Creative Telescoping on Multiple Sums

Christoph Koutschan and Elaine Wong

Abstract. We discuss the strategies and difficulties of determining a recurrence which a certain polynomial (in the form of a symbolic multiple sum) satisfies. The polynomial comes from an analysis of integral estimators derived via quasi-Monte Carlo methods.

Keywords. Symbolic Summation, Creative Telescoping, Holonomic Function, Hypergeometric Series.

1. The Problem

Recently, Wiart and Wong [6] derived a formula for the covariance of an integral estimator for functions satisfying a certain decay condition, based on a quasi-Monte Carlo framework developed by Wiart, Lemieux, and Dong [5]. This formula is written as the following polynomial in x,

$$G_s(x) := \sum_{k=1}^{m+s-1} \left(\sum_{r=1}^{s} \binom{s}{r} \binom{k-1}{r-1} \frac{b-1}{(-b)^r} \sum_{i=0}^{r-1-c_m(k)} (-b)^i \binom{r-1}{i} \right) (bx)^k, \qquad (1.1)$$

where $c_m(k) = \max(k-m, 0)$. The goal in [6] is to show that (1.1) is not positive for all $b, m, s \in \mathbb{N}$, $b \geq 2$ and $x \in [0, 1)$. We approach this problem by employing symbolic computation rather than analysis: using the available computer algebra software for holonomic functions [1, 2, 4], we carry out a guess-and-prove strategy that ultimately leads us to deduce a suitable closed form for (1.1). The result is an expression in terms of regularized beta functions which allows us to show the desired nonpositivity statement. This article serves to highlight the "proving" aspect of the strategy, i.e., the derivation and proof of a third-order recurrence that (1.1) satisfies. At the same time, we discuss some general solutions to address difficulties in summation problems of a similar nature.

Both authors were supported by the Austrian Science Fund (FWF): F5011-N15.

2. Motivation and Background

On a first glance, we note that all constituents of (1.1) have the property of being “holonomic”. For the purposes of this paper, we stay slightly informal (but still rigorously practical) and use the definition that holonomic functions are sequences which satisfy “sufficiently many” linear recurrences with polynomial coefficients. One convenience of dealing with such functions comes from the fact that holonomicity is preserved through basic operations: we will refer to these as “closure properties”. For example, the product of two holonomic functions is again holonomic [7, Proposition 3.2]. From a computational point of view, if we know the recurrences that the two holonomic functions satisfy, we can construct a recurrence that their product satisfies and even bound its order and the degrees of its polynomial coefficients.

Our expression (1.1) consists of summation quantifiers, (products of) binomial coefficients, and polynomial/exponential functions in the parameters. The binomial coefficient, for example, can be described completely via two recurrences and finitely many initial conditions. From our usual notion of binomial coefficients, we can immediately write down the two recurrences (valid on $\mathbb{Z} \times \mathbb{Z}$):

$$\begin{aligned} (n-k+1)\binom{n+1}{k} - (n+1)\binom{n}{k} &= 0, \\ (k+1)\binom{n}{k+1} - (n-k)\binom{n}{k} &= 0, \end{aligned} \tag{2.1}$$

and specify the initial conditions

$$\binom{0}{0} = 1, \quad \binom{-1}{0} = 0, \quad \binom{-1}{-1} = 0.$$

By doing so, we interpret the binomial coefficient over the integers in the traditional (combinatorial) way, namely, it is nonzero only for $0 \leq k \leq n$. We explicitly highlight this because the computer algebra system Mathematica uses a more general notion of the binomial coefficient, which extends its definition to the negative integers. It is important to note that this generalized binomial coefficient is defined by the very same recurrences, but just uses different initial conditions:

$$\binom{0}{0} = 1, \quad \binom{-1}{0} = 1, \quad \binom{-1}{-1} = 1.$$

Unfortunately, the more general view of the binomial coefficient affects the natural boundaries of our summation limits: the summands containing these coefficients behave inappropriately outside of our prescribed bounds, which is of course irrelevant regarding the definition of $G_s(x)$, but which may cause problems when evaluating boundary terms originating from the telescoping sums. We will have to address this issue later.

In the following, we will use this simple bivariate sequence $\binom{n}{k}$ to illustrate the main features of the holonomic systems approach. Using the notation S_n resp. S_k to represent the forward shift operator in the given variable, we can rewrite (2.1) so that each of the corresponding operators

$$\begin{aligned} (n-k+1)S_n - (n+1),\\ (k+1)S_k - (n-k), \end{aligned} \tag{2.2}$$

maps $\binom{n}{k}$ to the zero sequence. We say that these operators annihilate the given function. As one can see, the translation between recurrence and operator can be read off immediately. Viewing recurrences as operators enables the use of algebraic methods to manipulate them more efficiently. However, we then have to deal with objects that do not always commute. The appropriate algebraic framework to represent such operators is an Ore algebra. In the following technical definition, ∂ serves as a placeholder for any of our operator symbols S_n or S_k.

Definition 2.1. Let R be a ring.

