

Modern Summation Methods for Loop Integrals in Quantum Field Theory: The Packages `Sigma`, `EvaluateMultiSums` and `SumProduction`

C Schneider

Research Institute for Symbolic Computation (RISC)

Johannes Kepler University Linz

Altenbergerstr. 69, 4040 Linz, Austria

E-mail: `Carsten.Schneider@risc.jku.at`

Abstract. A large class of Feynman integrals, like e.g., two-point parameter integrals with at most one mass and containing local operator insertions, can be transformed to multi-sums over hypergeometric expressions. In this survey article we present a difference field approach for symbolic summation that enables one to simplify such definite nested sums to indefinite nested sums. In particular, the simplification is given –if possible– in terms of harmonic sums, generalized harmonic sums, cyclotomic harmonic sums or binomial sums. Special emphasis is put on the developed Mathematica packages `Sigma`, `EvaluateMultiSums` and `SumProduction` that assist in the task to perform these simplifications completely automatically for huge input expressions.

1. Introduction

This survey article aims at giving an overview of the available summation tools and packages in the setting of difference fields and aims at providing the basic insight how they work and how they can be applied to Feynman integrals.

The symbolic summation approach under consideration started with Karr’s telescoping algorithm in $\Pi\Sigma^*$ -fields [1]. There one can treat indefinite nested product-sum expressions. More precisely, one can search for optimal sum representations for these expressions such that the arising sums have minimal nesting depth [2–4], the polynomial expressions in the sums have minimal degrees [5, 6] and such that the sums used are algebraically independent [7, 8]. In particular, one can discover and prove definite sums over such expressions by exploiting the summation paradigms of parameterized telescoping and recurrence finding [9, 10]. In a nutshell, this toolkit implemented in the summation package `Sigma` [11] can be considered as a generalization of the well known hypergeometric summation machinery [12–14]. Meanwhile these difference field tools have been applied successfully, e.g., in combinatorics, number theory or numerics; for instance, we refer to [15–18].

In the last 7 years I have pushed forward these summation technologies and have developed new packages [8, 19–21] to carry out challenging calculations in particle physics. In cooperation with J. Blümlein (DESY, Zeuthen), e.g., Feynman integrals with at most one mass have been considered. They are defined in D -dimensional Minkowski space with one time- and $(D - 1)$ Euclidean space dimensions with $\varepsilon = D - 4 \in \mathbb{R}$ where $|\varepsilon| \ll 1$. The integrals depend on a discrete

Mellin parameter n which comes from local operator insertions. As worked out in [22, 23] these integrals can be transformed to integrals of the form

$$\mathcal{I}(\varepsilon, n) = C(\varepsilon, n, M) \int_0^1 dx_1 \dots \int_0^1 dx_m \frac{\sum_{i=1}^k \prod_{l=1}^{r_i} [P_{i,l}(x_1, \dots, x_m)]^{\alpha_{i,l}(\varepsilon, n)}}{[Q(x_1, \dots, x_m)]^{\beta(\varepsilon)}}, \quad (1)$$

with $k \in \mathbb{N} = \{0, 1, 2, \dots\}$, $r_1, \dots, r_k \in \mathbb{N}$ and the following ingredients. $C(\varepsilon, n, M)$ is a factor, which depends on the dimensional parameter ε , the integer parameter n and the mass M (being possible 0). The $P_i(x_1, \dots, x_m)$ and $Q(x_1, \dots, x_m)$ are polynomials in the integration variables x_i . Further, the exponent $\beta(\varepsilon)$ is a rational function in ε , i.e., $\beta(\varepsilon) \in \mathbb{Q}(\varepsilon)$, and similarly $\alpha_{i,l}(\varepsilon, n) = n_{i,l}n + \bar{\alpha}_{i,l}$ with $n_{i,l} \in \{0, 1\}$ and $\bar{\alpha}_{i,l} \in \mathbb{Q}(\varepsilon)$. For integrals without local operator insertions see also [24, 25].

Then given such an integral we seek for a Laurent series expansion for some $t \in \mathbb{Z}$:

$$\mathcal{I}(\varepsilon, n) = I_t(n)\varepsilon^t + I_{t+1}(n)\varepsilon^{t+1} + I_{t+2}(n)\varepsilon^{t+2} + \dots \quad (2)$$

More precisely, the crucial task is to determine the first coefficients $I_i(n)$ in closed form.

