theta_k e_k}{n} + O\left(\frac{1}{n^2}\right)\right), \end{aligned} \tag{45}$$

therefore

$$\prod_{k=1}^{K}(n+\theta_k)^{e_k} = n^{\sum_{k=1}^{K} e_k} \cdot \left(1 + \frac{\sum_{k=1}^{K} \theta_k e_k}{n} + O\left(\frac{1}{n^2}\right)\right). \tag{46}$$

Finally according to (41), the expansion sought is obtained by the product of (44) and (46) times C. That is

$$\begin{aligned} r(n) &= C \cdot n^{\sum_{k=1}^{K} e_k} \cdot \left(1 + \frac{\sum_{k=1}^{K} \theta_k e_k}{n} + O\left(\frac{1}{n^2}\right)\right) \\ &\quad \cdot \left(1 + \frac{\deg(N(n)) - \deg(U(n))}{n} + O\left(\frac{1}{n^2}\right)\right) \\ &= C \cdot n^{\sum_{k=1}^{K} e_k} \left(1 + \frac{\sum_{k=1}^{K} \theta_k e_k + \deg(N(n)) - \deg(U(n))}{n} + O\left(\frac{1}{n^2}\right)\right) \end{aligned} \tag{47}$$

from which one easily read off the data of the theorem. □

The first two relations in this theorem show that ν and b can be found directly from a Pochhammer part ratio. Indeed, observe that modulo $\mathbb{Z}$, the second relation of the theorem reads as

$$b = \sum_{k=1}^{K} \theta_k \, e_k. \tag{48}$$

The third relation will be considered later in this subsection. The next step is then to find candidates for ν and b from the ratios in (39). These can be found easily; for example $(n+1)/(n+1/2)^2 = n^{-1}(1 - 1/(4n^2) + O(1/n^3))$ and therefore $\nu = -1$ and $b = -1$ (modulo $\mathbb{Z}$). As mentioned earlier, the map $y \mapsto y - \lfloor y \rfloor - 1$ is used to find the correspondence of y modulo $\mathbb{Z}$ in $\mathbb{I}$.

Next, we explain how the local types of hypergeometric term solutions of (29) are computed. This step is considered with the highest priority in our algorithm, because if the set of local types of a given holonomic recurrence equation is empty, then there is no hypergeometric term solution over the considered field.

For this step, van Hoeij's algorithm uses the Newton polygon of the difference operator (see [19, Section 3]). However, we proceed differently. Our idea is to rewrite (29) for ratios of hypergeometric term solutions, substitute (40) inside, and compute the asymptotic expansion of the nonzero side to find equations for the local types by equating the result to 0. This process is the same Petkovšek used to develop its algorithm Poly (see [12, Algorithm Poly]).

Let a_n be a hypergeometric term solution of (29) such that $a_{n+1} = r(n)a_n$ for a rational function r. We can write the equation for $r(n)$ as

$$\sum_{i=0}^{d} P_i \prod_{j=0}^{i-1} r(n+i) = 0. \tag{49}$$

We assume

$$r(n) = c \cdot n^{\nu} \cdot \left(1 + O\left(\frac{1}{n}\right)\right) \tag{50}$$

and we substitute this in (49). Similarly as we did in the proof of Theorem 8, we make computations that yield the possible values of ν and c. If such values are found, say $(\nu_{\text{cand}}, c_{\text{cand}})$, then we rewrite $r(n)$ as

$$c_{\text{cand}} \cdot n^{\nu_{\text{cand}}} \cdot \left(1 + \frac{b}{n} + O\left(\frac{1}{n^2}\right)\right) \tag{51}$$

and we make new computations to find b.
Summarized, our procedure to find local types (ν, c, b) of hypergeometric term solutions of (29) consists in the following items:

1. we compute the possible values for ν;
2. for each value of ν,
   - 2-a we compute possible values for c,
   - 2-b for each value found for c, we use ν and c to compute the possible values for b;
   - 2-c for each value found for b, (ν, c, b) constitutes a local type of a hypergeometric term solution of (29).

Let us now explain how each value is computed.