1. If $\sigma: R \to R$ is a ring endomorphism and $\delta: R \to R$ is such that the "skew" Leibniz law is satisfied, that is,

$$\delta(fg) = \delta(f)g + \sigma(f)\delta(g)$$

   for all $f, g \in R$, then δ is called a σ-derivation.
2. Suppose now that there is an endomorphism $\sigma: R \to R$ and a σ-derivation $\delta: R \to R$. Suppose further that a ring structure is defined on the set $R[\partial]$ of univariate polynomials in ∂ with coefficients in R, equipped with the usual addition, and multiplication is such that

$$\partial^i \partial^j = \partial^{i+j} \quad \text{and} \quad \partial f = \sigma(f)\partial + \delta(f) \qquad \text{for all } i, j \in \mathbb{N} \text{ and } f \in R.$$

   Then $R[\partial]$ is an Ore algebra over R. We typically use the symbol $\mathbb{O}$ to denote such algebras.
3. Suppose that f is in a left-$R[\partial]$-module with action $\cdot : \mathbb{O} \times R \to R$ such that $1 \cdot f = f$ and $L_1 \cdot (L_2 \cdot f) = (L_1 L_2) \cdot f$ for all $L_1, L_2 \in \mathbb{O}$. Then we call

$$\operatorname{ann}(f) = \{L \in R[\partial] \mid L \cdot f = 0\}$$

   the annihilator of f in $R[\partial]$.

For the binomial coefficient $\binom{n}{k}$, the Ore algebra that we use is $\mathbb{O} = R[S_k, S_n]$ with $R = \mathbb{Q}(n,k)$. A quick check shows that shift operators satisfy the required commutation properties. In the definition of this Ore algebra, each σ denotes a forward shift operation (clearly a ring endomorphism) and $\delta \equiv 0$ (clearly a σ-derivation).

The reader may have wondered why we did not consider Pascal's rule as a potential defining recurrence for the binomial coefficient. The reason is that (2.1) are quite canonical generators for the set of all recurrences satisfied by $\binom{n}{k}$. In algebraic terms, we can formulate this statement precisely: the annihilator of $\binom{n}{k}$, which is a left ideal in $\mathbb{O}$, is generated by the operators (2.2), let's call them $P_1, P_2 \in \mathbb{O}$:

$$\operatorname{ann}\binom{n}{k} = \{C_1 \cdot P_1 + C_2 \cdot P_2 \mid C_1, C_2 \in \mathbb{O}\}.$$

Moreover, the two operators in (2.2) even form a (left) Gröbner basis of $\operatorname{ann}\binom{n}{k}$.

The key tool used in rigorously deriving a "grand" recurrence for $G_s(x)$ lies in the highly touted creative telescoping algorithm [8] for symbolic sums and integrals, as implemented in the HolonomicFunctions package [2]. In order to construct a recurrence for a symbolic parametric sum of the form

$$\sum_k \text{summand}, \tag{2.3}$$

the algorithm takes as input a list of generators, like the ones in (2.2), for an annihilating ideal of the summand. If the summand is given as a closed-form expression, then such a list is automatically computed, provided that it is recognized to be holonomic. The algorithm then identifies lists of operators $\mathcal{P}$ and $\mathcal{Q}$ (in the form of Ore polynomials in the algebra as described in Definition 2.1) such that for each $P \in \mathcal{P}$ and its corresponding $Q \in \mathcal{Q}$, the operator $P - (S_k - 1) \cdot Q$ is an element of the given annihilating ideal. The set $\mathcal{P}$ contains the so-called "telescopers" (all of which are free of k and S_k but may contain the other parameters), and the set $\mathcal{Q}$ the corresponding "certificates".

How do these objects help us? Summing with respect to k gives relations of the form

$$\sum_k P \cdot \text{summand} - \sum_k (S_k - 1) \cdot Q \cdot \text{summand} = 0. \tag{2.4}$$

In a best-case scenario, each of the P's commutes with the first summation in (2.4) (allowing us to pull it out of the sum so that we can view the elements of $\mathcal{P}$ being applied to the whole sum and not just the summand) and the second summation collapses to zero by telescoping (leaving no trace of the certificate). From there, we would conclude that $\mathcal{P}$ generates a left ideal of annihilating operators for (2.3), that is, it represents a set of recurrences which are satisfied by the sum. Coming back to our triple sum (1.1), we can repeatedly apply this process until a recurrence for the outermost (and hence the whole) sum is deduced.

However, life is not always that easy: during the application of this strategy to the particular summation problem (1.1), we encountered the following difficulties that are somewhat prototypical for the holonomic systems approach. This explains why, despite being automatable in principle, it still lacks a press-the-button implementation that would provide a computer proof of a claimed identity in a completely automatic way and without any human interaction.

1. The summand, i.e., the expression inside a summation quantifier, may take on nonzero values outside of the respective summation bounds. Thus, there is no reason to expect a priori that the second summation in (2.4) will evaluate to zero. And indeed, we found that it did not, and such terms constitute some of the "inhomogeneous parts" of the equation. An additional annihilator for them is required in order to homogenize the recurrence.

