Accelerating indefinite summation: simple classes of summands

Eugene V. Zima

Abstract. We present the history of indefinite summation starting with classics (Newton, Montmort, Taylor, Stirling, Euler, Boole, Jordan) followed by modern classics (Abramov, Gosper, Karr) to the current implementation in computer algebra system Maple. Along with historical presentation we describe several "acceleration techniques" of algorithms for indefinite summation which offer not only theoretical but also practical improvement in running time. Implementations of these algorithms in Maple are compared to standard Maple summation tools.

Mathematics Subject Classification (2000). Primary 68R05; Secondary 39-04.

Keywords. Indefinite summation, basis conversion.

1. Introduction

Let $\mathbb{K}$ be a field of characteristic zero, x – an independent variable, E – the shift operator with respect to x, i.e., $Ef(x) = f(x+1)$ for an arbitrary function $f(x)$, $\Delta = E - 1$ – the difference operator with respect to x.

The problem of indefinite summation (anti-differencing) in general is: given a closed form expression $F(x)$ to find a closed form expression $G(x)$, which satisfies the first order linear difference equation

$$(E-1)G(x) = F(x). \tag{1}$$

If found, write $G(x) = \sum F(x) + c$, where c is an arbitrary constant or any periodic function of x with period 1. [1]

The process of inverting the difference operator was called in classic literature "integration" or "finite integration":

"The operation of integration is therefore by definition the inverse of the operation denoted by the symbol Δ. As such it may with perfect propriety be denoted by

This work is partially supported by NSERC.

[1]We will usually omit this constant term assuming it implicitly.

the inverse form Δ^{-1}. *It is usual however to employ for this purpose a distinct symbol,* Σ*, the origin of which, as well as of the term integration by which its office is denoted, it will be proper to explain."* — George Boole (1860) [8]

Since Boole does not provide an attribution of this notation it is appropriate to mention that notation Σ for indefinite summation was first introduced by Euler: *"Just as we used the symbol* Δ *to signify a difference, so we use the symbol* Σ *to indicate a sum."* — Leonard Euler (1755) [12]

The second related problem arises if $G(x)$ solving (1) cannot be found. In this case one can try to solve the *additive decomposition problem*: construct two closed form expressions $R(x)$ and $H(x)$, such that

$$F(x) = (E-1)R(x) + H(x), \tag{1a}$$

where $H(x)$ is simpler than $F(x)$ in some sense (or as variation, as simple as possible). An equivalent formulation of the additive decomposition problem is

$$\sum_x F(x) = R(x) + \sum_x H(x), \tag{1b}$$

and $R(x)$ is sometimes called the *summable* part of $f(x)$ and $H(x)$ – the *non-summable* part.

If a closed form $G(x)$ satisfying (1) exists one takes $R(x) = G(x)$ and $H(x) = 0$ as an answer to the additive decomposition problem. The measure of simplicity can be different for different classes of functions. For example if $F(x)$ is a rational function over $\mathbb{K}$ one requires both $R(x)$ and $H(x)$ to be rational with $H(x)$ having the denominator of the lowest possible degree. We intentionally do not define the notion of closed form expression here. When a particular class of expressions is considered (such as polynomials, proper rational functions, quasi-rational functions, hypergeometric terms), "closed form" will mean an expression from the class under consideration. In this paper following [27] we call *quasi-rational function* (resp. *quasi-polynomial*) a function of the form $\lambda^x \frac{f(x)}{g(x)}$ (resp. of the form $\lambda^x f(x)$), where $f(x), g(x) \in \mathbb{K}[x]$, $\lambda \neq 0$, $\lambda \neq 1$.

The history of indefinite summation naturally splits into "pre-computer algebra" and "computer-algebra" periods. Indefinite summation plays in discrete mathematics the same role as indefinite integration in calculus, and starts to grow from the same roots. The problem was considered in classical works of Newton, Bernoulli, Taylor, Nicole, Montmort, Stirling, Euler, Boole. Classics were concerned with the methods to obtain solution of (1) and did not consider (1a).

The history of the algorithmic treatment of indefinite summation problem oriented towards efficient implementation in computer algebra systems is also relatively long. Starting with a classical series of works by Abramov opened by his publication [1] (which gives not only a factorization-free algorithm for rational summation, but describes its Lisp implementation), works of Gosper [16, 17], Moenck [29], and Karr [22], the subject is well understood and developed.

It is even more surprising that some of the simplest and most understood algorithms for indefinite summation can still be improved, not only theoretically but also practically. We will give an overview of "basic" algorithms for indefinite summation, and will offer immediate improvements to them. This includes algorithms for (quasi-)polynomial and (quasi-)rational summation. Our "accelerated instances" of the algorithms covered here are all based on relatively old and simple ideas and can be generally described as "direct" algorithms, as opposed to "indirect" algorithms that for example involve reduction of the summation problem (using variations of the method of undetermined coefficients) to linear algebra. All our improvements are based on techniques used in computer algebra for years (basis conversion [14, 10], compact representation [15, 9], various forms of partial fraction decomposition [22, 24, 3], chains of recurrences [7, 41], etc.) We mention only those directly related to the content of this paper. These techniques were discovered, forgotten and rediscovered again, but still did not find their proper place in implementation of computer algebra systems [18, 14].

The results of classics were neglected or rediscovered many times. One particular example is a number of papers on computing sums of the form $\sum_x a^x x^n$ published in Fibonacci Quarterly over past decades in spite of the fact, that two algorithms for finding such sums are published more than 150 years ago in Boole's "Treatise on the calculus of finite differences" [8].

The paper is organized as follows. In Section 2 we describe what was known and used in pre-computer algebra era. Section 3 contains the history of recent 40 years of progress in indefinite summation (excluding hypergeometric summation) and related topics. Section 4 describes our improvements to basic algorithms for indefinite summation and provides some experimental results.

Note that we consider here only simple classes of summands. For example we do not discuss hypergeometric summation here (only mention Gosper's algorithm several times when it is applied to simple summand). In our references to Karr's works we do not discuss general problem of summation in the so-called ΠΣ-fields, but only refer to one of his basic algorithms for rational functions summation described in [24]. Karr's theory was recently generalized and streamlined in works of Schneider (see for example [36, 37, 38]), but these developments are out of the scope of this paper.

2. Works of classics

In this section we will try to describe how classics solve (1) using mainly books of Boole [8], Euler [12], description of works of Nicole, Taylor and Montmort from [39], and book of Jordan [19] which collect the the classic results on indefinite summation. The importance of operation of anti-differencing (a.k.a. indefinite summation, a.k.a. finite integration) was obvious to the classics. All techniques described in their work are immediately followed by examples of its use in applications.

One of the most important applications of the Calculus of Finite Differences is to the finite summation of series." — George Boole (1860) [8]

Euler (after introducing summation in Chapter 1 of [12]), spends whole Chapter 2 to demonstrate the use of indefinite summation in the theory of series.

There are two books which accumulate classic result on indefinite summation: Boole's (1860) [8] and Jordan's (1939) [19]. Boole's treatment of the problem is most condensed and deserves in our opinion to be presented here. What follows in different font is "almost" exact quotation from Boole's book section on "Integrable forms" (modulo small changes in notation, our footnotes and some shortcuts) [2].