- Computing ν:

  Substitute (50) in (49) gives the following terms on the left-hand side

  $$c^i \cdot n^{i\cdot\nu} \cdot P_i \cdot \left(1 + O\left(\frac{1}{n}\right)\right), \quad (0 \leqslant i \leqslant d) \tag{52}$$

  which is equivalent to

  $$l_i \cdot c^i \cdot n^{i\cdot\nu + \deg(P_i)} \cdot \left(1 + O\left(\frac{1}{n}\right)\right), \quad (0 \leqslant i \leqslant d) \tag{53}$$

where l_i denotes the leading coefficient of P_i. Since we are dealing with an equality with right-hand side 0, the terms having the highest power of n in the asymptotic expansion of the equation left-hand must be zero. However this is only possible if a term of the form (53) has the same power of n with some other terms so that they add to 0. Therefore we deduce that possible candidates for ν are integer solutions of linear equations coming from equalities of powers of n for two different terms of the form (53). That is for $0 \leqslant i \neq j \leqslant d$, we have the equation

$$i \cdot \nu + \deg(P_i) = j \cdot \nu + \deg(P_j) \tag{54}$$

and therefore a possible value for ν is

$$\nu_{i,j} = \frac{\deg(P_j) - \deg(P_i)}{i - j}, \tag{55}$$

if the computed value is an integer.

We then compute $\binom{d}{2}$ such values for (29) and keep the integers. Note that two different couples (i, j) and (i', j') may give the same value for ν, meaning that the corresponding addition to zero involves all their constant coefficients. This is the point of the next item.

- Computing c:

  Assume that we have found a value $\nu_{i,j} \in \mathbb{Z}$ corresponding to k terms in the equation (49) with indices $0 \leqslant u_1 \neq u_2 \neq \ldots \neq u_k \leqslant d$. Then from (53) one easily see that a candidate for c is a solution of the polynomial equation

  $$l_{u_1} \cdot c^{u_1} + l_{u_2} \cdot c^{u_2} + \cdots + l_{u_k} \cdot c^{u_k} = 0. \tag{56}$$

  Since the corresponding terms must add to zero in the asymptotic expansion, their leading coefficients must equal zero. Note that (56) is solved over the considered field $\mathbb{K}$.

  Thus, for each value $c_{i,j} \in \mathbb{K}$ which is a zero of (56) for a given $\nu_{i,j}$, $(\nu_{i,j}, c_{i,j})$ is already a possible couple to be completed to get the local type of a hypergeometric term solution of (29).

- Computing b:

  For a computed couple $(\nu_{i,j}, c_{i,j})$ as explained above, we rewrite $r(n)$ as

  $$c_{i,j} \cdot n^{\nu_{i,j}} \cdot \left(1 + \frac{b}{n} + O\left(\frac{1}{n^2}\right)\right), \tag{57}$$

with unknown b.

After substituting (57) in (49) and computing again the asymptotic expansion, terms with highest powers of n add to zero, and therefore the left-hand side of the resulting equation must have a leading term with coefficient as a polynomial in the variable b. Since that polynomial must be zero, the possible values for b are its roots. This can be done by computing asymptotic expansion and solving the coefficients equal to zero for the unknown b. Finally if we find values for $b \in \mathbb{K}$ then we have found for each b a local type (ν, c, b) of a possible hypergeometric term solution of (29) over $\mathbb{K}$.

Thus we get the following algorithm.

Algorithm 3 Compute local types of all hypergeometric term solutions of (29)

Input: Polynomials

$$P_i(n) \in \mathbb{K}[n], i = 0, \ldots, d \mid P_d(n) \cdot P_0(n) \neq 0$$

Output: The set of all local types of hypergeometric term solutions of the holonomic RE

$$\sum_{i=0}^{d} P_i(n) a_{n+i} = 0.$$

1. Set $L = \{\}$.
2. For all pairs $\{i, j\} \in \{0, 1, \ldots, d\}$, compute

$$\nu_{i,j} = \frac{\deg(P_j) - \deg(P_i)}{i - j}. \tag{58}$$

3. For each integer $\nu_{i,j}$ computed in (58), compute the set of solutions in $\mathbb{K}$, say $S_{c,i,j}$, of the polynomial equation

$$l_{u_1} \cdot c^{u_1} + l_{u_2} \cdot c^{u_2} + \cdots + l_{u_j} \cdot c^{u_k} = 0, \tag{59}$$

where $l_{u_1}, l_{u_2}, \ldots, l_{u_k}$ are the leading coefficients of the polynomials $P_{u_1}, P_{u_2}, \ldots, P_{u_k}$, $0 \leqslant u_1 \neq u_2 \neq \ldots \neq u_k \leqslant d$ satisfying (58) for the same integer $\nu_{i,j}$.