2. The upper boundaries contain the variable s, and the operators in $\mathcal{P}$ contain shifts in s, causing difficulties with moving $P \in \mathcal{P}$ to be outside of the sum.
3. Some of the operators in $\mathcal{Q}$ contain singularities at the boundary values so we were forced to exclude these values (which required compensation elsewhere). This is because the sum in (2.4) containing the certificates is designed to collapse to only boundary value evaluations and we encounter problems if the summands are undefined at such values. Further issues could surface if those summands were also undefined at some intermediary value. Luckily, this was not the case here.
4. Mathematica, in its symbolic zeal, rewrites the innermost sum in (1.1) as a hypergeometric ${}_2F_1$ series and the second innermost sum as a DifferenceRoot. While the values of the ${}_2F_1$ function match with our sum within the domain in question, there are still an infinite number of values for which it doesn't. The DifferenceRoot is Mathematica's version of a recurrence together with initial values, but unfortunately not helpful for our purposes because it is incompatible with HolonomicFunctions and does not support multivariate recurrences that are needed for creative telescoping.

We illustrate the first three difficulties with a toy example. For thoroughness, we apply our strategy fully to this example to give a sufficient idea of the broader behavior. From this point on, we will periodically perform the service of demonstrating how computers and humans interact, by highlighting (in brackets) when paper-and-pencil reasoning is used and when automation is applied.

Suppose we want to rigorously determine an annihilating operator for

$$\sum_{k=5}^{n} \binom{n}{k} \tag{2.5}$$

for $n \geq 5$. In other words, we would like to identify a recurrence that it satisfies. The creative telescoping algorithm (computer) outputs the telescoper $P = S_n - 2$ and the certificate $Q = \frac{k}{k-n-1}$. Then (2.4) implies

$$\sum_{k=5}^{n} (S_n - 2)\binom{n}{k} - \underbrace{\left(\frac{k}{k-n-1}\binom{n}{k}\bigg|_{k=n+1} - \frac{k}{k-n-1}\binom{n}{k}\bigg|_{k=5} \right)}_{\text{collapsed sum with singularity at } k=n+1} = 0,$$

with the summation containing the certificate collapsing to only evaluations at the boundary values. We note that $\binom{n}{k}$ is nonzero for $k = 0, \ldots, 4$, i.e., outside of the summation bounds. After substituting $k = 5$ we get a nonzero contribution in the certificate (compare this to the situation where the lower bound is ≤ 0). We next note that the certificate has a singularity at $k = n + 1$ and this prevents the left expression in the large brackets from being evaluated. We can therefore choose to sum up to $n - 1$ instead (then the boundary evaluation will occur at $k = n$ rather than $k = n+1$). With this, (2.4) turns into

$$\sum_{k=5}^{n-1}(S_n-2)\binom{n}{k}-\underbrace{\left(\frac{n^4-6n^3+11n^2-30n}{24}\right)}_{\text{inhomogeneous part}}=0. \tag{2.6}$$

This fixes the issue of the singularity (alternatively, we could have rewritten $\frac{k}{k-n-1}\binom{n}{k}=\frac{-k}{n+1}\binom{n+1}{k}$ to get rid of the pole). Next, we note that the upper summation bound contains the parameter n while our telescoper P contains the shift operator S_n: applying the operator to the whole sum affects both the upper bound and $\binom{n}{k}$. This is fixed with the (human) observation that

$$\sum_{k=5}^{n-1}(S_n-2)\binom{n}{k}=(S_n-2)\sum_{k=5}^{n}\binom{n}{k}\underbrace{-\binom{n+1}{n}-\binom{n+1}{n+1}+2\binom{n}{n}}_{\text{compensated terms }=-n}.$$

In this situation, we say that the operator and the summation does not commute. Then (2.6) simplifies to the inhomogeneous recurrence

$$(S_n-2)\sum_{k=5}^{n}\binom{n}{k}=\frac{n^4-6n^3+11n^2-6n}{24}.$$

If one prefers a homogeneous recurrence, an annihilating operator for the right-hand side can be determined to be $(n-3)S_n-(n+1)$. We can therefore conclude that

$$\big((n-3)S_n-(n+1)\big)\cdot(S_n-2)=(n-3)S_n^2+(5-3n)S_n+2(n+1)$$

is the desired annihilating operator for (2.5). As is expected in such simple examples, the recurrence can be solved (by the computer) to obtain a closed form for (2.5). Of course, it agrees with the one that one directly gets from invoking the binomial theorem.

In the above example, we can see that there is a "dance" between human and the computer and only upon a careful collaboration does it bear fruit. We now proceed to use a similar strategy to attack the big sum $G_s(x)$ and furthermore present some alternatives to improve performance. The total computation time largely depends on how complicated the summands and inhomogeneous parts turn out to be after (human) simplification. The next section outlines some of these strategies and in particular highlights how we were able to successfully derive (and prove) a recurrence for $G_s(x)$.

3. A Playbook for the Holonomic Approach

This section illustrates how to generally overcome the difficulties listed in the previous section and how to effectively perform the human-computer dance to prove our main result. We envision that the discussion leads to a deeper understanding of the practical issues when applying the holonomic systems approach and makes it accessible for future applications. The Mathematica notebook containing implementations of these strategies can be found in the online supplementary material [3].