1st Form. Factorial expressions of the form $x(x-1)\dots(x-m+1) = x^{(m)}$. We have

$$\Delta x^{(m+1)} = (m+1)x^{(m)};$$

$$\text{therefore}\ \ \Sigma x^{(m)} = \frac{x^{(m+1)}}{(m+1)} + C.$$

Thus also, if $u(x) = ax+b$, we have

$$\Sigma u(x)u(x-1)\dots u(x-m+1) = \frac{u(x)u(x-1)\dots u(x-m)}{(m+1)a} + C. \tag{2}$$

2nd Form. Rational and integral function of x. [3]

For, by Chap. II. Art. 5, any such function is reducible to a series of factorials of the preceding form, each of which may be integrated separately. We find for $\Sigma u(x)$ the general theorem

$$\Sigma u(x) = C + u(0)x + \Delta u(0)\frac{x^{(2)}}{1\cdot 2} + \Delta^2 u(0)\frac{x^{(3)}}{1\cdot 2\cdot 3} + \&c. \tag{3}$$

which terminates when $u(x)$ is rational and integral. [4]

It is obvious that a rational and integral function of x may also be integrated by assuming for its integral a similar function of a degree higher by unity but with arbitrary coefficients whose values are to be determined by the condition that the difference of the assumed integral shall be equal to the function given.

3rd Form. Factorial expressions of the form

$$\frac{1}{u(x)u(x+1)\dots u(x+m)},$$

where $u(x)$ is of the form $ax+b$.

We have corresponding to the above form of $u(x)$

$$\Delta\frac{1}{u(x)u(x+1)\dots u(x+m-1)} = \frac{-am}{u(x)u(x+1)\dots u(x+m)}.$$

$$\text{Hence}\ \ \Sigma\frac{1}{u(x)u(x+1)\dots u(x+m)} = -\frac{1}{am(u(x)u(x+1)\dots u(x+m-1))}. \tag{4}$$

It will be observed that there must be at least two factors in the denominator of the expression to be integrated. No finite expression for $\Sigma\frac{1}{ax+b}$ exists.

To the above form certain more general forms are reducible. Thus we can integrate any rational fraction of the form

$$\frac{\phi(x)}{u(x)u(x+1)\dots u(x+m)},$$

[2] This deserves to be presented here because of clear style and completeness of the content.
[3] This was commonly used term for polynomials in the 18th and 19th century.
[4] Here $\Delta^k u(0)$ is the k-th difference of $u(x)$ evaluated at 0.

$u(x)$ being of the form $ax+b$, and $\phi(x)$ a rational and integral function of x of a degree lower by at least two unities than the degree of the denominator. For, expressing $\phi(x)$ in the form

$$\phi(x) = Au(x)+Bu(x)u(x+1)+Cu(x)u(x+1)u(x+2)+\cdots+Eu(x)u(x+1)\cdots u(x+m-2)$$

A, B ... being constants to be determined by equating coefficients, or by an obvious extension of the theorem of Chap. II. Art. 5, we find

$$\Sigma\frac{\phi(x)}{u(x)u(x+1)\ldots u(x+m)} = A\Sigma\frac{1}{u(x+1)u(x+2)\cdots u(x+m)}+$$
$$+B\Sigma\frac{1}{u(x+2)u(x+3)\cdots u(x+m)}+\cdots+E\Sigma\frac{1}{u(x+m-1)u(x+m)}, \tag{5}$$

and each term can now be integrated by (4).

Again, supposing the numerator of a rational fraction to be of a degree less by at least two unities than the denominator, but intermediate factors alone to be wanting in the latter to give to it the factorial character above described, then, these factors being supplied to both numerator and denominator, the fraction may be integrated as in the last case.

As all that is known of the integration of rational functions is virtually continued in the two primary theorems of (2) and (4), it is desirable to express these in the simplest form. Supposing then $u(x)=ax+b$, let

$$u(x)u(x-1)\ldots u(x-m+1) = (ax+b)^{(m)},$$
$$\frac{1}{u(x)u(x+1)\ldots u(x+m-1)} = (ax+b)^{(-m)},$$

then

$$\Sigma(ax+b)^{(m)} = \frac{(ax+b)^{(m+1)}}{a(m+1)}+C, \tag{6}$$

whether m be positive or negative. [5] The analogy of this result with the theorem

$$\int(ax+b)^m = \frac{(ax+b)^{m+1}}{a(m+1)}+C,$$

is obvious.

4th Form. Functions of the form $a^x\phi(x)$ in which $\phi(x)$ is rational and integral.

Since $\Delta a^x=(a-1)a^x$, we have

$$\Sigma a^x = \frac{a^x}{a-1}+C.$$

To deduce $\Sigma a^x\phi(x)$ we may now employ either a method of integration by parts or a symbolical method founded upon the relations between the exponential a^x and the symbol Δ.

Applying integration by parts we have

$$\Sigma a^x\phi(x) = \frac{1}{a-1}\left\{\phi(x)a^x - a\Sigma a^x\Delta\phi(x)\right\}. \tag{7}$$

Thus the integration of $a^x\phi(x)$ is made to depend upon that of $a^x\Delta\phi(x)$; this again will by the same method depend upon that of $a^x\Delta^2\phi(x)$, and so on. Hence $\phi(x)$ being by hypothesis rational and integral, the process may be continued until the function under the

[5] Assuming $m\neq -1$ of course

sign Σ vanishes. This will happen after $n+1$ operations if $\phi(x)$ be of the nth degree; and the integral will be obtained in finite terms.[6]

But the symbolical method above referred to leads to the same result by a single operation [7]:

$$\Sigma a^x\phi(x) = \frac{a^x}{a-1}\left\{\phi(x) - \frac{a}{a-1}\Delta\phi(x) + \frac{a^2}{(a-1)^2}\Delta^2\phi(x) - \frac{a^3}{(a-1)^3}\Delta^3\phi(x) + \&c.\right\} + C. \tag{8}$$

The series within the brackets stops at the nth difference of $\phi(x)$, supposing $\phi(x)$ of the nth degree.

Several more historical comments are needed here.

The method of finding indefinite sum for polynomials is usually attributed to Stirling. However, according to [39]:

"While engaged in a study of the Methodus Differentialis of Jas. Stirling (1730) I have been struck by the fact that Nicole's papers on the same subject, printed in the Memoires de l'Academie Royale des Sciences (Paris), appear to form a fitting prelude to the work published by Stirling. The dates of Nicole's Papers are 1717, 1723, 1724, 1727, and it is almost certain that Stirling was well acquainted with their contents..." After describing four works of Nicole inspired by Brook Taylor (which give essentially the method of finite integration for polynomials and rational functions similar to one presented above), Tweeddie concludes

"From the foregoing it seems natural to infer that Nicole was the first to introduce the Inverse Factorial Series. His first Memoir bears the date 30th January 1717. In the Philosophical Transactions for the months of July, August, and September 1717 (No. 353), there was published a Memoir entitled De Seriebus Infinitis Tractatus. Pars Prima. Auctore Petro Remundo de Monmort, R.S.S. Una cum Appendice et Additamento per D. Brook Taylor, R.S. Sec.

From this Memoir it is clear that both Montmort and Brook Taylor had at the same time been busy with similar ideas."

Montmort in particular shows how to find $\Sigma\phi(x)/(x+n)...(x+(p-1)n)$ in the same way as Nicole, i.e., by writing $\phi(x)$ in the form $A_0 + A_1x + A_2x(x+n) + \ldots$. He also shows how to sum $\Sigma 1/(x+a)(x+b)(x+c)\ldots$, where $a, b, c\ldots$ are positive integers, by multiplying the numerator and denominator by the product $(x+a+1)(x+a+2)\ldots(x+b-1)(x+b+1)\ldots$, etc., and then proceeding as above.

He also gives an expansion of $1/(x+a)$ in the form $A/x + B/x(x+1) +$ etc., commonly described as Stirling's Series.

Tweedie continues:

Brook Taylor in his Appendix shows in a masterly manner how to deduce by his method of Increments the conclusions obtained by Montmort.

In the summation of $\Sigma\phi(x)/(x+a)(x+b)\ldots$ he points out that the degree of $\phi(x)$ must be less by 2 than that of the denominator. He then represents the fraction $\phi(x)/(x+a)(x+b)\ldots$ as a sum of partial fractions

$$\frac{A}{x+a} + \frac{B}{x+b} + \ldots \tag{9}$$

[6]This is essentially a very short description of recursive algorithm for summation of a quasi-polynomial, in which we need to assume $a \neq 1$.