Algorithm 3 Compute the local types of all hypergeometric term solutions of (29)

3. (a) For each element $c_{i,j}$ of $S_{c,i,j}$ set

$$r(n) = c_{i,j} \cdot n^{\nu_{i,j}} \cdot \left(1 + \frac{b}{n}\right). \tag{60}$$

   (b) Compute the coefficient $T_{i,j}(b)$ of the first non-zero term of the asymptotic expansion of

$$\sum_{i=0}^{d} P_i \prod_{j=0}^{i-1} r(n+i). \tag{61}$$

   (c) Solve $T_{i,j}(b) = 0$ in $\mathbb{K}$ for the unknown b and define $S_{b,i,j}$ to be the set of solutions.

   (d) For each element $b_{i,j} \in S_{b,i,j}$, add the triple $(\nu_{i,j}, c_{i,j}, b)$ to L.

4. Return L.

Theorem 9. *Algorithm* 3 *finds all the local types* (ν, c, b) *of hypergeometric term solutions of* (29).

Remark 10.

- *When extension fields are allowed, Algorithm* 3 *can be used to bound the degree of such extensions from the computation of* c*. However, this is not always enough because the singularities of the linear operator may define a larger bound. Heuristic tests are often used to decide on how to fix the bound of algebraic extensions. More theoretical details on dealing with algebraic extensions are described in [5, Section 8].*

- *Another important notice about extension fields is a property similar to the conjugate root theorem (see [5, Lemma 3]). Indeed, when a local type* (ν, c, b) *where* c *is defined over an extension field leads to a basis of hypergeometric term solutions, say* B_c*, then local types corresponding to conjugates of* c *lead to bases of same dimensions as* B_c*, that only differ from* B_c *by conjugations of* c*. Therefore an important efficiency is gained by using this property. This is used in our implementations to reduce computations of conjugate solutions into a single one.*

- *We mention that computations of Algorithm* 3 *can sometimes be used to reduce the number of iterations in the implementation. Indeed, when two linearly independent*

hypergeometric term solutions have the same local type, Algorithm 3 *computes it at least twice. Therefore collecting local types in a list might be advantageous, so that when a basis of hypergeometric terms corresponding to a particular local type is found, the latter is discarded from the list of local types. This is a useful tool when the number of computed local types (with repeated values) is less than the order of the given holonomic recurrence equation.*

Thus any ratio candidates whose local type is not in the list of local types (deprived of values for c) should not be used in further steps. We implemented a Maxima function `localtype(L,n)` which takes the polynomial coefficients of a holonomic recurrence equation in `L` in the variable `n` and returns a list of triples $[\nu, c, b]$. Applying it for (RE) yields

```
(% i1) localtype(expand(REcoeff(RE,a[n])),n);
```

$$[[-2\,,-1\,,-1]\,,[-1\,,1\,,-1]\,,[0\,,-1\,,-1]\,,[0\,,1\,,-1]]. \qquad (\%\ \text{o1})$$

The command `REcoeff` is our code to collect coefficients, which are expanded afterward using the Maxima command `expand`. From the obtained output it follows that $1/(n+1)^3$ and $(n+1)$ should also be removed from potential ratios of Pochhammer parts in (39). Note, however, that it is from the computed local types that we get the possible values for C in (36) according to the third relation in Theorem 8. These will be used in the next step together with their corresponding Pochhammer part ratios.

4.3. Rational part of hypergeometric terms

The algorithm goes further in filtering the set of Pochhammer part ratios. Indeed, once we have found all those better candidates for ratios of Pochhammer parts, we need to use again the second Fuchs relation from Theorem 8 in order to find $\delta = \deg(N(n)) - \deg(U(n))$, where $N(n)$ and $U(n)$ are the numerator and the denominator of R in (37). In fact, since we have found values for b and its possible ratio candidates, which means that we can compute $\sum_{k=1}^{K} \theta_k \cdot e_k$, we therefore deduce that these candidates are valid if and only if they satisfy