Remark. There are computer algebra tools [19] available to solve this problem for integrals of the form (1). Namely, we can calculate a recurrence in n for (1) using Ablinger's `MultiIntegrate` package [26], an optimized and refined version of the Almkvist–Zeilberger algorithm [27]. Given such a recurrence, we can calculate the desired ε -expansion by algorithms [23] given in `Sigma`. We remark that the method of hyperlogarithms [28] can be also used to evaluate integrals of the form (1) for specific values $n \in \mathbb{N}$ if one can set $\varepsilon = 0$. An adaption of this method for symbolic n has been described and applied to massive 3-loop ladder graphs in [29].

For some types of integrals these approaches work nicely. However, for many cases it was more suitable to proceed as follows. Using the method in [23] the given integral (1) can be transformed to proper hypergeometric multi-sums of the format¹

$$\mathcal{S}(\varepsilon, n) = \sum_{n_1=1}^{\infty} \dots \sum_{n_r=1}^{\infty} \sum_{k_1=1}^{L_1(n)} \dots \sum_{k_v=1}^{L_v(n, k_1, \dots, k_{v-1})} \sum_{i=1}^l C_i(\varepsilon, n, M) \frac{\Gamma(t_{1,i}) \dots \Gamma(t_{v',i})}{\Gamma(t_{v'+1,i}) \dots \Gamma(t_{w',i})}. \quad (3)$$

Here the upper bounds $L_1(n), \dots, L_v(n, k_1, \dots, k_{v-1})$ are ∞ or integer linear expressions in the dependent parameters (i.e., linear combinations of the variables over the integers), and $t_{l,i}$ are linear combinations of the $n_1, \dots, n_r, k_1, \dots, k_v, \varepsilon$ over \mathbb{Q} .

Remark. At this level, one can apply similar techniques as for the integral representation (1). Namely, as carried out in [23] we can calculate recurrence relations for such multi-sums (3) by using either the WZ-approach [30] and efficient and refined algorithms developed in [31]. Similarly, we can use a common framework [32] within the summation package `Sigma` that combines difference field [33] and holonomic summation techniques [34] to compute recurrences for the given sum (3). Then given such a recurrence, one can use again `Sigma`'s recurrence solver to calculate the coefficients of the Laurent-series expansion (2).

Finding recurrence relations for the integrals of the form (1) and sums of the form (3) involving the ε -parameter is a rather tough problem. However, we can get rid of the parameter ε , if the sums (3) are uniformal convergent: First one expands the summand of (3), say

$$F(n, n_1, \dots, n_r, k_1, \dots, k_v) = F_t(n, n_1, \dots, k_v)\varepsilon^t + F_{t+1}(n, n_1, \dots, k_v)\varepsilon^{t+1} + \dots \quad (4)$$

¹ For convenience, we assume that the summand is written in terms of the Gamma function $\Gamma(x)$. Later, also Pochhammer symbols or binomial coefficients are used which can be rewritten in terms of Gamma-functions.

with $t \in \mathbb{Z}$ by using formulas such as [23]

$$\Gamma(k+1+\bar{\varepsilon}) = \frac{\Gamma(k)\Gamma(1+\bar{\varepsilon})}{B(k,1+\bar{\varepsilon})} \text{ and } B(k,1+\bar{\varepsilon}) = \frac{1}{k} \exp\left(\sum_{i=1}^{\infty} \frac{(-\bar{\varepsilon})^i}{i} S_i(k)\right) = \frac{1}{k} \sum_{i=0}^{\infty} (-\bar{\varepsilon})^i \underbrace{S_{1,\dots,1}}_i(k)$$

with $\bar{\varepsilon} = r\varepsilon$ for some $r \in \mathbb{Q}$. Here $B(x,y) = \Gamma(x)\Gamma(y)/\Gamma(x+y)$ denotes the Beta-function and $S_{1,\dots,1}(n)$ is a special instance of the harmonic sums [35, 36] defined by

$$S_{c_1,\dots,c_r}(k) = \sum_{i_1=1}^k \frac{\text{sign}(c_1)^{i_1}}{i_1^{|c_1|}} \sum_{i_2=1}^{i_1} \frac{\text{sign}(c_2)^{i_2}}{i_2^{|c_2|}} \dots \sum_{i_r=1}^{i_{r-1}} \frac{\text{sign}(c_r)^{i_r}}{i_r^{|c_r|}} \quad (5)$$

with c_1, \dots, c_r being nonzero integers. If there are no poles² within the summation range, one can apply the summation signs to each of the coefficients in (4). Namely the i th coefficient of the ε -expansion (2) of (3) can be written in the form