[7]Details can be found in [8] on pages 52–53.

where

$$A + B + \cdots \equiv 0. \tag{10}$$

As we already mentioned in Introduction, Euler devoted the first chapter of his book [12] to the indefinite summation of polynomials and rational functions. In particular he writes:

35. We should observe this method carefully, since sums of differences of this kind cannot be found by the previous method. If the difference has a numerator or the denominator has factors that do not form an arithmetic progression, then the safest method for finding sums is to express the fraction as the sum of partial fractions. Although we may not be able to find the sum of an individual fraction, it may be possible to consider them in pairs. We have only to see whether it may be possible to use the formula

$$\Sigma\frac{1}{x+(n+1)\omega} - \Sigma\frac{1}{x+n\omega} = \frac{1}{x+n\omega}. \tag{11}$$

Although neither of these sums is known, still their difference is known.

36. In these cases the problem is reduced to finding the partial fractions, and this is treated at length in a previous book. [8]

He then gives several examples of summation, one of them being (in modern notation)

$$\Sigma\frac{3}{x(x+3)} = \Sigma\frac{1}{x} - \Sigma\frac{1}{x+3} = \Sigma\frac{1}{x+1} - \Sigma\frac{1}{x+3} - \frac{1}{x} =$$

$$= \Sigma\frac{1}{x+2} - \Sigma\frac{1}{x+3} - \frac{1}{x} - \frac{1}{x+1} = -\frac{1}{x} - \frac{1}{x+1} - \frac{1}{x+2}.$$

We note here that Euler's examples are always instructive, and one can easily extend them to more complicated summands, using relations similar to (11) such as

$$\Sigma\frac{a^{x+1}}{x+1} - \Sigma\frac{a^x}{x} = \frac{a^x}{x},$$

or

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

The last reference in this section is to the book of Jordan [19]. According to [29] "Jordan's book is more like a cookbook of approaches, rather than a rigorous algorithmic treatment..." and we disagree with this characteristic. Jordan's book provides even more details on the summation of rational functions than Boole's book.

[8] Here ω is an increment of independent variable x and can be replaced by 1.

3. Computer algebra era

Polynomials and quasi-polynomials are always summable. Computer algebra systems implementations of indefinite sums for polynomials and quasi-polynomials essentially adopted classic results without change (see for example [29]). One exclusion is probably a description of quasi-polynomial summation in [27], which sets up a triangular system of linear equations to find the coefficients of the result.

Intrigue starts with summation of rational functions. Summarizing above, pre-computer algebra classics had no problems with indefinite summation of rational functions provided the denominator is a product of linear factors. However, because of lack of efficient factorization algorithms, early computer algebra development was solving the problem in factorization-free manner.

Let $F = f/g \in \mathbb{K}(x)$, with $f, g \in \mathbb{K}[x] \setminus \{0\}$ coprime. Using division with remainder to split off the polynomial part, which can be summed with no problems, we may assume that $\deg f < \deg g = n$, so that F is *proper*.

The notion of *dispersion* plays a crucial role in the development of modern algorithms for indefinite summation. First introduced by Abramov in his classical work [1] as the maximal integer distance between roots of the denominator g of a reduced rational function $F = f/g \in \mathbb{K}(x)$, it has since been a key notion in several algorithmic developments. We denote dispersion of the input by ρ. Consider $\sum_x F$. If $\rho = 0$ then we take $R = 0$ and $H = F$ in (1a), see [1, 3]. In fact, the condition that the denominator of H has minimal degree is equivalent to the dispersion of H being zero. Now, let $\rho > 0$. One technical difficulty (which makes indefinite rational summation different from indefinite rational integration) is that the degree of the denominator of the indefinite sum can be proportional to ρ, whose value in turn can be exponentially large in the bit size of the summand.

For instance, if $F(x) = \frac{1}{x^2+\rho x}$, (with ρ natural number) then the solution (up to an additive constant) of the rational summation problem (1) is

$$G(x) = -\frac{1}{\rho}\left(\frac{1}{x} + \frac{1}{x+1} + \cdots + \frac{1}{x+\rho-1}\right),$$

whose numerator and denominator have degrees linear in ρ, and coefficients of bit-size linear in ρ.

We distinguish two kinds of dependency of the running time of a summation algorithm on dispersion ρ in cases when ρ is large:

- *essential* (non-removable) dependency, when an algorithm is at least linear in ρ^ϵ for some positive $\epsilon \leq 1$ and the expanded output has size also linear in ρ^ϵ;
- *non-essential* (potentially removable) dependency, when an algorithm is at least linear in ρ^ϵ, but the expanded output has size polynomial in the input size.

The essential dependency is a feature of the summation problem (output happens to be exponentially large in the input size) and non-essential dependency is a feature of some particular algorithm (output is small but the algorithm exhibits exponentially large intermediate expressions). Many of existing algorithms exhibit non-essential dependency of the running time on ρ which makes them unnecessarily slow for the cases of large ρ and small output size.

The history of progress in "modern" rational summation can be split in 3 periods with unusual gaps in between. The first algorithms were "unknown" for a decade to the computer algebra community at large. Activity in this research was largely driven by the needs in computer algebra systems development: indefinite summation of rational functions (or more generally – finding solutions of linear difference equations with polynomial coefficients) forms the basis for more advanced summation techniques [22]. The situation here is parallel to the situation in symbolic integration.

Consider two popular problems associated with the indefinite rational summation (as in [34]):

P1: For a given proper rational function $F(x)$ find a pair of rational functions $R(x)$ and $H(x)$, satisfying (1a) with $H(x)$ having denominator of minimal possible degree.

P2: For a given proper rational function $F(x)$ find a solution of problem **P1** with minimal possible degree of the denominator of the summable part $R(x)$, provided that $H(x) \neq 0$, i.e., $F(x)$ is *not rational summable.*

For illustration purposes we will compare behavior of different algorithms on two examples:

$$\sum_x \frac{-2\,x+999}{(x+1)\,(x-999)\,x\,(x-1000)} = \frac{1}{x\,(x-1000)}, \tag{12}$$

$$\sum_x \frac{x^3-1998\,x^2+996999\,x+999999}{(x+1)\,(x-999)\,x\,(x-1000)} = \frac{1}{x\,(x-1000)} + \sum_x \frac{1}{x}. \tag{13}$$

The dispersion of the summand in both of these examples is 1001.

3.1. Modern Classics

In his classical work [1] Sergei Abramov first introduced the notion of dispersion, and described an algorithm for rational indefinite summation. The algorithm was implemented in Lisp and used author's own polynomial arithmetic package. He later provided an algorithm to solve problem **P1** [2]. This result was somehow unknown for some period of time. Even in 1993 some authors state (see [25]) "At first glance, this improvement might seem to be of minor interest since Gosper's algorithm is not primarily intended for the special case of rational summation. But we have to stress that in many computer algebra systems it is the only summation algorithm available."

It is worthwhile to mention here the work of Moenck [29] in which he tries to generalize (5) to the case of denominators with factors of arbitrary degree. Moenck's algorithm was used in Maple for some period of time until some serious flaw was discovered in it (see [32]).

Abramov's work becomes popular and widely adopted by computer algebra systems two decades later.

During the decade of 1990s there were a number of algorithms and improvements developed, see for example [3, 27, 30, 32, 33, 40, 26]. In particular [32] gives a complete overview of these algorithms and improvements to them. All these algorithms carefully avoid factorization in $\mathbb{K}[x]$ and fall into one of the following two categories.