$$\delta = b - \sum_{k=1}^{K} \theta_k \cdot e_k \in \mathbb{Z}. \qquad (62)$$

In this case the verification of ratios of Pochhammer parts for the value of b should consist in checking if the difference $b - \sum_{k=1}^{K} \theta_k \cdot e_k$ is an integer.
However, (62) can be used in the algorithm only if b is not computed modulo $\mathbb{Z}$. Another approach is to use an asymptotic expansion. Since now we have the ratios with their

corresponding values of c, according to (37) we can write

$$r(n) = \frac{R(n+1)}{R(n)} \cdot c \cdot \frac{h(n+1)}{h(n)}. \tag{63}$$

Moreover

$$\frac{R(n+1)}{R(n)} = 1 + \frac{\delta}{n} + O\left(\frac{1}{n^2}\right), \tag{64}$$

where δ is as in (62). Thus the asymptotic expansion of (49) (left-hand side) with $r(n)$ used by combining (63) and (64) must have a leading term as a polynomial coefficient in the variable δ. Therefore the values of δ are integer roots (if there are some) of that polynomial. If there are not such roots, then the rational function $c \cdot h(n+1)/h(n)$ is removed from the potential Pochhammer parts of (29). Our implementation uses this second approach.

Mostly after this step the number of Pochhammer part ratios of (29) is considerably reduced or equal to the exact number of hypergeometric term solutions.

Now, it only remains to find the rational function R in (36) whose a holonomic recurrence equation can be easily computed. Let $c \cdot h(n+1)/h(n)$ be one of the remaining ratios times its corresponding c for the local type. Then the recurrence equation

$$\sum_{i=0}^{d} P_i \cdot R(n+i) \cdot c^{n+i} \cdot \frac{h(n+1+i)}{h(n+i)} = 0, \tag{65}$$

is an equation for the unknown rational function $R(n)$ that we can easily modify to a holonomic recurrence equation. However, there is no need to use a complete algorithm for computing rational solutions of holonomic recurrence equations. Indeed, since we already have the difference between the degrees of the numerators and the denominators of rational solutions of (65), it is enough to use an algorithm that computes a universal denominator[7] $U(n)$ of all rational solutions of (65), and use δ or its maximum value (for the second approach we proposed) (see (62)) to compute a degree bound $\delta + \deg(U(n))$ for the degrees of the corresponding numerators. Substituting $N(n)/U(n)$ in (65) where $N(n)$ is an arbitrary polynomial of degree $\delta + \deg(U(n))$ results in a linear system in the coefficients of the arbitrary polynomial $N(n)$. Finally solving that system gives a basis of all the rational functions $R(n) = N(n)/U(n)$ sought.

Regarding the computation of a universal denominator, Abramov has proposed most key results for that purpose (see [2, 1, 3]). A crucial step in Abramov's original algorithm is

[7] A universal denominator of rational solutions of a holonomic recurrence equation is a polynomial that is divisible by all the denominators of rational solutions of that holonomic equation [1].

to compute the dispersion set of two polynomials[8]. The dispersion set can efficiently be obtained from full factorization as described in [11]. We use this method in our implementation of Abramov's algorithm to compute universal denominators. As a little story, note that we have found neither in Maxima nor in Maple a satisfactory (in terms of efficiency) implementation for computing dispersion sets. Therefore we implemented the algorithm in [11] and added it as a by-product to our `FPS` package. This will be used in Maple starting from the 2022 release.
We think this last step of computing $R(n)$ might sometimes make a difference in efficiency between our algorithm and van Hoeij's original version. In his approach, van Hoeij uses a special algorithm from his idea of finite singularities to determine $R(n)$ (see [18]). Though we mentioned that a complete algorithm for that purpose is not necessary, the algorithm in [18] is sometimes suitable with the computations in [19]. However, comparisons in [3] show that using our approach or van Hoeij's at this step does not guarantee the efficiency gain of one over the other.

4.4. Our algorithm

We can now present the complete algorithm of this paper.

Algorithm 4 Compute hypergeometric term solutions of (29)

Input: Polynomials

$$P_i(n) \in \mathbb{K}(n), i = 0, \ldots, d, \mid P_d(n) \cdot P_0(n) \neq 0.$$

Output: A basis of hypergeometric term solutions of the holonomic recurrence equation

$$\sum_{i=0}^{d} P_i(n) a_{n+i} = 0 \tag{66}$$

over $\mathbb{K}$.