Theorem 3.1. *For $b, m, s \in \mathbb{N}, b \geq 2$, the polynomial given in* (1.1),

$$G_s(x) := \sum_{k=1}^{m+s-1} \left(\sum_{r=1}^{s} \binom{s}{r} \binom{k-1}{r-1} \frac{b-1}{(-b)^r} \sum_{i=0}^{r-1-c_m(k)} (-b)^i \binom{r-1}{i} \right) (bx)^k,$$

satisfies the recurrence

$$\begin{aligned}&(s+2)(bx-1) \cdot G_{s+3}\\&+ (m(bx-1)(x-1) + bsx(x-2) + bx(x-3) - s(2x-3) - 3x + 5) \cdot G_{s+2}\\&- (x-1)(bmx + bsx + bx + mx - 2m + sx - 3s + x - 4) \cdot G_{s+1}\\&+ (x-1)^2(m+s+1) \cdot G_s = 0.\end{aligned}$$

We note that this result is already contained in [6, Lemma 15] with computational details found in [3]. The following discussion serves to outline alternate (and in some cases, faster) proof strategies, to provide some exposition for technical details that were not mentioned in [6], and to explicitly resolve some of the issues mentioned in the previous section in as much generality as possible under the context of using our problem as a case study. We hope that this will be useful for future practitioners.

3.1. Eliminating the Max

Before we dive in, we make a few remarks about how to view $G_s(x)$ to make our life easier. On the one hand, the summation (1.1) can be separated into two parts $G_s = G_s^{(1)} + G_s^{(2)}$, in order to remove the max function in the upper limit of the innermost sum. After a mild simplification (human), these two parts look as follows:

$$G_s^{(1)} := -\sum_{k=1}^{m+s-1} \sum_{r=1}^{s} \binom{s}{r} \binom{k-1}{r-1} \left(\frac{b-1}{b}\right)^r (bx)^k,$$

$$G_s^{(2)} := \sum_{k=m+1}^{m+s-1} \sum_{r=1}^{s} \binom{s}{r} \binom{k-1}{r-1} \frac{1-b}{(-b)^r} \sum_{i=r-(k-m)}^{r-1} (-b)^i \binom{r-1}{i} (bx)^k.$$

Observe that $-G_s^{(2)}$ is the collection of terms that is added to G_s to enable the sum to collapse to $G_s^{(1)}$. We write this out to show that initially applying the full strategy to "only" the double sum gives a hypothetical lower bound for the time and effort required to treat the whole thing. We note that if $k > m + s - 1$, then there is no reason to expect that either of the inner sums would be zero, which may cause the inhomogeneous parts in (2.4) to survive.

The split sums also serve as an example of how to apply closure properties: the sum of holonomic functions is still holonomic [7, Proposition 3.1], so

$$\operatorname{ann}\left(G_s^{(1)} + G_s^{(2)}\right)$$

can be deduced by executing (computer) the corresponding "closure property of addition" algorithm after separately computing a respective annihilating ideal for each of the two terms. This closure property can also be applied in intermediate computations (for example, during the treatment of the inhomogeneous parts). However, the user should be aware that there is a risk that the recurrence order (more precisely: the holonomic rank) may increase during each such application (but not more than the sum of their orders).

On the other hand, if we choose to deal with the full triple sum right from the start, then we can do better. By observing that when $k - m < 0$, we have the situation where $r - 1 < i \leq r - 1 - (k - m)$ forces the innermost binomial coefficient $\binom{r-1}{i}$ to be 0. Thus, the max function in $G_s(x)$ can be safely removed. Addition is commutative, so it does not matter if we move all summations to the front and consider only one summand with three indexed parameters. In other words, $G_s(x)$ can be rewritten as

$$\underbrace{\sum_{k=1}^{m+s-1} \sum_{r=1}^{s} \sum_{i=0}^{r-1-(k-m)}}_{\text{quantifiers grouped}} \underbrace{\binom{s}{r}\binom{k-1}{r-1}\binom{r-1}{i}\frac{b-1}{(-b)^{r-i}}(bx)^k}_{\text{one summand}}. \tag{3.1}$$

3.2. Guessing

Another "preparation" step involves employing the Guess package [1] to predict the recurrence that our polynomial satisfies, by using sufficiently generic evaluations of (3.1). This step gives us confidence that a nice enough recurrence exists, and it serves as an additional sanity check. For our problem, the (computer) guessing procedure already produced the claimed minimal third-order recurrence (from Theorem 3.1) in the parameter s. To prove that the guess is correct, it is enough to compute the same recurrence (or a higher order one) via creative telescoping. In the latter case, one also has to verify that the guessed recurrence (operator) is a right factor of the bigger one, and consider a sufficient number of initial values.

In general, the creative telescoping algorithm works very well and can be applied directly without adjustments if all conditions are "ideal". In such cases, the outputted telescoper corresponds exactly to the recurrence that we want. If certain issues arise such that (human) adjusting for them produces extraneous terms that are not a part of the original sum and/or come from compensations due to the rebuilding of the original sum, then we need to collect all such terms and find a collective annihilator for them for the purpose of "homogenizing" the recurrence given by the telescoper.

3.3. Dealing with Singularities

We found that some certificates Q contain singularities on the boundary values of the inner sum. This implies that the limits of the sum must be adjusted so that we avoid evaluating at those points. In fact, the summation range must be adjusted so that there are no singularities at **all** intermediate values.