- **Iterative** (Hermite reduction analogous) algorithms will start with $R = 0$ and $H = F$ and decrease the dispersion of H by one at each iteration, reducing the non-rational part H and growing the rational part R. The number of iterations is equal to ρ, see [2].
- **Linear algebra based** (Ostrogradsky analogous) algorithms first build universal denominators u and v such that the denominator of R will divide u and the denominator of H will divide v. Then, the problem is reduced to solving a system of linear equations with size $\deg u + \deg v$, see [30, 3, 32]. Since $\deg u \geq \rho$ the choice of u of the lowest possible degree is obviously crucial here. The idea to build an universal denominator here is essentially the same as multiplying the numerator and denominator of the summand by missing factors in Boole's description above.

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

Consider examples (12) and (13). On the one hand, iterative algorithms will require about 1000 steps of polynomial gcd computations. On the other hand, the universal denominator constructed by the linear algebra based algorithms will have degree about 1000. In general, ρ may be as large as the magnitude of the trailing coefficient of g. Thus, the cost of the iterative and linear algebra based algorithms for computing a decomposition as in (1a) may be exponential in the size of the input.

Note here that Euler would not have problems with (12). After computing the partial fraction decomposition of the summand:

$$\frac{1}{1000(x-999)} - \frac{1}{1000\,(x-1000)} + \frac{1}{1000x} - \frac{1}{1000\,(x+1)},$$

and applying (11) to the first and the second pair of fractions, one obtains

$$\frac{1}{1000\,(x-1000)} - \frac{1}{1000x}.$$

This would probably take him less time manually than-computer based factorization-free implementation in early computer algebra systems.

3.2. Gosper's algorithm

In [17] Gosper described decision procedure for summability of hypergeometric term that can be used (and was used indeed according to [25]) to solve (1). Observe that it does not solve either of problems **P1** or **P2** when the input is not rational summable. That is, the algorithm is not applicable to the example shown in (13) since $H = 1/x \neq 0$. Another serious drawback is that Gosper's algorithm starts with computing of the so called Gosper form of $f(x+1)g(x)/g(x+1)f(x)$, and the cost of this step alone is proportional to the dispersion of the input. For example, Gosper form of the certificate of summand from (12) involves a polynomial of degree 999. After this form is computed Gosper's algorithm will look for a polynomial solution of the first-order difference equation with right-hand side being polynomial of degree about 1000, and use this solution and the right-hand side to write down the numerator and denominator of the indefinite sum, which have a huge common factor. It is obvious that this dependency of the running time of Gosper's algorithm on dispersion of the input is potentially removable (as output in (12) is small).

In the 1990s, a lot of effort was expended to improve the behavior of Gosper's algorithm on rational input. This led in particular to the introduction of the notion of Greatest Factorial Factorization [30], which plays an important role in computer algebra applications nowadays.

In spite of all improvements in Gosper's algorithm [25, 26, 30], the first step is still computing the Gosper-Petkovšek form with running time proportional to the dispersion ρ of the denominator $g(x)$. The only exception seems to be [40], which, when applied to rational summation, avoids computing the Gosper-Petkovšek form and solving the linear system corresponding to Gosper equation. However, as far as we can tell, this algorithm still involves a loop with ρ iterations, i.e., exhibits non-essential dependency of running time on the dispersion of the input.

3.3. Direct algorithms

Direct algorithms for indefinite summation of rational functions are based on partial fraction decomposition, compact representation of the result up to the very last step, and fast minimization of degree of the denominator in the summable part.

Following [30] we call two polynomials $u(x), v(x) \in \mathbb{K}[x]$ *shift equivalent* if there exists $h \in \mathbb{Z}$, such that $u(x+h) = v(x)$. It is clear that a set of distinct shift equivalent polynomials $u_1(x), \ldots, u_m(x)$ has a *smallest* element – a polynomial $v(x)$ such that all other polynomials are the result of a non-negative integer shift of $v(x)$, i.e., $u_i = v(x+h_i) = E^{h_i}v(x)$, where $h_i \geq 0, h_i \in \mathbb{Z}, i = 1, \ldots, m$.

Different forms of partial fraction decomposition of rational summand were considered in relation to the problems of indefinite rational summation. The three most popular forms can be expressed generally as follows. Any given proper reduced rational function $f(x)/g(x)$ $(f(x), g(x) \in \mathbb{K}[x])$ can be uniquely represented as

$$\frac{f(x)}{g(x)} = \sum_{i=1}^{v} \sum_{j=1}^{m_i} \sum_{k=0}^{n_{ij}} \frac{f_{ijk}}{E^k g_i^j}, \tag{14}$$

where:

(i) g_i are monic distinct factors, irreducible over $\overline{\mathbb{K}}$, or
(ii) g_i are monic distinct factors, irreducible over $\mathbb{K}$, or
(iii) g_i are monic distinct factors, shiftless over $\mathbb{K}$.

Here v is the number of different shift equivalence classes (components) in the denominator of the summand; m_i is the highest multiplicity of a factor in class i; n_{ij} – the largest shift of a factor of multiplicity j in class i. All polynomials f_{ijk} have $\deg f_{ijk} < \deg g_i$. This means in particular that in case **(i)** $\deg f_{ijk} = 0$ and $f_{ijk} \in \overline{\mathbb{K}}$.

For the simplicity of description of the direct algorithm, (14) can be rewritten in the form

$$\frac{f(x)}{g(x)} = \sum_{i=1}^{v} \sum_{j=1}^{m_i} M_{ij}(E) \frac{1}{g_i^j}, \tag{15}$$

where $M_{ij}(E)$ is a linear difference operator with coefficients in $\overline{\mathbb{K}}$ or in $\mathbb{K}[x]$. This representation is unique in each of the cases **(i)** – **(iii)**. If $\frac{f(x)}{g(x)}$ is summable, the summable part $R(x)$ can be also uniquely written in a form analogous to (15):

$$\sum_{i=1}^{v} \sum_{j=1}^{m_i} L_{ij}(E) \frac{1}{g_i^j}, \tag{16}$$

and therefore

$$(E-1)L_{ij}(E) = M_{ij}(E), \tag{17}$$

i.e., every operator $M_{ij}(E)$ in (15) is left-divisible by $E-1$. Let $M_{ij}(E) = a_p E^p + a_{p-1}E^{p-1} + \ldots + a_1 E + a_0$. Then the remainder from the left division of $M_{ij}(E)$ by $E-1$ is simply

$$r_{ij} = E^{-p} a_p + E^{-(p-1)} a_{p-1} + \ldots + E^{-1} a_1 + a_0. \tag{18}$$

The summability criterion states that the polynomials r_{ij} in (18) must be identically equal to zero for all i, j in order for the input rational function to be summable.

Observe that
(i) requires factorization of the denominator into linear factors,
(ii) requires factorization in $\mathbb{K}[x]$ but does not require computation of the dispersion of the denominators, and
(iii) does not require factorization in $\mathbb{K}[x]$ but does require the knowledge of the so-called dispersion set (the set of all integer distances between the roots of the denominator).

The first description of direct rational summation algorithm is due to Karr [21] (see also popular survey of Lafon [24], where above-mentioned summability criterion is formulated explicitly with proper reference to Karr's work). Because – at that time – factorization was considered time-consuming it was effectively forgotten, and factorization-free (gcd-based) algorithms were used in computer algebra systems for years. Note here that this criterion (known in the form (10) to classics, and given in the form $r_{ij} = 0$ with r_{ij} from (18) here and in Karr's work) was "rediscovered" again in 2000 (see in [28]).

Decomposition **(i)** was used in [3] to establish a summability criterion and to describe the structure of a universal denominator. This criterion is the same as

(10) since a linear difference operator with constant coefficients is left-divisible by $E-1$ if and only if the sum of coefficients is equal to 0. This is also equivalent to the classical condition of summability which is that the degree of the numerator is at least 2 less than the degree of the denominator.

Later the direct algorithm based on **(i)** was prototyped in Maple (see [11]). Note here that because of the way Maple represents full partial fraction decomposition (using notation of the sum over roots of irreducible polynomials) the prototype essentially worked with representation of summand **(ii)**.