1. Set $H = \{\}$.
2. Use Algorithm 3 to compute the set $\mathcal{L}$ of all local types at infinity of hypergeometric term solutions of (66).
3. If $\mathcal{L} = \emptyset$, then stop and return H.

---

[8]The dispersion set of $A(n)$ and $B(n)$ can be defined as the set of all non-negative integer roots of the resultant polynomial of $A(n)$ and $B(n+h)$ in the variable h.

Algorithm 4 Compute hypergeometric term solutions of (29)

4. Construct the set of couple (numerator, denominator)

$$\mathcal{P} := \Bigg\{ (p(n), q(n)) \in \mathbb{K}[n]^2 \ : \ p(n) \text{ and } q(n) \text{ are monic factors modulo } \mathbb{Z}$$

$$\text{with roots real parts in } [-1, 0) \text{ of } P_0(n-1) \text{ and } P_d(n-d) \text{ respectively} \Bigg\}, \quad (67)$$

   for ratio candidates of Pochhammer parts.

5. Remove from $\mathcal{P}$ all couple whose $p(n)$ exponents are less than the minimum multiplicity of the corresponding root modulo $\mathbb{Z}$ in the trailing polynomial coefficient $P_0(n)$. Similarly, clear $\mathcal{P}$ by the same consideration for $q(n)$ exponents and the leading polynomial coefficient $P_d(n)$. Finally substitute each remaining couple $(p(n), q(n))$ in $\mathcal{P}$ by $\frac{p(n)}{q(n)}$.
6. Fix the bound of algebraic extensions and remove elements in $\mathcal{P}$ and $\mathcal{L}$ that have a larger algebraic degree.
7. Construct the set F_1 of $c \cdot r$, $r \in \mathcal{P}$ such that $c \cdot r$ has its local type at infinity as an element of $\mathcal{L}$.

$$F_1 := \left\{ c \cdot r \ : \ r = n^{\nu_r} \left(1 + \frac{b_r}{n} + O\left(\frac{1}{n^2} \right) \right) \in \mathcal{P} \text{ and } (\nu_r, c, b_n) \in \mathcal{L} \right\}. \quad (68)$$

8. Set $F_2 := \{\}$. For each element $f(n)$ of F_1
   (a) Compute a recurrence equation, say E_f with the coefficients

$$P_i \cdot \prod_{j=0}^{i} f(n+i), \ i = 0, \ldots, d, \quad (69)$$

   for the rational function $R(n)$ in (37) of the possible hypergeometric term solutions.
   (b) Substitute the terms $R(n+i)$ by $(1 + \frac{\delta}{n+i})$, $i = 0, \ldots, d$, in E_f and compute the coefficient of the leading term of the asymptotic expansion of the left hand side of E_f, say $Q_f(\delta)$.
   (c) Compute the set S_{δ_f} of integer roots of $Q_f(\delta)$.
   (d) If $S_{\delta_f} = \emptyset$ then $f(n)$ is discarded.

Algorithm 4 Compute hypergeometric term solutions of (29)

8. (e) Else set $\delta_f := \max(S_f)$, rewrite E_f in a holonomic form and add $(f(n), \delta_f, E_f)$ in F_2.
9. If $F_2 = \emptyset$ then stop and return H.
10. For each $(f(n), \delta_f, E_f) \in F_2$
    (a) Compute the universal denominator $U_f(n)$ of rational solutions of E_f by using the approach in [11] to find the needed dispersion set.
    (b) Update E_f as E'_f with $U_f(n)$ to get a holonomic recurrence for numerators of rational solutions of E_f.
    (c) Set $d_{N_f} := \deg(U_f(n)) + \delta_f$, and find a basis of polynomial solutions of degree at most d_{N_f} of E'_f.
    (d) Use Algorithm 2 to compute $h_f(n) = \prod_{k=0}^{n-1} f(k)$.
    (e) For each $N_f(n) \in S_{N_f}$ add $\frac{N_f(n)}{U_f(n)} \cdot h_f(n)$ to H.
11. Return H

We implemented Algorithm 4 in Maxima as `HypervanHoeij(RE,a[n],[K])`, with the default value `Q` (for rationals[9]) for `K` representing the field where solutions are computed. One must specify `C` for `K` to allow computations over extension fields of $\mathbb{Q}$. Using this implementation to solve (RE) yields

```
(% i1) HypervanHoeij(RE,a[n]);
```