$$I_i(n) = \sum_{n_1=1}^{\infty} \dots \sum_{n_r=1}^{\infty} \sum_{k_1=1}^{L_1(n)} \dots \sum_{k_v=1}^{L_v(n, k_1, \dots, k_{v-1})} \sum_{i=1}^l F_i(n, n_1, \dots, n_r, k_1, \dots, k_v) \quad (6)$$

where the summand is given by hypergeometric terms (e.g., in terms of the Gamma-function) and harmonic sums that arise in the numerators. Then the essential problem is the simplification of these sums to special functions, like harmonic sums (5), generalized harmonic sums [37, 38]

$$S_{c_1,\dots,c_r}(x_1, \dots, x_r; k) = \sum_{i_1=1}^k \frac{x_1^{i_1}}{i_1^{|c_1|}} \sum_{i_2=1}^{i_1} \frac{x_2^{i_2}}{i_2^{|c_2|}} \dots \sum_{i_r=1}^{i_{r-1}} \frac{x_r^{i_r}}{i_r^{|c_r|}} \quad (7)$$

with $c_i \in \mathbb{N} \setminus \{0\}$ and $x_i \in \mathbb{R} \setminus \{0\}$, or cyclotomic harmonic sums [39]

$$S_{(a_1, b_1, c_1), \dots, (a_r, b_r, c_r)}(x_1, \dots, x_r; k) = \sum_{i_1=1}^k \frac{x_1^{i_1}}{(a_1 i_1 + b_1)^{c_1}} \sum_{i_2=1}^{i_1} \frac{x_2^{i_2}}{(a_2 i_2 + b_2)^{c_2}} \dots \sum_{i_r=1}^{i_{r-1}} \frac{x_r^{i_r}}{(a_r i_r + b_r)^{c_r}} \quad (8)$$

where the a_i, c_i are positive integers, the b_i are non-negative integers with $a_i > b_i$, and $x_i \in \mathbb{R} \setminus \{0\}$. More generally, also binomial sums might arise [40]; for a detailed survey see [41]. For special cases (i.e., if the Gamma-functions in (3) arise in certain form), this simplification can be carried out with efficient methods; see, e.g., [35, 37, 42]. For more general classes the algorithms and packages presented in this article have been heavily used [29, 40, 43–52].

The backbone of all these calculations are the summation tools of **Sigma** based on difference fields. In Section 2 we will get a glimpse of how the difference field machinery works and elaborate the crucial summation techniques based on this approach.

In order to apply these tools systematically, the package **EvaluateMultiSums** [8, 21] has been developed. In this way, one obtains a completely automatic method to simplify multi-sums of the form (6) in terms of indefinite nested product-sums expressions covering as special cases the sums (5), (7) and (8). The underlying ideas of this package are presented in Section 3.

During our calculations in [40, 49, 52] we were faced with expressions with several thousand multi-sums. In order to assist this mass-production the package **SumProduction** [20] has been developed. In particular, it contains a multi-sum simplifier (again based on the package **Sigma**) that crunches such large expressions to few master sums and the **EvaluateMultiSums** package is only applied to those sums. In addition, a useful framework is provided to apply all the calculations in parallel on distributed machines. This latter package is presented in Section 4. A conclusion of the symbolic summation approach in difference fields and the produced packages is given in Section 5.

² If there are poles, these extra evaluations are treated separately by first plugging the values into f and by expanding this expression afterwards.

2. Symbolic summation in difference fields: the Sigma package

An important milestone of symbolic summation is Gosper's telescoping algorithm [53]: given a hypergeometric expression³ $f(k)$, it finds –in case of existence– a hypergeometric expression $g(k)$ such that the telescoping equation

$$f(k) = g(k+1) - g(k) \quad (9)$$

holds. Then given $g(k)$, one can sum (9) over k and obtains, e.g., the identity

$$\sum_{k=1}^a f(k) = g(a+1) - g(1). \quad (10)$$

Moreover, the breakthrough concerning applications was lead by Zeilberger's extension of Gosper's algorithm to creative telescoping [12] in the framework of his holonomic system approach [54]: it enables one to derive recurrence relations for definite hypergeometric sums. In particular, solving such recurrences in terms of hypergeometric expressions [13] gave rise to the following toolbox: given a definite proper hypergeometric sum, one can decide algorithmically if it can be simplified in terms of a linear combination of hypergeometric expressions. For details on this machinery we refer to the pioneering book [14]; for a most recent point of view see [55]. In the last decades many further improvements and generalizations have been accomplished, like for holonomic sequences [34, 56], for non-holonomic sequences like the Stirling numbers [57] or for expressions represented in terms of difference fields. The latter approach started with Karr's telescoping algorithm in $\Pi\Sigma^*$ -fields [1]. There one can treat indefinite nested product-sums in a very elegant way; for details see [8].