Representation **(iii)** with a fast algorithm to compute dispersion set was used in [15]. It provided the first polynomial time rational function summability test, and avoided any intermediate expression swell. If the output is exponentially large in the input size, the only part of the algorithm that exponentially depends on the input size is writing the result in expanded form. This was the first rational indefinite summation algorithm with only essential dependency of the running time on dispersion of the input for the case of summable input.

However, when the input is not summable this algorithm can still exhibit non-essential dependency of the running time on ρ. The reason is that in this case the problem of additive decomposition has infinitely many solutions with different non-summable parts. Consider a simple example from [34] (Example 1). Here is a solution to additive decomposition problem

$$\sum_x \frac{x^2-100}{x\,(x+1)\,(x+100)} = \frac{2}{x} + \frac{1}{x+1} + \cdots + \frac{1}{x+99} + \sum_x \frac{1}{x}, \tag{19}$$

with the denominator of non-summable part of smallest possible degree. This is the form of the output that is returned by the algorithm from [15]. However, the same summation problem has another additive decomposition

$$\sum_x \frac{x^2-100}{x\,(x+1)\,(x+100)} = \frac{1}{x} + \sum_x \frac{1}{x+100}. \tag{20}$$

The difference of these decompositions is that the degree of the denominator of the summable part in (20) is minimal among all decompositions with denominator of the non-summable part of degree 1.

The problem of minimization of the denominator of the summable part was partially solved in [31], and [34] gives a complete solution. However it suffered from two major drawbacks:
- as a preliminary step it uses the classical algorithm from [3], which exhibits non-essential dependency of the running time on the dispersion of the input (apparently the author was unaware of [15]);
- the output of this preliminary step is “massaged” and transformed into the minimal form, however the size of the input of this step can already be exponential in the size of $F(x)$ even if the resulting minimal form is small.

This means that both steps of the algorithm from [34] exhibit non-essential dependency of the running time on the dispersion of the input, because in order to produce the minimal possible output (as in the right-hand side of (20)) the algorithm first builds the right-hand side of (19), and then post-processes it. The solution to the problem that is linear in the minimal possible output size was given in [44] (see also [43]). In [35] the author rediscovers Karr's rational summation algorithm [21] and gives its modification that minimizes the denominator of non-summable part in the case when input is non-summable without referring to Karr's works or at least to Lafon's survey [24].

It was finally shown in [43] that in the cases when the input is not summable, the choice of answer with minimal degree of the denominator of the summable part can be can be performed in no more than n steps.

4. Direct indefinite summation in practice

As far as we can judge, design and implementation of the indefinite summation algorithms in the computer algebra system Maple follows the pattern laid out in [6] (see Section 2.3 there). Although the flow chart looks reasonable, it has several flaws both in the design and in the implementation. Our main criticism is that Maple tries to apply the general hypergeometric treatment (Gosper's [17] or Abramov-Petkovšek's [5] algorithms) to the summand once it detects that the summand is not polynomial, quasi-polynomial or rational. There is a wide class of summation problems with quasi-rational summands that deserve specialized treatment. Theoretically Gosper's algorithm can be used for indefinite summation of polynomials and quasi-polynomials also, and it will succeed. It is a noble idea to have one algorithm for all problems, but for example Gosper's algorithm is not necessarily a good choice even to be applied to a rational summand, as it was shown many times (see, for example [15, 42]).

Example 1. Consider the following two summation problems:

$$\sum_x T(x) = \frac{5^x}{(x+1)(x+200)}, \tag{21}$$

where $T(x) = \dfrac{2(2x^2+401x+299)5^x}{(x+1)(x+2)(x+200)(x+201)}$ and

$$\sum_x \frac{\left(9\,x^4+1434\,x^3+70075\,x^2+1017440\,x-252800\right)5^x}{(x+40)\,(x+80)\,(x+79)\,(x+1)\,x} = \tag{22}$$

$$\frac{5^x\,(2\,x+79)}{(x+79)\,x} + \sum_x \frac{5^x}{x+40}$$

For the sum (21) the Gosper-Petkovšek form of the certificate of the summand consists of polynomials $Q(x) = 5(x+1)$, $R(x) = x+202$ and
$P(x) = \left(x^2+\frac{401}{2}x+\frac{299}{2}\right)(x+199)\,(x+198)\cdots(x+3)$.
This last polynomial has degree one less than the dispersion of the input (199 in this case), and is saturated. After finding a polynomial solution $U(x)$ (with degree

bound 199) of the Gosper equation

$$Q(x)U(x+1) - R(x-1)U(x) = P(x),$$

one forms the final result as

$$\frac{Q(x-1)U(x)}{P(x)} T(x) = \frac{5^x}{(x+1)\,(x+200)},$$

and most of the terms of $P(x)$ and $U(x)$ cancel.

For the sum (22) Gosper's algorithm will not return any answer (since the input is non-summable), so the general additive decomposition method from [5] is used. It is worthwhile to mention that this algorithm uses a number of steps at least linear in the dispersion of the input (which is 80 in this particular case), and returns the result, whose summable part has the numerator of degree 40, the denominator of degree 41, and the non-summable part is

$$\sum_x 9094947017729282379150390625\, \frac{5^x}{x+80}.$$

Unlike traditional algorithms, our direct algorithms are based on a change of representation of the summand.

4.1. Polynomials and quasi-polynomials

In the calculus of finite differences it is always advantageous to express polynomials by Newton's formula. – Charles Jordan (1939)

In case of indefinite summation of (quasi-)polynomials special representation means just change of basis. The change from monomial basis to the falling factorial basis is a common place in many sources to provide algorithms for polynomial summation, that involves either computation of Stirling numbers, or Bernoulli polynomials (see for example [24, 13]). However, the binomial basis $\binom{x}{i}, i = 0, 1, \ldots, n$ happens to be the most suitable in this context. Indeed, assume all polynomials are given in the binomial basis, i.e., a polynomial $P_n(x)$ of degree n is represented by a list $\varphi_0, \varphi_1, \ldots, \varphi_n \in \mathbb{K}$ such that

$$[\varphi_0, \varphi_1, \ldots, \varphi_n](x) = \varphi_0 \binom{x}{0} + \varphi_1 \binom{x}{1} + \ldots + \varphi_n \binom{x}{n} \tag{23}$$

(We will further omit mentioning explicit dependency of l.h.s. of x since most of the time the independent variable x is implicitly assumed.)

Addition, subtraction and multiplication by a constant in binomial basis are componentwise (similarly to the monomial basis). Note that $\varphi_0, \varphi_1, \ldots, \varphi_n$ in (23) are just finite forward differences (elements of the Newton series, or pure-plus chains of recurrences) of $P_n(x)$ taken at $x = 0$ with step 1 [7, 41]. Recall from [41] (or obtain in an elementary manner from the Pascal triangle definition) that

$$\begin{aligned} E[\varphi_0, \varphi_1, \ldots, \varphi_n] &= [\varphi_0 + \varphi_1, \varphi_1 + \varphi_2, \ldots, \varphi_{n-1} + \varphi_n, \varphi_n] \\ (E-1)[\varphi_0, \varphi_1, \ldots, \varphi_n] &= [\varphi_1, \varphi_2, \ldots, \varphi_n] \\ \sum_x [\varphi_1, \varphi_2 \ldots, \varphi_n] &= [\varphi_0, \varphi_1, \varphi_2, \ldots, \varphi_n] \end{aligned}$$

for an arbitrary constant φ_0. This suggests a constant time indefinite summation algorithm in binomial basis, which can be expressed in Maple (assuming that input polynomial is represented by the list of coefficients in binomial basis and summation variable is implicit) as:

```
poly_sum := y->[0,op(y)];
```

This is probably the shortest and fastest algorithm for indefinite summation, and it is not surprising that it was known long before the first computer algebra system was even prototyped (see the example on page 103 of [20], which is the third edition of the original book published in 1939). However we are not aware of any implementation of this algorithm in a computer algebra system. Although, even for manual manipulation of the problem the formula

$$\sum_x \binom{x}{m} = \binom{x}{m+1} + C \text{ is better than the popular } \sum_x x^{(m)} = \frac{x^{(m+1)}}{(m+1)} + C.$$

Just compare the right-hand sides to see how to save 6 symbols in the output. Note that in more general context the binomial basis was used in [4] where it is shown why the binomial basis is the basis of choice when solving linear difference equations with polynomial coefficients. However Maple standard summation routines ignore this fact.