Evaluation took 0.2970 seconds (0.3020 elapsed) using 96.149MB.

$$\left\{(n+1)\,(n+2)\,(n+3)\,,\frac{(-1)^n}{n}\,,\frac{1}{n\,!}\,,\frac{(-1)^n\,4^{2n}\,n\,!^2}{(2n)\,!^2}\right\}, \qquad (\%\ \text{o1})$$

with timing (allowed with the Maxima command `showtime`) 0.2970 second and 96.149 MB memory used. In Maple, we implemented our algorithm as `rectohyperterm(RE,a(n))`. Let us do the same computation with our Maple implementation and `LREtools[hypergeomsols]`.

```
> RE:=FPS[sumhyperRE]([binomial(n+3,n),1/n!,(-1)^n/n
  ,(-1)^n/pochhammer(1/2,n)^2],a(n)):
```

[9] Rationally valued, not symbolically rational: a parameter declared as rational is not considered as such in the implementation.

```
> Usage(FPS[rectohyperterm](RE,a(n)))
memory used=12.92MiB, alloc change=0 bytes, cpu time=219.00
   ms, real time=223.00ms, gc time=0ns
```

$$\left\{\frac{1}{n!}, \frac{(-1)^n}{n}, (n+3)(n+2)(n+1), \frac{n!^2\,(-16)^n}{(2n)!^2}\right\} \tag{70}$$

```
> Usage(LREtools[hypergeomsols](RE,a(n),\{\},output=
    basis))
memory used=20.12MiB, alloc change=25.99MiB, cpu time
   =375.00ms, real time=337.00ms, gc time=156.25ms
```

$$\left[n^3+6n^2+11n+6, \frac{(-1)^n}{n}, \frac{1}{\Gamma(n+1)}, \frac{(-1)^n}{\Gamma\left(\frac{1}{2}+n\right)^2}\right] \tag{71}$$

The `Usage` command from the `CodeTools` package is used to display timings and memory used. Here one can see the advantage of having an implementation that uses extension fields from user specifications. Allowing extension fields for this example unnecessarily increases the timing (will be the same as for `LREtools[hypergeomsols]`) of computations since (RE) is of order 4 and we already have 4 hypergeometric term solutions in the output.

Next, we give examples to show how conjugate hypergeometric terms are computed in our Maple implementation. To allow computations over algebraic extension fields with `rectohyperterm`, the third argument to specify is `complex`.

```
> RE:=FPS[sumhyperRE]([I^n/n!,(-I)^n/n!,n!*(1+I*sqrt
    (7))^n/k^n,n!*(1-I*sqrt(7))^n/k^n],a(n)):
> S:=FPS[rectohyperterm](RE,a(n),complex)
```

$$S := \left\{\frac{RootOf\left(\_Z^2+1\right)^n}{n!}, RootOf\left(k^2\_Z^2-2k\_Z+8\right)^n n!\right\} \tag{72}$$

One can recover the four hypergeometric term solutions by applying the Maple command *allvalues*.

```
> map(allvalues,S)
```

$$\left\{\frac{(-\mathrm{I})^n}{n!}, \frac{\mathrm{I}^n}{n!}, \left(\frac{2\left(\frac{1}{2}-\frac{\mathrm{I}\sqrt{7}}{2}\right)}{k}\right)^n n!, \left(\frac{2\left(\frac{1}{2}+\frac{\mathrm{I}\sqrt{7}}{2}\right)}{k}\right)^n n!\right\} \tag{73}$$

The latter is directly obtained using the internal command as shown below.

```
> LREtools[hypergeomsols](RE,a(n),{},output=basis)
```

$$\left[\frac{\mathrm{I}^n}{\Gamma(n+1)}, \frac{(-\mathrm{I})^n}{\Gamma(n+1)}, \left(\frac{1+\mathrm{I}\sqrt{7}}{k}\right)^n \Gamma(n+1), \left(-\frac{\mathrm{I}\sqrt{7}-1}{k}\right)^n \Gamma(n+1)\right] \quad (74)$$