Definition. Let $f(k)$ be an expression that evaluates at non-negative integers (from a certain point on) to elements of a field \mathbb{K} containing as subfield the rational numbers \mathbb{Q} . Then $f(k)$ is called indefinite nested product-sum expression w.r.t. k (over \mathbb{K}) if it is composed by elements from the rational function field $\mathbb{K}(k)$, the four operations $(+, -, \cdot, /)$, and indefinite sums and products of the type $\sum_{i=l}^k h(i)$ or $\prod_{i=l}^k h(i)$ where $l \in \mathbb{N}$ and where $h(i)$ is an indefinite nested product-sum expression w.r.t. i over \mathbb{K} which is free of k .

Typical examples are the sums given in (5), (7) and (8).

2.1. The basic mechanism in difference fields

The basic ideas in the setting of difference fields are presented. This part can be omitted by those readers who are mainly interested in the application of the summation tools.

Consider the following indefinite summation problem: simplify $\sum_{k=1}^n f(k)$ with $f(k) = k S_1(k)$ where $S_1(k) = \sum_{i=1}^k \frac{1}{i}$ denotes the k -th harmonic numbers. To accomplish this task, we hunt for a solution $g(k)$ in terms of $S_1(k)$ such that (9) holds. In terms of the shift operator \mathcal{S}_k w.r.t. k equation (9) reads as follows:

$$f(k) = \mathcal{S}_k g(k) - g(k). \quad (11)$$

First, the summation objects will be represented step by step in a field \mathbb{F} , and along with that the shift operator \mathcal{S}_k is rephrased by a field automorphism $\sigma : \mathbb{F} \rightarrow \mathbb{F}$.

- i) We start with the rational numbers \mathbb{Q} and define the (only possible) field automorphism $\sigma : \mathbb{Q} \rightarrow \mathbb{Q}$ with $\sigma(q) = q$ for all $q \in \mathbb{Q}$.

³ A sequence $f(k)$ (resp. an expression that evaluates to a sequence) is called hypergeometric if there is a rational function $r(x)$ and a $\lambda \in \mathbb{N}$ such that $r(k) = \frac{f(k+1)}{f(k)}$ for all $k \in \mathbb{N}$ with $k \geq \lambda$.

- ii) Next, we need to model k with the shift behavior $\mathcal{S}_k k = k + 1$: Since all elements in \mathbb{Q} are constant (i.e., $\sigma(q) = q$ for all $q \in \mathbb{Q}$), we adjoin a variable t_1 to \mathbb{Q} and extend the automorphism to $\sigma : \mathbb{Q}(t_1) \rightarrow \mathbb{Q}(t_1)$ with $\sigma(t_1) = t_1 + 1$. In other words, for $f \in \mathbb{Q}(t_1)$ the element $\sigma(f)$ is obtained by replacing any occurrence of t_1 by $t_1 + 1$.
- iii) Finally, we represent $S_1(k)$ with the shift behavior $\mathcal{S}_k S_1(k) = S_1(k) + \frac{1}{k+1}$: First, we try to find such an element in $\mathbb{Q}(t_1)$. Namely we look for a $\gamma \in \mathbb{Q}(t_1)$ such that $\sigma(\gamma) = \gamma + \frac{1}{t_1+1}$ or equivalently $\sigma(\gamma) - \gamma = \frac{1}{t_1+1}$ holds. Using our telescoping algorithms from [58] (or, e.g., Gosper's algorithm) proves that such an element γ does not exist. Therefore we adjoin the variable t_2 to $\mathbb{Q}(t_1)$ and extend the automorphism $\sigma : \mathbb{Q}(t_1)(t_2) \rightarrow \mathbb{Q}(t_1)(t_2)$ subject to $\sigma(t_2) = t_2 + \frac{1}{t_1+1}$. In other words, for $f \in \mathbb{Q}(t_1)(t_2)$ the element $\sigma(f)$ can be calculated by replacing any occurrences of t_1 by $t_1 + 1$ and of t_2 by $t_2 + \frac{1}{t_1+1}$.