To illustrate this a little more generally, suppose that, upon applying the creative telescoping algorithm on the summation with respect to r, the computer outputs a certificate $Q \in \mathbb{Q}(s, r)$ containing poles at $r_i \in [1, s+1] \subset \mathbb{N}$ for a finite number of i. All parameters besides r are treated symbolically.

Then we can see that the sum $\sum_{r=1}^{s}(S_r - 1)\cdot Q\cdot F(s,r)$ cannot be determined because evaluations at those poles for the summation range $[1,s]$ are not possible and therefore the sum is undefined.

Such singularities can be removed from the offending sum so that the evaluation(s) can happen. We also remove the exact same values from the summations containing the telescopers to match so that (2.4) makes sense. The summations with the telescopers can then be subsequently "filled in" with the removed summands (balanced of course by subtracting those same terms from the inhomogeneous part). This strategy can be quite effective if all of the poles are collected contiguously at either of the summation bounds, or if there are only one or two. Note that if the number of compensated terms required is too large, it may be be better to consider another strategy, such as rewriting terms in some alternative (but equivalent) form to avoid the pole entirely (as was suggested in the analysis for (2.5)).

3.4. Pulling Operators Outside of the Sum

We also found that the telescoper P does not commute with our summation. To illustrate this a little more generally, suppose the operator P is in the Ore algebra generated by the shift operator S_s. In other words, suppose that P can be written as some polynomial in S_s, for example,

$$P = p_0 + p_1 S_s + \cdots + p_j S_s^j,$$

where the p_i $(i = 0,\dots,j)$ may be rational functions containing the parameter s. If we apply such a P to a summation of the form $\sum_{k=1}^{m+s-1} H(s,k)$, then we face the issue that the application not only affects the parameter s in the summand $H(s,k)$, but also the upper limit $m+s-1$. Then if we apply P to the whole sum, we get

$$p_0 \sum_{k=1}^{m+s-1} H(s,k) \;+\; p_1 \sum_{k=1}^{m+s} H(s+1,k) \;+\cdots+\; p_j \sum_{k=1}^{m+s+j-1} H(s+j,k). \tag{3.2}$$

It is quite obvious that this is not the same as applying P to only the summand $H(s,k)$:

$$\sum_{k=1}^{m+s-1} \Big(p_0 H(s,k) + p_1 H(s+1,k) + \cdots + p_j H(s+j,k)\Big). \tag{3.3}$$

However, we can simulate the "factoring out" of the P in (3.3) if we peel off a sufficient (and finite) number of terms from each sum in (3.2) such that its upper limits are all $m+s-1$. Then (3.3) can be replaced by the peeled version of (3.2) with P on the outside and the removed terms can then be merged with the inhomogeneous part.

3.5. Treatment of Inhomogeneous Parts

In all of our examples, the adjustment of the summation limits to avoid singularities in the certificates was completed first. After that, it is a priori not clear if one should proceed to adjust for the operator commuting with the summand or to fill in the terms that were removed for the singularities.

The (human) decision may depend on the number of singularities, where the singularities are located, how complicated the telescoper expression is, and whether or not the lower/upper boundaries are influenced by the telescopers. This makes it difficult to automate adjustments effectively.

Once we have collected all of the inhomogeneous parts, we face the question of how to process them. In principle, we could just write them down and try to use Mathematica's symbolic power to simplify them as much as possible. Unfortunately, this does not work very well on our problem, with the only progress being that some of the inhomogeneous parts conveniently collapse to zero (so we remove them). Instead, we take advantage of the fact that we can write all of the inhomogeneous parts as different shifts and substitutions of the given summand. More precisely, the total of some of these parts can be expressed as an operator applied to the summand, followed by a substitution. Then, an annihilator for this total can be derived by applying the closure properties "application of an operator" and "integer-linear substitution". In this way, we completely avoid dealing with expressions like Mathematica's DifferenceRoots.

We illustrate this strategy with an example that comes from the inhomogeneous parts for $G^{(1)}$. It involves observing patterns in a complicated expression to construct an operator that would give the same result when it is applied to some simpler version of the expression. First, we identified all terms that contain a certain hypergeometric ${}_2F_1$ series:

$$\begin{aligned} &-\frac{(b-1)(m+s+1)(bx)^{m+s+1}}{b^2x}\cdot {}_2F_1\left(1-s,-m-s;2;\frac{b-1}{b}\right)\\ &+\frac{(b-1)(m+bs)(bx)^{m+s}}{b^2}\cdot {}_2F_1\left(1-s,1-m-s;2;\frac{b-1}{b}\right)\\ &\frac{(b-1)(s+1)(bx-1)(bx)^{m+s}}{b}\cdot {}_2F_1\left(1-m-s,-s;2;\frac{b-1}{b}\right). \end{aligned} \tag{3.4}$$

From there, we can see that selecting the operator

$$\frac{(b-1)(m+s+1)}{b^2x(bx)}S_m^2-\frac{(b-1)(m+bs)}{b^2(bx)}S_m+\frac{(b-1)(s+1)(bx-1)}{b(bx)}S_s \tag{3.5}$$

and applying it to $(bx)^{m+s}\cdot {}_2F_1\big(1-s,2-m-s;2;\frac{b-1}{b}\big)$ gives (3.4). The annihilator for (3.4) can therefore be obtained by "applying" (3.5) (in the sense of closure properties) to the annihilator of this single expression. This process is much faster than trying to directly compute an annihilating ideal of the sum of hypergeometric series, with the added benefit that the order of the recurrence will usually be smaller compared to applying the Annihilator command directly to expressions such as (3.4). In our particular example, the latter method even caused the program to crash.