However, in certain cases this direct method complicates the outputs for further use (see [15]) because it reconstructs the solutions itself. In the following example, the output given by `LREtools[hypergeomsols]` contains the one of `FPS[rectohyperterm]` which is enough to recover the hypergeometric term solutions sought.

```
> S:=[solve(z^3+7*z^2+z+28,z)]:
> RE:=FPS[sumhyperRE]([S[1]^n,S[2]^n,S[3]^n],a(n))
```

$$RE := 28a(n) + a(n+1) + 7a(n+2) + a(n+3) = 0 \quad (75)$$

```
> LREtools[hypergeomsols](RE,a(n),{},output=basis)
```

$$\left[\left(-7 - RootOf\left(\_Z^3 + 7\_Z^2 + \_Z + 28\right)\right.\right.$$
$$- RootOf\left(\_Z^2 + \left(7 + RootOf\left(\_Z^3 + 7\_Z^2 + \_Z + 28\right)\right)\_Z\right.$$
$$\left.\left. + RootOf\left(\_Z^3 + 7\_Z^2 + \_Z + 28\right)^2 + 7RootOf\left(\_Z^3 + 7\_Z^2 + \_Z + 28\right) + 1\right)\right)^n, RootOf\left(\_Z^3\right.$$
$$\left. + 7\_Z^2 + \_Z + 28\right)^n, RootOf\left(\_Z^2 + \left(7 + RootOf\left(\_Z^3 + 7\_Z^2 + \_Z + 28\right)\right)\_Z\right.$$
$$\left. + RootOf\left(\_Z^3 + 7\_Z^2 + \_Z + 28\right)^2 + 7RootOf\left(\_Z^3 + 7\_Z^2 + \_Z + 28\right) + 1\right)^n\Big] \quad (76)$$

```
> FPS[rectohyperterm](RE,a(n),complex)
```

$$\left\{RootOf\left(\_Z^3 + 7\_Z^2 + \_Z + 28\right)^n\right\} \quad (77)$$

5. Some comparisons

Our Maple implementation of the given algorithm was tested on many recurrence equations. Regarding efficiency, the difference between our implementation and that of van Hoeij is in the order of milliseconds: for solutions over the rationals, our implementation generally gives a better efficiency; and for solutions over extension fields, van Hoeij's code

is generally faster, but in both cases, the timings are very closed. Our Maxima implementation usually comes third when we compare efficiencies, though there are some examples where this is not verified. However, we think that for comparisons involving implementations in Maple and Maxima or two different CASes in general, a first look must be taken at kernels and data structures of both systems. Without going into details on these computer science "buildings", we only mention that Maple has a C-based kernel whereas Maxima has a Common Lisp-based one, and this could make some differences in the speed of both systems. The reader can visit the programming website `https://open.kattis.com` to see how often C programs are the fastest.

The new Maple command `LREtools[RightFactors]` is a new implementation of van Hoeij which allows computation over algebraic extension fields on specific user demands. This means that the suspected algebraic numbers are specified directly as input. Hypergeometric term ratios are computed by this command as first-order right factors. This implementation is of course more efficient in most cases since the extension fields allowed are bounded by the user. However, from the input and output points of view, our implementation does not easily compare to this new command in terms of efficiency. We have made some computations where we observed that the difference of CPU times between both approaches could be interpreted by the fact that our approach further implements Algorithm 2 to find "simple" formulas.

We mention that our implementations, presented at the International Congress of Mathematical Software (ICMS) 2020 and the Maple Conference 2020, helped fix bugs in `LREtools[hypergeomsols]` on its Maple 2020 version (check the limits presented in [16], an old version of this paper). However, the extension of computations to symbolic functions still raises issues with the current `LREtools[hypergeomsols]` as shown below with Maple 2021.1.

```
> RE3:=FPS[sumhyperRE]([ln(x)^n,ln(x*y)^n],a(n))
```

$$RE3 := a(n)\ln(x)\ln(xy) + (-\ln(x) - \ln(xy))\,a(n+1) + a(n+2) = 0 \tag{78}$$

```
> LREtools[hypergeomsols](RE3,a(n),{},output=basis)
Error, (in mod/Normal/Factored) not implemented

> FPS[rectohyperterm](RE3,a(n),complex)
```

$$\{\ln(x)^n, \ln(xy)^n\} \tag{79}$$

We believe that this bug can be fixed as well. Apart from giving a survey, the point here is also to show how having our algorithm contributes to ensuring and presenting equivalences of theoretical arguments in [12, 19, 5], and improve cutting edge implementations.