In short, we have constructed a difference field (\mathbb{F}, σ) with the rational function field $\mathbb{F} = \mathbb{Q}(t_1)(t_2)$ together with a field automorphism σ . There k and $S_1(k)$ are represented by t_1 and t_2 , respectively, and the shift operator \mathcal{S}_k with $\mathcal{S}_k k = k + 1$ and $\mathcal{S}_k S_1(k) = S_1(k) + \frac{1}{k+1}$ is reflected by σ . In this setting, $f(k)$ is given by $\phi = t_1^2 t_2$ and one seeks $\gamma \in \mathbb{Q}(t_1)(t_2)$ such that $\phi = \sigma(\gamma) - \gamma$. With our algorithm we calculate $\gamma = \frac{1}{4}(t_1 - 1)t_1(2t_2 - 1)$ which delivers the solution $g(k) = \frac{1}{4}(k - 1)k(2S_1(k) - 1)$ for (11) and thus for (9). As a consequence we get (10) which produces the simplification

$$\sum_{k=1}^n k S_1(k) = \frac{1}{4}(2n(n+1)S_1(n) - (n-1)n). \quad (12)$$

Summarizing, we applied the following strategy; for more details we refer to [8].

- (1) Represent the involved indefinite nested product-sum expressions, whose evaluation leads to elements from a field \mathbb{K} , in a difference field (\mathbb{F}, σ) . Here $\mathbb{F} = \mathbb{K}(t_1) \dots (t_e)$ is a rational function field where the generators t_i represent the sums and products. Moreover, the shift behavior of the objects is described by a field automorphism $\sigma : \mathbb{F} \rightarrow \mathbb{F}$ where for $1 \leq i \leq e$ either the sum relation $\sigma(t_i) = t_i + a_i$ or the product relation $\sigma(t_i) = a_i t_i$ with $0 \neq a_i \in \mathbb{F}_{i-1} := \mathbb{K}(t_1) \dots (t_{i-1})$ hold. As indicated in the example above, it is crucial that a new variable t_i with $\sigma(t_i) = t_i + a_i$ (similar for products) is only adjoined to \mathbb{F}_{i-1} if there is no $\gamma \in \mathbb{F}_{i-1}$ with $\sigma(\gamma) = \gamma + a_i$. Exactly this problem is solvable [1]; for improved algorithms see [3, 58] and references therein.
- (2) Solve the underlying summation problem (e.g., again telescoping, but also parameterized telescoping and recurrence solving given below) in this setting.
- (3) Reformulate the solution to an expression in terms of indefinite nested product-sum expressions that yields a solution of the given summation problem.

Exactly this construction produces difference fields $(\mathbb{K}(t_1) \dots (t_e), \sigma)$ whose constants remain unchanged; see [1, 9, 59]. Difference fields with this property are also called $\Pi\Sigma^*$ -fields. Formally, they can be defined as follows.

Definition. Let \mathbb{F} be a field with characteristic 0 (i.e., the rational numbers are contained as sub-field) and let σ be a field automorphism of \mathbb{F} . The constant field of \mathbb{F} is defined by $\mathbb{K} = \{f \in \mathbb{F} | \sigma(f) = f\}$. A difference field (\mathbb{F}, σ) with constant field \mathbb{K} is called a $\Pi\Sigma^*$ -field if $\mathbb{F} = \mathbb{K}(t_1) \dots (t_e)$ where for all $1 \leq i \leq e$ each $\mathbb{F}_i = \mathbb{K}(t_1) \dots (t_i)$ is a transcendental field extension of $\mathbb{F}_{i-1} = \mathbb{K}(t_1) \dots (t_{i-1})$ (we set $\mathbb{F}_0 = \mathbb{K}$) and σ has the property that $\sigma(t_i) = a t_i$ or $\sigma(t_i) = t_i + a$ for some $a \in \mathbb{F}_{i-1}$.

In conclusion, all the summation paradigms presented in the next subsections rely on this mechanism: Reformulate the given problem in a $\Pi\Sigma^*$ -field (or ring), solve it there and formulate the result back such that it is a solution of the input problem. In the following we will give an overview of Sigma's symbolic summation toolbox that is based on the difference field (resp. ring) approach introduced above.

2.2. Simplification of indefinite nested product-sum expressions

Whenever **Sigma** deals with indefinite nested product-sum expressions, in particular when it outputs such expressions, the following problem is solved implicitly.

Problem EAR: Elimination of algebraic relations. *Given* an indefinite nested product-sum expression $f(k)$. *Find* an indefinite nested product-sum expression $F(k)$ and $\lambda \in \mathbb{N}$ such that $f(k) = F(k)$ for all $k \geq \lambda$ and such that the occurring sums are algebraically independent.