If input is given in monomial basis, then the running time of summation is dominated by the time to convert the input to binomial basis and back. Note that the current algorithm for polynomial summation in Maple uses a formula based on Bernoulli polynomials and its complexity is at least quadratic in the degree of the summand.

Representation of polynomials in binomial basis is especially appealing, because it can be used as a standard dense representation for polynomials. A lengthy chain of computations involving ring operations, (nested) indefinite summation, differencing, etc. can be performed in this basis from the beginning. If output in standard basis is required, conversion may be performed when necessary. On one hand, all ring operations have the same asymptotic time complexity in both bases (see [14]). On the other hand, operations of differencing and indefinite summation have time complexity bounded by a constant (the best one can hope for), and shift is performed in linear time.

We strongly agree with the old comment of Jordan on importance of Newton's formula for the statistician:
This is not yet sufficiently recognized, since nearly always the statistician expands his polynomial in power series in spite of the fact that he is generally concerned with the differences and sums of his functions, so that he is obliged to calculate these quantities laboriously. In Newton's expansion they would be given immediately. (see [19])
This is especially true for Maple, because, for example, Newton interpolation routines return the resulting interpolating polynomial effectively in Newton form.

Consider now a quasi-polynomial with polynomial part represented in binomial basis:

$$q(x) = \lambda^x[\varphi_0, \varphi_1, \dots, \varphi_n],$$

where $\lambda \neq 1$, $\lambda \neq 0$. It is well known (see for example [27]) that the indefinite sum of $q(x)$ has polynomial part of the same degree n, i.e.:

$$\sum_x q(x) = r(x) = \lambda^x[\xi_0, \xi_1, \dots, \xi_n]. \tag{24}$$

Now,

$$Er(x) = \lambda^{x+1}[\xi_0 + \xi_1, \xi_1 + \xi_2, \dots, \xi_{n-1} + \xi_n, \xi_n] =$$

$$\lambda^x[\lambda\xi_0 + \lambda\xi_1, \dots, \lambda\xi_{n-1} + \lambda\xi_n, \lambda\xi_n],$$

$$(E-1)r(x) = \lambda^x[(\lambda-1)\xi_0 + \lambda\xi_1, \dots, (\lambda-1)\xi_{n-1} + \lambda\xi_n, (\lambda-1)\xi_n],$$

and application of the operator $(E-1)$ to l.h.s. and r.h.s. of (24) gives

$$[\varphi_0, \varphi_1, \dots, \varphi_n] = [(\lambda-1)\xi_0 + \lambda\xi_1, \dots, (\lambda-1)\xi_{n-1} + \lambda\xi_n, (\lambda-1)\xi_n],$$

i.e., the coefficients of the polynomial part of the indefinite sum can be computed as

$$\xi_n = \frac{\varphi_n}{\lambda-1}, \xi_i = \frac{\varphi_i - \lambda\xi_{i+1}}{\lambda-1}, i = n-1, n-2, \dots, 0.$$

This gives a linear time algorithm for indefinite summation of quasi-polynomials.

If the input is given in the monomial basis, then the running time of summation is again dominated by the time to convert the input to binomial basis and back. The current implementation in Maple is at least quadratic in the degree of polynomial part of the summand (see Subsection 4.3 for a comparison of timings).

4.2. Rational and quasi-rational functions

The safest method for finding sums is to express the fraction as the sum of partial fractions. Although we may not be able to find the sum of an individual fraction, it may be possible to consider them in pairs. — Leonard Euler (1742)

As a piece of computer algebra folklore we mention here that, to our knowledge, one of the most popular computer algebra systems has never used factorization-free algorithms for the rational summation. It always proceeded with partial fraction decomposition and used variations of Euler's simplification (11).

Contrary to this, Maple system for a long time had an obsession with factorization-free implementations of rational indefinite summation. In some versions of Maple this obsession has led to anecdotal situations. For example, Maple summation routine would compute the dispersion set of the denominator of the input summand using factorization, and then "forget" about the fact that the denominator was already factored to run factorization-free implementation of Abramov's algorithm, that requires dispersion set as the input.

Recall that $F = f/g \in \mathbb{K}(x)$, with $f, g \in \mathbb{K}[x] \setminus \{0\}$ coprime and $\deg f < \deg g = n$. Direct algorithms for indefinite summation of rational and quasi-rational functions are based on partial fraction decomposition, compact representation of the result up to the very last step, and fast minimization of degree of the denominator in the summable part. We show that in cases when the input is not summable, the choice of answer with minimal degree of the denominator of the summable part can be performed in a number of steps bounded by n.

Even though the direct algorithm for rational indefinite summation was described in many sources, we give a brief sketch here, using synthetic division arguments (as in (17)). Synthetic division provides a clear, undistorted explanation of the structure of both the summable and the non-summable parts, and it is worth considering since it is hidden (implicit) in both – the original works of Karr [21, 22, 23], and in [35]. Since the size of the output depends on the sparsity of the left quotient, and in order to describe the sparsity of a linear difference operator, we define the *support* of an operator $L = a_m(x)E^m + \cdots + a_1(x)E + a_0(x)$ as $\mathbb{S}(L) = \{i | 0 \le i \le m, a_i(x) \ne 0\}$ (set of indices of monomials that appear with non-zero coefficients).

The algorithm proceeds as follows.

1. Starting with $f(x)/g(x)$ ($f(x), g(x) \in \mathbb{K}[x]$) compute partial fraction decomposition **(ii)** in form (15). Note that further processing can be performed independently (see for example [34]) in each shift-equivalence class of denominator $g(x)$. Let's concentrate on a single shift equivalence class (further, index i is fixed and thus omitted for simplicity):

$$\sum_{j=1}^{m} M_j \frac{1}{g^j}, \tag{25}$$

2. Compute remainders r_j as in (18) and left quotients L_j in sparse form (recall that coefficients of the left quotient are just partial sums obtained when computing the remainder).

3. If all polynomials r_j are zero, there is nothing to do; return the answer in sparse form $\sum_{j=1}^{m} L_j \frac{1}{g^j}$. The size of the result (the degree of the denominator of the summable part) in this case is:
$m \deg(g) \cdot |\mathbb{S}(L_m)| +$
$(m-1)\deg(g) \cdot |\mathbb{S}(L_{m-1}) \backslash \mathbb{S}(L_m)| +$
$(m-2)\deg(g) \cdot |\mathbb{S}(L_{m-2}) \backslash \mathbb{S}(L_{m-1}) \backslash \mathbb{S}(L_m)| + \ldots$
$\cdots + 1 \cdot \deg(g) \cdot |\mathbb{S}(L_1) \backslash \bigcup_{k=2}^{m} \mathbb{S}(L_k)|$.