Although van Hoeij's algorithm is mentioned at *this footnote link* [10] to a Mathematica webpage, the implementation in `Rsolve` remains quite slow: we computed the solutions of (RE) and got a much more complicated result after about 7 minutes. It is difficult to decide which algorithm is used as the code is hidden from users. Petkovšek's algorithm is the most popular implementation encountered in many CASes. Our result and its implementation bring Maxima to the top level in computing hypergeometric term solutions of holonomic recurrence equations.

Acknowledgment 1. *The author is grateful to Wolfram Koepf for his advice.*

[1] Abramov, S. A., Rational solutions of linear difference and q-difference equations with polynomial coefficients, Program. Comput. Softw. 25 (6) (1999) 369–374.

[2] Abramov, S. A., Barkatou, M. A., Rational solutions of first order linear difference systems, in: ISSAC, vol. 98, Editors: Weispfenning, Volker and Trager, Barry, Association for Computing Machinery, New York, 124–131, 1998.

[3] Abramov, S. A., Gheffar, A., Khmelnov, D., Rational solutions of linear difference equations: Universal denominators and denominator bounds, Program. Comput. Software 37 (2) (2011) 78–86.

[4] Barkatou, M. A., Duval, M., Sur les séries formelles solutions d'équations aux différences polynomiales, Ann. Inst. Fourier. 44 (2) (1994) 495–524.

[5] Cluzeau, T., van Hoeij, M., Computing hypergeometric solutions of linear recurrence equations, Appl. Algebra Engrg. Comm. Comput. 17 (2) (2006) 83–115.

[6] Duval, M., Lemmes de Hensel et factorisation formelle pour les opérateurs aux différences, Funkcial. Ekvac. 26 (3) (1996) 349–368.

[7] Geddes, K. O., Czapor, S. R., Labahn, G., Algorithms for Computer Algebra, Kluwer Academic Publishers, Massachusetts, ISBN 0-7923-9259-0, 1992.

[8] Koepf, W., Symbolic Computation of Formal Power Series With Macsyma, in: Functional Analytic Methods In Complex Analysis And Applications To Partial Differential Equations, Editors: Tutschke, Wolfgang and Mshimba, Ali, World Scientific, Singapore, New Jersey, London, Hong Kong, 306–328, 1995.

[10] `https://reference.wolfram.com/language/tutorial/SomeNotesOnInternalImplementation.html`

[9] Koepf, W., Hypergeometric Summation, An Algorithmic Approach to Summation and Special Function Identities, Springer-Verlag, London, 2nd edn., 2014.

[10] Koepf, W., Computer Algebra: An Algorithm-Oriented Introduction, Springer Nature, Switzerland, 2021.

[11] Man, Y.-K., Wright, F. J., Fast polynomial dispersion computation and its application to indefinite summation, in: ISSAC, Editor: MacCallum, Malcolm, Association for Computing Machinery, New York, 175–180, 1994.

[12] Petkovšek, M., Hypergeometric solutions of linear recurrences with polynomial coefficients, J. Symb. Comput. 14 (2-3) (1992) 243–264.

[13] Salvy, B., Zimmermann, P., GFUN: a maple package for the manipulation of generating and Holonomic functions in one variable, [Research Report] INRIA, lRT-0143, 1992.

[14] Stanley, R. P., Differentiably finite power series, European J. Combin. 1 (2) (1980) 175–188.

[15] Teguia Tabuguia, B., Power Series Representations of Hypergeometric Types and Non-Holonomic Functions in Computer Algebra, Ph.D. thesis, University of Kassel, `https://kobra.uni-kassel.de/handle/123456789/11598`, 2020.

[16] Teguia Tabuguia, B., A variant of van Hoeij's algorithm to compute hypergeometric term solutions of holonomic recurrence equations, arXiv:2012.11513 [cs.SC] .

[17] Teguia Tabuguia, B., Koepf, W., Symbolic conversion of holonomic functions to hypergeometric type power series, Computer Algebra issue, Journal of Programming and Computer Software (to appear in February 2022) .

[18] Van Hoeij, M., Rational solutions of linear difference equations, in: ISSAC, Editors: Weispfenning, Volker and Trager, Barry, Association for Computing Machinery, New York, 120–123, 1998.

[19] Van Hoeij, M., Finite singularities and hypergeometric solutions of linear recurrence equations, J. Pure Appl. Algebra 139 (1-3) (1999) 109–131.