As worked out above, the following mechanism is applied: $f(k)$ is rephrased in a suitable $\Pi\Sigma^*$ -field $(\mathbb{K}(t_1) \dots (t_e), \sigma)$ (or ring); this is accomplished by solving iteratively the telescoping problem. Here the sums (similarly the products) are represented by the variables t_i . Finally, reinterpreting the t_i as sums produces an alternative expression $F(k)$ of $f(k)$ where the occurring sums are algebraically independent; the solution to this problem relies on results of [7] and is worked out in detail in [8]. We remark that that the found relations for harmonic sums, cyclotomic sums and generalized harmonic sums coincide with the derived relations [38, 39, 60, 61] that are obtained by using the underlying quasi-shuffle algebras [35, 37, 62].

After loading the **Sigma** package into Mathematica:

```
In[1]:= << Sigma.m
        Sigma - A summation package by Carsten Schneider © RISC
```

the simplification in (12) can be accomplished as follows⁴:

```
In[2]:= SigmaReduce[Sum[k S[1, k], n]
Out[2]= 1/4 (2n(n+1)S[1, n] - (n-1)n)
```

Here the sum $\sum_{k=1}^n k S_1(k)$ has been reduced in terms of the objects k and $S_1(k)$. In particular, by difference field theory it follows that the sequence given by $S_1(k)$ is transcendental (resp. algebraically independent) over the rational sequences, more precisely, the sequences that one obtains by evaluating the elements of $\mathbb{Q}(k)$.

In particular, using improved telescoping algorithms, the underlying $\Pi\Sigma^*$ -field can be constructed in such a way that the sums are given with certain optimality criteria.

Refined telescoping. *Given* an indefinite nested product-sum expression $f(k)$. *Find* an indefinite nested product-sum expression $g(k)$ such that (9) holds and such that $g(k)$ is as “simple” as possible.

Then summing the found equation (9) over k yields, e.g., the identity (10) where also the right hand side is as simple as possible. Subsequently, we present the main features of **Sigma**; some of these simplifications are carried out by default, some must be activated explicitly (see below).

2.2.1. Sum representations with optimal nesting depth The found expression $F(k)$ can be given with minimal nesting depth and for any occurring sum in $F(k)$ there is no other indefinite nested sum representation with lower nesting depth; for details see [2–4]. A typical example is as follows.

```
In[3]:= SigmaReduce[2^n Sum[1/i Sum[1/2^j Sum[2^k/k Sum[1/2^l], n], n], n]
Out[3]= 2^n ((Sum[1/i, n])^2 + (Sum[1/2^i, n]) Sum[1/i, n] + Sum[1/i^2, n] + Sum[1/2^i i^2, n] - 3 Sum[Sum[1/2^j, j=1, i], i=1, n] - Sum[Sum[1/2^i, i=1, j], j=1, n])
```

Here the product $2^n = \prod_{i=1}^n 2$ and the arising sums are algebraically independent over the rational sequences, and the sums have minimal nesting depth. In particular, the expression

⁴ Here $S[c_1, \dots, c_r, k]$ denotes the harmonic sum (5).

in `Out[3]` cannot be written in terms of indefinite nested sums with lower nesting depth. Similarly, the following sum expression can be reduced to its optimal nesting depth.

$$\begin{aligned} \text{In[4]} &:= \text{SigmaReduce}\left[\sum_{k=0}^a \left(\sum_{j=0}^k \frac{x^j}{\binom{n}{j}}\right)^2\right] \\ \text{Out[4]} &= \frac{-2x^{a+1}(a+1)}{(x+1)\binom{n}{a}} \sum_{i_1=0}^a \frac{x^{i_1}}{\binom{n}{i_1}} + \frac{(n+x+1)}{x+1} \sum_{i_1=0}^a \frac{x^{2i_1}}{\binom{n}{i_1}^2} + \frac{(x-1)}{x+1} \sum_{i_1=0}^a \frac{x^{2i_1} i_1}{\binom{n}{i_1}^2} + \frac{(ax+a-n+2x)}{x+1} \left(\sum_{i_1=0}^a \frac{x^{i_1}}{\binom{n}{i_1}}\right)^2 \end{aligned}$$

We emphasize that this depth-optimal representation leads to efficient algorithms for telescoping, creative telescoping and recurrence solving; for details we refer to [3, 63]. Thus by efficiency reasons this feature of `Sigma` is activated by default.