4. If at least one polynomial r_j is non zero, the input is not rational summable and one needs to find remainder in the form $r(x+c)E^c, 0 \le c \le \max_i(n_{ij})$ which minimizes the denominator of the summable part, keeping dispersion of the non-summable part equal to 0. This is trivial when the shift-equivalence class is represented by a single operator (has terms of the same multiplicity in partial fraction decomposition above). In this case one seeks the sparsest left quotient L_c

(quotient with minimal support) in

$$M = (E-1)L_c + r(x+c)E^c, 0 \le c \le \deg_E(M),$$

and

$$\min_{0\le c\le \deg_E(M)} |\mathbb{S}(L_c)| = \min_{c\in\mathbb{S}(M)} |\mathbb{S}(L_c)|.$$

Recall that when the dispersion of the input is large, $\deg_E(M)$ can be as large as the dispersion, but M has at most $\deg(g)$ nonzero terms, so $|\mathbb{S}(M)| \le \deg(g)$, and the smallest summable part can be found, in this case, in a number of steps which is linear in the degree of the denominator (by scanning sparse representation of the left quotient from step **2** right-to-left).
When the shift-equivalence class is represented by several operators

$$\sum_x \sum_{j=1}^m M_j \frac{1}{g^j} = \sum_{j=1}^m L_j \frac{1}{g^j} + \sum_x \sum_{j=1}^m \frac{r_j}{g^j}$$

one looks for remainders in the form $r_j(x+c)E^c$ (with the same c for all of them to keep dispersion of the non-summable part equal to 0) which minimizes degree of the denominator of the summable part. Let $k = \max\{j | r_j \ne 0\}$. It is easy to show that the minimum is achieved at $c \in \mathbb{S}(M_k)$. This means that the smallest summable part can be computed in the number of steps bounded by the degree of the denominator of the input (the details are given in [44] with several optimizations and description of a data structure similar to priority queue used to achieve linear time complexity of this last step). Note that algorithm in [35] requires a number of steps quadratic in the degree of the denominator in this step.
Remark. We used decomposition **(ii)** here, because decomposition **(iii)** (although superior in the case of summable input) may not guarantee minimality of the summable part for non-summable input. The following example is due to Jürgen Gerhard. Consider an input converted to the form **(iii)**

$$\big((1-x^2)E^{100} - E^{99} + E - (x^2+2)\big)\frac{1}{x^4+x^2}.$$

It is non-summable and any choice of the remainder will give dense quotients (denominator of the summable part will have degree 400). However, after factoring the denominator into $x^2(x^2+1)$, the summand becomes

$$(E-1)(E^{99}+1)\left(\frac{1}{x^2} - \frac{1}{x^2+1}\right) - \frac{1}{x^2} - E^{100}\frac{1}{x^2+1},$$

and the denominator of the summable part has degree 8. This suggests a combination: decomposition **(iii)** can be used in steps 1-3, and in those shift equivalence classes which contain nonzero remainders, denominators are further factored.

The treatment of indefinite quasi-rational sums

$$\sum_x \lambda^x \frac{f(x)}{g(x)},\ f(x), g(x) \in \mathbb{K}[x], \lambda \ne 0, \lambda \ne 1,$$

is very similar. We first represent $\frac{f(x)}{g(x)}$ in the form (15)

$$\frac{f(x)}{g(x)} = \sum_{i=1}^{v}\sum_{j=1}^{m_i} M_{ij}(E)\frac{1}{g_i^j},$$

and then substitute E/λ for E in each operator M_{ij}. This gives representation of quasi-rational input in the form

$$\lambda^x\frac{f(x)}{g(x)} = \sum_{i=1}^{v}\sum_{j=1}^{m_i} M'_{ij}(E)\frac{\lambda^x}{g_i^j},$$

and this input is summable if and only if every operator $M'_{ij}(E) = M_{ij}(E/\lambda)$ is left-divisible by $(E-1)$. The search for minimal summable part in case of non-summable input is the same as in the sketch of rational summation algorithm above.

4.3. Experiments

Polynomials

We are unaware of any implementation of fast basis conversion in Maple, and also were unable to obtain NTL implementation from authors of [10]. However Figure 4 in [10] gives a good idea of possible speedup of indefinite summation of polynomials. It compares timings of fast basis conversion with the naive basis conversion (which is essentially the time for basis conversion in our old implementation).

In order to show the potential gain from working with polynomials in the binomial basis, we compare timings in Maple to evaluate the following expression

$$V(x) = \sum_x(\sum_x P_1 + (\sum_x P_2 + P_1))$$

with random polynomials P_1, P_2 [9]. The timing comparison is presented in Table 1. Note that for basis conversion we use our ancient implementation [7] of chains of recurrences which is quadratic in the degree of the polynomials. According to

n	`Maple`	`Prototype`	`incl. Basis_conv`
100	0.031	0.016	<0.016
200	0.140	0.047	0.031
400	0.578	0.312	0.272
800	4.617	1.794	1.698
1600	33.088	12.558	11.912

Table 1. Timings in seconds for $V(x)$

the results in [10], the last column (in this range of degrees) can be reduced by a factor between 2 and 6 if fast basis conversion is used.

[9]All random polynomials in our experiments have integer coefficients between -99 and 99.

Quasi-polynomials

The comparison (especially in the case of quasi-polynomials, and quasi-polynomials whose coefficients depend on λ) is very favorable fir the direct algorithms. The following tables compare timings for summation of quasi-polynomials with random polynomial part performed by Maple 16 using `SumTools[Indefinite]`, `SumTools[Hypergeometric]` and our prototype. For the first and the second tables random dense polynomials $P_n(x)$ of degree n were generated, for the third table random dense polynomials in $P_n(x, \lambda)$ of degree n were used. A dash in the table indicates that Maple 16 did not return a result after 1000 seconds, and "Err" indicates that Maple 16 returned an error. Most of the time reported by our prototype was spent in basis conversion. As soon as fast basis conversion is implemented in Maple, these timings will improve tremendously (see Figure 4 in [10]).

n	`Indefinite`	`Hypergeometric`	`Quasi_Poly`
20	0.031	0.078	<0.016
40	0.156	0.078	<0.016
80	1.264	0.234	0.016
160	14.805	1.202	0.016
320	40.888	4.883	0.156

TABLE 2. Timings in seconds for $5^x P_n(x)$

n	`Indefinite`	`Hypergeometric`	`Quasi_Poly`
20	0.078	0.484	<0.016
40	0.796	7.893	<0.016
80	9.516	442.965	0.016
160	81.105	–	0.031
320	Err	–	0.218

TABLE 3. Timings in seconds for $\lambda^x P_n(x)$

n	`Indefinite`	`Hypergeometric`	`Quasi_Poly`
20	0.109	0.687	<0.016
40	0.967	7.941	0.031
80	12.246	415.322	0.124
160	101.478	–	1.139
320	290.208	–	10.156

TABLE 4. Timings in seconds $\lambda^x P_n(x, \lambda)$

Quasi-rational functions

Example (21) takes 0.031 seconds with our implementation, 0.889 with `SumTools[IndefiniteSum][Indefinite]` in Maple 16, and 0.265 seconds with `SumTools[Hypergeometric][SumDecomposition]`. Example (22) takes 0.031 seconds with our implementation and provides minimal degree of the denominator of the summable part, 0.437 with `SumTools[IndefiniteSum][Indefinite]` in Maple 16, and 0.390 seconds with `SumTools[Hypergeometric][SumDecomposition]` and returns summable part with the denominator of degree 41 with standard Maple implementation.

Note that these examples have relatively small value of the dispersion. The speedup can be made arbitrarily large, by increasing the value of the dispersion. For instance, the sum

$$\sum_x \frac{\left(8\,x^3 + 12006\,x^2 + 4005998\,x - 1001000\right) 5^x}{x^4 + 2002\,x^3 + 1003001\,x^2 + 1001000\,x}$$

is evaluated to

$$2\,\frac{5^x\,(x+500)}{x\,(x+1000)}$$

in 0.047 seconds by our code, and in 6.849 seconds by `SumTools[Hypergeometric][SumDecomposition]`.

Acknowledgments

The author wishes to express his thanks to the anonymous referees for their careful reading and many useful suggestions that helped to improve this paper.

References

[1] S. A. Abramov. The summation of rational functions. *Ž. Vyčisl. Mat. i Mat. Fiz.*, 11:1071–1075, 1971.

[2] S. A. Abramov. The rational component of the solution of a first order linear recurrence relation with rational right hand side. *Ž. Vyčisl. Mat. i Mat. Fiz.*, 15(4):1035–1039, 1090, 1975.