We can therefore see that directly constructing an operator by closely inspecting patterns in the inhomogeneous parts can be computationally effective. In a way, the fact that we use ${}_2F_1$'s in the previous argument is inconsequential to the construction of the operator to be applied (it could have easily be replaced with a symbolic expression that exhibits similar shift behaviors, for example). Thus, an annihilating operator for the inhomogeneous terms can be deduced in this way *before* administering any substitutions. As mentioned before, this could also involve removing all terms that would collapse to zero anyway and building from scratch the new operator by only using the shifts needed to produce compensation terms that may have resulted from the treatment of singularities and commutation. We similarly applied this strategy to $G_s^{(2)}$ and it cost us 30 hours.

3.6. Substitution Speedup

The collection of all inhomogeneous parts and its subsequent removal via its annihilator can eat up a lot up computation time depending on how complicated these parts are. In particular, we experience this in the computation of the annihilators for the inhomogeneous part of $G_s^{(2)}$ when applying the closure property of integer-linear substitution. Thus, as an alternative to blindly applying the corresponding computer command and not knowing what is going on behind the scenes while waiting patiently for the code to finish, we can take better control of the process by making a few additional optimizations, resulting in a significant speedup in computation time.

With the “application of an operator” closure property we produced an annihilating ideal, with its Gröbner basis denoted by $U^{(2)}$, for the combined inhomogeneous parts of $G_s^{(2)}$, denoted by $H(s,k)$, but without the necessary substitution $k \to m+s$ according to the upper summation bound. This implies that it is necessary to apply the closure property “integer-linear substitution” to $U^{(2)}$.

The above Gröbner basis $U^{(2)}$ has the set of irreducible monomials $\{S_s, S_k, 1\}$ and hence is of holonomic rank 3. The theory tells us that for $H(s, m+s)$ we should expect a recurrence of order 3 in s, however, we know (from the longer computation in Section 3.5) that our final recurrence is of order 2 in s.

For constructing such a recurrence, we want to find an operator T in the left ideal generated by $U^{(2)}$ with the support $\{S_s^2 S_k^2, S_s S_k, 1\}$, corresponding to a bivariate recurrence involving the terms

$$H(s+2, k+2),\ H(s+1, k+1),\ H(s,k),$$

which, after the substitution $k \to m+s$, turns into the desired second-order recurrence for $H(s, m+s)$. We therefore make the following ansatz for T:

$$T = c_2(k,s) S_k^2 S_s^2 + c_1(k,s) S_k S_s + c_0(k,s),$$

where the coefficients $c_0(k,s), c_1(k,s), c_2(k,s)$ are to be determined. Gröbner basis theory tells us that this T is an element of the annihilating ideal (in other words: represents a valid recurrence for $H(s,k)$) if and only if it reduces to 0 by the Gröbner basis $U^{(2)}$. Reducing the ansatz T by $U^{(2)}$ results in a linear combination of the basis monomials $\{S_s, S_k, 1\}$, i.e., an Ore polynomial of the form

$$E_2(k,s,c_0,c_1,c_2) S_s + E_1(k,s,c_0,c_1,c_2) S_k + E_0(k,s,c_0,c_1,c_2)$$

with rational functions E_0, E_1, E_2. This polynomial is zero if and only if $E_0 = E_1 = E_2 = 0$, so we proceed to solve this system for c_0, c_1, c_2.

Ultimately, this procedure gives us an Ore polynomial with the support we want that annihilates the inhomogeneous parts after substituting $k \to m+s$ in the coefficients of T and omitting the shift operator S_k. This is essentially the same as saying: substitute $k \to m+s$ into the recurrence $T \cdot H(s,k)$. However, since this substitution tends to decrease the size of expressions (it reduces the number of variables), it is desirable to perform it as early as possible, and not only at the very end.

Indeed, we were able to speed up our computation significantly by performing the substitution already during the reduction of the monomials of T, but care has to be taken: to match the leading monomials, we may have to multiply by (a power of) S_k, and for this (noncommutative) multiplication one needs to keep the variable k. However, it can be substituted immediately afterwards. This leads to a less dramatic swell of expressions. Other sources of speedup include a manual selection strategy of Gröbner basis elements to be used for the reduction, and the order in which these reductions are made.

It might be worthwhile to note that there are places in this process where we got lucky: as fate would have it, the coefficient E_2 of S_s is zero (but only for $k = m+s$), and this gives us the luck of finding a recurrence of order two instead of three! This procedure has now reduced our total computation time to 1.4 hours.