2.2.2. Reducing the number of objects and the degrees in the summand The depth-optimal representation can be refined further as follows: *find* an alternative sum representation such that for the outermost summands the number of occurring objects is as small as possible. This problem was originally solved in [5]. For a more efficient and simplified algorithm see [58]. In the following example we can eliminate $S_1(k)$ from the summand:

$$\begin{aligned} \text{In[5]} &:= \text{SigmaReduce}\left[\sum_{k=0}^a (-1)^k S[1, k]^2 \binom{n}{k}, \text{a, SimplifyByExt} \rightarrow \text{DepthNumber}\right] \\ \text{Out[5]} &= -(a-n)(n^2 S[1, a]^2 + 2n S[1, a] + 2) \frac{(-1)^a \binom{n}{a}}{n^3} - \frac{2}{n^2} - \frac{1}{n} \sum_{i_1=1}^a \frac{(-1)^{i_1}}{i_1} \binom{n}{i_1} \end{aligned}$$

Furthermore, one can calculate representations such that the degrees (w.r.t. the top extension of a $\Pi\Sigma^*$ -field) in the numerators and denominators of the summands are minimal [6]:

$$\begin{aligned} \text{In[6]} &:= \text{SigmaReduce}\left[\sum_{k=0}^a (-1)^k S[1, k]^3 \binom{n}{k}\right] \\ \text{Out[6]} &= -\frac{3}{n^2} \sum_{i_1=1}^a \frac{(-1)^{i_1} \binom{n}{i_1}}{i_1} - \frac{3}{n} \sum_{i_1=1}^a \frac{(-1)^{i_1} \binom{n}{i_1} S[1, i_1]}{i_1} + \frac{1}{n} \sum_{i_1=1}^a \frac{(-1)^{i_1} \binom{n}{i_1}}{i_1^2} + (-1)^a \binom{n}{a} \left(\frac{6(n-a)}{n^4} + \frac{6(n-a)S[1, a]}{n^3} - \frac{3(a-n)S[1, a]^2}{n^2} + \frac{(n-a)S[1, a]^3}{n}\right) - \frac{6}{n^3} \end{aligned}$$

For algorithms dealing with the product case we point to [64, 65].

2.2.3. Minimal degrees w.r.t. the summation index By default `Sigma` outputs sums such that the denominators have minimal degrees w.r.t. the summation index (i.e., if possible, the denominator w.r.t. the summation index is linear). This feature is of particular importance to rewrite sums in terms of harmonic sums and their generalized versions. A typical example is

$$\begin{aligned} \text{In[7]} &:= \text{SigmaReduce}\left[\sum_{k=1}^a \left(\frac{-2+k}{10(1+k^2)} + \frac{(1-4k-2k^2)S[1, k]}{10(1+k^2)(2+2k+k^2)} + \frac{(1-4k-2k^2)S[3, k]}{5(1+k^2)(2+2k+k^2)}\right), \text{a}\right] \\ \text{Out[7]} &= \frac{a^2+4a+5}{10(a^2+2a+2)} S[1, a] - \frac{(a-1)(a+1)}{5(a^2+2a+2)} S[3, a] - \frac{2}{5} \sum_{k=1}^a \frac{1}{k^2} \end{aligned}$$

2.3. The summation paradigms for definite summation

Definite sums over indefinite nested product sums⁵, like

$$A(n) = \sum_{k=0}^n \binom{n}{k} S_1(k)^2 = \sum_{k=0}^n \left(\prod_{i=1}^k \frac{n+1-i}{i}\right) \left(\sum_{i=1}^k \frac{1}{i}\right)^2, \quad (13)$$

can be handled by the following summation paradigms (see Fig. 1).

2.3.1. Finding recurrences by parameterized (creative) telescoping First, there is the following tool in the setting of $\Pi\Sigma^*$ -fields to obtain recurrences; for the most recent summary see [58].

⁵ Definite means that the upper bound is ∞ or consists of parameters that occur also inside of the sum.

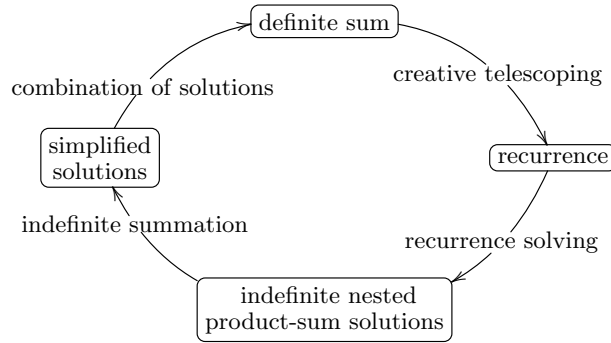


Figure 1. Sigma's summation spiral; see [10].