[3] S. A. Abramov. Indefinite sums of rational functions. In *Proceedings of the 1995 International Symposium on Symbolic and Algebraic Computation*, ISSAC '95, pages 303–308, New York, NY, USA, 1995. ACM.

[4] S. A. Abramov, M. Bronstein, and M. Petkovšek. On polynomial solutions of linear operator equations. In *Proceedings of the 1995 International Symposium on Symbolic and Algebraic Computation*, ISSAC '95, pages 290–296, New York, NY, USA, 1995. ACM.

[5] S. A. Abramov and M. Petkovšek. Rational normal forms and minimal decompositions of hypergeometric terms. *Journal of Symbolic Computation*, 33(5):521–543, 2002. Computer algebra (London, ON, 2001).

[6] S. A. Abramov et al. Telescoping in the context of symbolic summation in Maple. *J. Symbolic Comput.*, 38(4):1303–1326, 2004.

[7] O. Bachmann, P. S. Wang, and E. V. Zima. Chains of recurrences — a method to expedite the evaluation of closed-form functions. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC '94, pages 242–249, New York, NY, USA, 1994. ACM.

[8] G. Boole. *A Treatise on the Calculus of Finite Differences.* Cambridge Library Collection. Cambridge University Press, Cambridge, 2009. Reprint of the 1860 original.

[9] A. Bostan, F. Chyzak, B. Salvy, and T. Cluzeau. Low complexity algorithms for linear recurrences. In *Proceedings of the 2006 International Symposium on Symbolic and Algebraic Computation*, ISSAC '06, pages 31–38, New York, NY, USA, 2006. ACM.

[10] A. Bostan, B. Salvy, and E. Schost. Power series composition and change of basis. In *Proceedings of the International Symposium on Symbolic and Algebraic Computation*, ISSAC '08, pages 269–276, New York, NY, USA, 2008. ACM.

[11] G. P. Egorychev and E. V. Zima. On integral representation and algorithmic approaches to the evaluation of combinatorial sums. Technical Report CS-2002-02, School of Computer Science, University of Waterloo, January 2002.

[12] Euler. *Foundations of Differential Calculus.* Springer-Verlag, New York, 2000. Translated from the Latin by John D. Blanton.

[13] J. von zur Gathen and J. Gerhard. *Modern Computer Algebra.* Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[14] J. Gerhard. *Modular Algorithms in Symbolic Summation and Symbolic Integration*, volume 3218 of *Lecture Notes in Computer Science.* Springer, 2004.

[15] J. Gerhard, M. Giesbrecht, A. Storjohann, and E. V. Zima. Shiftless decomposition and polynomial-time rational summation. In *Proceedings of the 2003 International Symposium on Symbolic and Algebraic Computation*, pages 119–126 (electronic), New York, 2003. ACM.

[16] R. W. Gosper, Jr. Indefinite hypergeometric sums in MACSYMA. In *Proceedings of the 1977 MACSYMA Users' Conference*, pages 237–251, 1977.

[17] R. W. Gosper, Jr. Decision procedure for indefinite hypergeometric summation. *Proc. Nat. Acad. Sci. U.S.A.*, 75(1):40–42, 1978.

[18] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete Mathematics.* Addison-Wesley Publishing Company Advanced Book Program, Reading, MA, 1989. A foundation for computer science.

[19] C. Jordan. *Calculus of Finite Differences.* Hungarian Agent Eggenberger Book-Shop, Budapest, 1939.

[20] C. Jordan. *Calculus of Finite Differences.* Third Edition. Introduction by Harry C. Carver. Chelsea Publishing Co., New York, 1965.

[21] M. Karr. Summation in finite terms (preliminary version). Technical Report CA-7602-2911, Massachusetts Computer Associates Inc., 1976.

[22] M. Karr. Summation in finite terms. *Journal of the Association for Computing Machinery*, 28(2):305–350, 1981.

[23] M. Karr. Theory of summation in finite terms. *Journal of Symbolic Computation*, 1(3):303–315, 1985.

[24] J. C. Lafon. Summation in finite terms. In B. Buchberger, G. E. Collins, R. Loos, and R. Albrecht, editors, *Computer algebra. Symbolic and algebraic computation*, pages 71–77. Springer, Vienna, 1983.

[25] P. Lisoněk, P. Paule, and V. Strehl. Improvement of the degree setting in Gosper's algorithm. *J. Symb. Comput.*, 16(3):243–258, Sept. 1993.

[26] D. E. G. Malm and T. N. Subramaniam. The summation of rational functions by an extended Gosper algorithm. *J. Symb. Comput.*, 19(4):293–304, Apr. 1995.

[27] Y.-K. Man. On computing closed forms for indefinite summations. *Journal of Symbolic Computation*, 16(4):355–376, Oct. 1993.

[28] L. F. Matusevich. Rational summation of rational functions. *Beiträge Algebra Geom.*, 41(2):531–536, 2000.

[29] R. Moenck. On computing closed forms for summations. In *Proceedings of the 1977 MACSYMA Users' Conference*, pages 225–236, 1977.

[30] P. Paule. Greatest factorial factorization and symbolic summation. *J. Symb. Comput.*, 20(3):235–268, Sept. 1995.

[31] R. Pirastu. A note on the minimality problem in indefinite summation of rational functions. In *Séminaire Lotharingien de Combinatoire (Saint-Nabor, 1993)*, volume 1994/21 of *Prépubl. Inst. Rech. Math. Av.*, pages 95–101. Univ. Louis Pasteur, Strasbourg, 1994.

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

[33] R. Pirastu and V. Strehl. Rational summation and Gosper-Petkovšek representation. *J. Symb. Comput.*, 20(5-6):617–635, Nov. 1995.

[34] S. P. Polyakov. Indefinite summation of rational functions with additional minimization of the summable part. *Programming and Computer Software*, 34(2):48–53, 2008.

[35] S. P. Polyakov. Indefinite summation of rational functions with factorization of denominators. *Programming and Computer Software*, 37(6):322–325, 2011.

[36] C. Schneider. An implementation of Karr's summation algorithm in Mathematica. *Sém. Lothar. Combin.*, 43:Art. S43b, 10 pp. (electronic), 1999.

[37] C. Schneider. Symbolic summation with single-nested sum extensions. In *ISSAC 2004*, pages 282–289. ACM, New York, 2004.

[38] C. Schneider. A new Sigma approach to multi-summation. *Adv. in Appl. Math.*, 34(4):740–767, 2005.

[39] C. Tweedie. Nicole's contribution to the foundations of the calculus of finite differences. *Proceedings of the Edinburgh Mathematical Society*, 36:22–39, 2 1917.

[40] M. van Hoeij. Rational solutions of linear difference equations. In *Proceedings of the 1998 international symposium on Symbolic and algebraic computation*, ISSAC '98, pages 120–123, New York, NY, USA, 1998. ACM.

[41] E. V. Zima. Simplification and optimization transformations of chains of recurrences. In *Proceedings of the 1995 International Symposium on Symbolic and Algebraic Computation*, ISSAC '95, pages 42–50, New York, NY, USA, 1995. ACM.

[42] E. V. Zima. Direct algorithm for indefinite quasi-rational summation. Technical Report CARGO-2010-03, CARGO lab, WLU, November 2010.

[43] E. V. Zima. Synthetic division in the context of indefinite rational summation. Technical Report CARGO-2011-01, CARGO lab, WLU, January 2011.

[44] E. V. Zima. Synthetic division in the context of indefinite summation. In *Proceedings of the 2011 International Workshop on Symbolic-Numeric Computation*, SNC '11, pages 151–152, New York, NY, USA, 2011. ACM.

Eugene V. Zima
Wilfrid Laurier University,
Waterloo, Ontario, Canada
e-mail: `ezima@wlu.ca`