3.7. Gamma Insertions

This next strategy takes a more holistic approach in that we treat the triple sum (3.1) all at once and make adjustments to the single summand in order to deal with the "unnatural boundary" problems simultaneously. For our problem, this is accomplished by making the observation that the issue of "unnatural boundaries" occurs whenever all three binomial coefficients in our summand are nonzero beyond the limits of our summation. Assuming that m and s are fixed positive integers, we let B_1 denote the collection of points $(k,i,r) \in \mathbb{Z}^3$ for which the summand in (3.1) is nonzero, and let B_2 denote all points $(k,i,r) \in \mathbb{Z}^3$ that are inside the summation ranges. While B_2 corresponds to all integer points of a bounded polytope in $\mathbb{R}^3$, the set B_1 is unbounded (see Fig. 1). We essentially want to sum over all points (k,i,r) in the intersection of B_1 and B_2 (depicted in blue), while we want to avoid those points in $B_1 \setminus B_2$ (depicted in red). Hence, the set of "bad points" also forms an infinite polytope and we remove these points using gamma functions.

We first recall that $\Gamma(k)$ has poles exactly at the non-positive integers, and therefore $\frac{1}{\Gamma(k)}$ has zeros at $k = 0, -1, -2, \ldots$. Then the summand can be modified by the following gamma functions in order to enforce natural boundaries:

$$C(\varepsilon,i,k,r) := \binom{s}{r} \cdot \binom{k-1}{r-1} \cdot \binom{r-1}{i} \cdot \frac{\Gamma(k+\varepsilon)}{\Gamma(k)} \cdot \frac{\Gamma(r-i-(k-m)+\varepsilon)}{\Gamma(r-i-(k-m))}$$

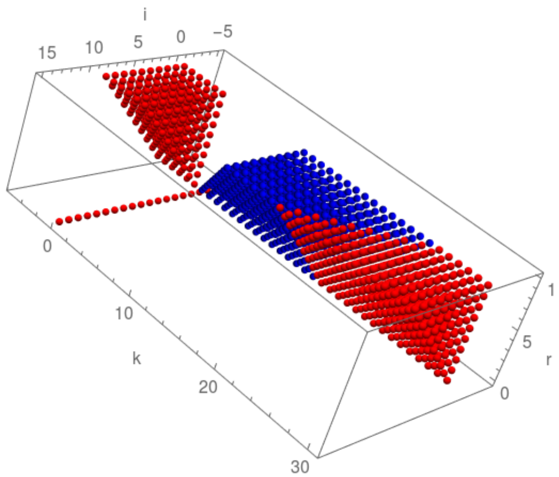

FIGURE 1. "Bad" points $B_1 \setminus B_2$ (red) and "good" points $B_1 \cap B_2$ (blue) for fixed $m = 15$ and $s = 10$.

with some new symbol ε. Upon sending $\varepsilon \to 0$ we get back the original product of binomial coefficients, while the two additional factors force the expression $C(\varepsilon, i, k, r)$ to be zero whenever $k \leq 0$ or $r - i - (k - m) \leq 0$. In other words, the introduction of the reciprocal gammas is balanced out by the perturbed gammas in the numerators, which conveniently avoids division by zero and gives an equivalent (final) result for the original problem after setting $\varepsilon = 0$. We comment that the gamma function with the perturbation is a hypergeometric (and therefore holonomic) term that takes on a finite value away from the pole.

We have now achieved a summand such that creative telescoping can be applied without having to worry about undesirable terms from beyond the summation boundaries. Another consequence of this is that the telescoper can be pulled out of the summation. Unfortunately, this procedure does not completely remove the threat of singularities that may show up in the certificates, so inhomogeneous parts can surface here and must be treated (this turned out to be the case in our situation). Except for that, the net effect of introducing gammas is so that we can apply the creative telescoping algorithm (three times) to the triple sum

$$\sum_{k=1}^{m+s-1} \sum_{r=1}^{s} \sum_{i=0}^{r-1-(k-m)} C(\varepsilon, i, k, r) \cdot \frac{b-1}{(-b)^{r-i}} (bx)^k$$

and afterwards take the limit $\varepsilon \to 0$ to obtain the desired recurrence for $G_s(x)$. Unfortunately, we do not get the minimal-order recurrence, but a fourth-order one. This has allowed us to reduce our computation time to about 11 minutes.

However, this is not yet the end of the story. In Fig. 1 we were trapped by Mathematica's definition of the binomial coefficient (cf. the discussion in Section 2): actually, the summand is zero for $k \leq 0$ if we employ the intended "correct" definition, which implies that $\binom{n}{k}$ is zero unless $0 \leq k \leq n$. The conditions for the summand to be nonzero (implied by the three binomial coefficients) somehow correspond to the summation bounds (given by the three summation quantifiers), which is illustrated in the following table (it is actually a curiosity of this problem that we can find such correspondence):

Factor in Summand	Nonzero Range (B_1)	Summation Bounds (B_2)
$\binom{s}{r}$	$0 \leq r \leq s$	$1 \leq r \leq s$
$\binom{k-1}{r-1}$	$0 \leq r-1 \leq k-1$	$1 \leq k \leq m+s-1$
$\binom{r-1}{i}$	$0 \leq i \leq r-1$	$0 \leq i \leq r-1-(k-m)$

After close inspection of this table, it becomes evident that only one gamma correction is actually needed (and hence Fig. 1 does not show the true situation). We can therefore redefine

$$C(\varepsilon, i, k, r) := \binom{s}{r} \cdot \binom{k-1}{r-1} \cdot \binom{r-1}{i} \cdot \frac{\Gamma(r-i-(k-m)+\varepsilon)}{\Gamma(r-i-(k-m))}.$$

This observation speeds up the computations significantly, and the winning time is 30 seconds. Moreover, we obtain the minimal recurrence of order three. The reader may now wonder how we can tell the HolonomicFunctions package that this computation should be executed with a different definition of the binomial coefficient (that differs from Mathematica's)? The answer is: we do not have to, since it is completely irrelevant (from the viewpoint of the package), because both versions of the binomial coefficient satisfy the very same recurrence equations, as we have seen in Section 2! The difference only becomes relevant when we evaluate the summand at particular values (which is done outside of the package), e.g., when checking initial conditions.

4. Conclusions and Future Work

In this expository article we demonstrated the usage of the HolonomicFunctions package to deal with an intricate triple sum coming from an application in quasi-Monte Carlo integration. We had two goals in mind for this paper: first, we felt the need to deliver some technical details for a key lemma in [6], where only the main ideas of the computer algebra proof were mentioned; second, we wanted to provide a somewhat easy-to-digest tutorial for proving special function and combinatorial identities with the help of the computer by expounding on the difficulties that may arise in similar applications and

<table>
<tr><th>Approximate
Comp. Time</th><th>Strategies
Implemented</th><th>Result</th></tr>
<tr><td>30 hours</td><td>– split sums
– sing./comm. corrections
– closure properties</td><td rowspan="2">fifth-order recurrence in the ideal generated by the guessed recurrence</td></tr>
<tr><td>1.4 hours</td><td>– split sums
– sing./comm. corrections
– closure properties
– substitution speedup</td></tr>
<tr><td>11 minutes</td><td>– one triple sum
– two gamma insertions
– sing. corrections</td><td>fourth-order recurrence in the ideal generated by the guessed recurrence</td></tr>
<tr><td>30 seconds</td><td>– one triple sum
– one gamma insertion
– sing. corrections</td><td>same third-order recurrence as the guessed recurrence</td></tr>
</table>

Figure 2. Results and Comparisons

a highlighting a few creative ways to cure them. We hope that we have convinced the reader that it is not always so cut and dry to prove a given identity with the holonomic systems approach. While in principle it allows one to prove holonomic identities in an automated way, we have seen that in practice, even with using current state-of-the-art software tools, many steps in the proof require human interaction. At many positions in the proving process, we had to make a choice on how to proceed, and the decision may both influence the optimality of the final result and the time that is required to obtain it. Fig. 2 gives an impressive overview how much difference in runtime such choices can make. In the future we plan to automate some of the proof steps that had to be done "by hand" in this case study, for example, in the analysis of singularities in the certificate(s) and dealing with the issue of commutation.

Acknowledgements

We would like to thank the organizers of CASC 2020 for providing an occasion and opportunity to give a talk about the work on which this article is based. We were encouraged by the positive feedback from the audience, which motivated this post-proceedings contribution. We also express our appreciation to Hao Du and Ali Uncu for their support and helpful commentary.

References

[1] Kauers, M.: Guessing handbook. Tech. Rep. 09-07, RISC Report Series, Johannes Kepler University, Linz, Austria (2009), http://www.risc.jku.at/research/combinat/software/Guess/

[2] Koutschan, C.: HolonomicFunctions (users guide). Tech. Rep. 10-01, RISC Report Series, Johannes Kepler University, Linz, Austria (2010), http://www.risc.jku.at/research/combinat/software/HolonomicFunctions/

[3] Koutschan, C., Wong, E.: Mathematica implementations for this paper in the notebook CTsums.nb and for [6] in the notebook computations.nb. https://wongey.github.io/digital-nets-walsh/

[4] Schneider, C.: Symbolic summation assists combinatorics. Séminaire Lotharingien de Combinatoire **56**, 1–36 (2007), article B56b, http://www.risc.jku.at/research/combinat/software/Sigma/

[5] Wiart, J., Lemieux, C., Dong, G.: On the dependence structure of scrambled (t, m, s)-nets. arXiv:1903.09877 (2019)

[6] Wiart, J., Wong, E.: Walsh functions, scrambled $(0, m, s)$-nets, and negative covariance: applying symbolic computation to quasi-Monte Carlo integration. Mathematics and Computers in Simulation (2020), to appear. E-print available on arXiv:2006.06225.

[7] Zeilberger, D.: A holonomic systems approach to special functions identities. Journal of Computational and Applied Mathematics **32**, 321–368 (1990)

[8] Zeilberger, D.: The method of creative telescoping. Journal of Symbolic Computation **11**(3), 195–204 (1991)

Christoph Koutschan
RICAM, Austrian Academy of Sciences
Altenberger Straße 69
4040 Linz
Austria
e-mail: `christoph.koutschan@ricam.oeaw.ac.at`

Elaine Wong
RICAM, Austrian Academy of Sciences
Altenberger Straße 69
4040 Linz
Austria
e-mail: `elaine.wong@ricam.oeaw.ac.at`