Problem PT: Parameterized Telescoping. Given indefinite nested product-sum expressions $f_0(k), \dots, f_\delta(k)$. Find constants a_0, \dots, a_δ , not all 0 and all free of k , and find an indefinite nested product-sum expression $g(k)$ being not more complicated than the $f_i(k)$ such that

$$a_0 f_0(k) + a_1 f_1(k) + \dots + a_\delta f_\delta(k) = g(k+1) - g(k). \quad (14)$$

For simplicity suppose that the found relation (14) holds for $0 \leq k \leq a$. Then by telescoping one gets, e.g., the sum relation

$$a_0 \sum_{k=0}^a f_0(k) + a_1 \sum_{k=0}^a f_1(k) + \dots + a_\delta \sum_{k=0}^a f_\delta(k) = g(a+1) - g(0) \quad (15)$$

where the right hand side is simpler than the sums of the left hand side.

Specializing to $f_i(k) := f(n+i, k)$ for a bivariate expression yields the creative telescoping paradigm. Here one loops over⁶ $\delta = 0, 1, 2, \dots$ and tries to solve the corresponding parameterized telescoping problem. If the method stops, then we can deduce (15), and we get the recurrence

$$a_0(n) A'(n) + a_1(n) A'(n+1) + \dots + a_\delta(n) A'(n+\delta) = g(a+1) - g(0)$$

for the sum $A'(n) = \sum_{k=0}^a f(n, k)$. To this end, specializing $a = n$ and taking care of extra terms yield a recurrence for the sum $A(n) = \sum_{k=0}^n f(n, k)$ of the form

$$a_0(n) A(n) + a_1(n) A(n+1) + \dots + a_\delta(n) A(n+\delta) = h(n). \quad (16)$$

For instance, take $f_i(k) = \binom{n+i}{k} S_1(k)^2 = \prod_{j=1}^i \frac{n+j}{n-k+j} \binom{n}{k} S_1(k)^2$. Then we find a parameterized telescoping solution (14) for $\delta = 4$. Performing the steps above, we finally get a recurrence of order 4 for our sum (13). All these steps can be carried out for $A(n) = \text{SUM}[n]$ as follows:

$$\text{In}[8]:= \text{mySum} = \sum_{k=1}^n \binom{n}{k} S[1, k]^2;$$

$$\text{In}[9]:= \text{rec} = \text{GenerateRecurrence}[\text{mySum}, n][[1]]$$

$$\text{Out}[9]= 8(n+1)(n+3)\text{SUM}[n] - 4(5n^2 + 25n + 29)\text{SUM}[n+1] + 2(3n+8)(3n+10)\text{SUM}[n+2] \\ - (7n^2 + 49n + 86)\text{SUM}[n+3] + (n+4)^2\text{SUM}[n+4] = 1$$

We remark that the refined telescoping algorithms from Subsection 2.2 provide also refined tools for parameterized/creative telescoping [3, 58]: For our sum (13) we can calculate a recurrence of order 2 instead of order 4 by using extra sum extensions:

⁶ Note that the special case $\delta = 0$ is telescoping.

$$\text{In}[10]:= \text{GenerateRecurrence}[\text{mySum}, n, \text{SimplifyByExt} \rightarrow \text{DepthNumber}][[1]]$$

$$\text{Out}[10]= 4(n+1)\text{SUM}[n] - 2(2n+3)\text{SUM}[n+1] + (n+2)\text{SUM}[n+2] == \frac{(3n+4)}{n+2} \sum_{i_1=0}^n \frac{\binom{n}{i_1}}{1+n-i_1} + \sum_{i_1=1}^n \frac{\binom{n}{i_1}}{i_1} + \frac{1}{n+2}$$

Note that the found sums on the right hand side are definite. Simplifying these sums (by using just the methods that we describe here), we end up at the recurrence

$$4(n+1)A(n) - 2(2n+3)A(n+1) + (n+2)A(n+2) = S_1(2; n) - S_1(n) + \frac{2^{n+1}(3n+4) - (2n+3)}{(n+1)(n+2)}.$$

Usually such shorter recurrences are easier to solve. In order to demonstrate the summation tools below, we will continue with the recurrence given in Out[9].

2.3.2. Solving recurrences Next, we can apply the following recurrence solver [66] which is based on [33, 67] and generalizes ideas of [13, 68].

Problem RS: Recurrence Solving.

Given a recurrence of the form (16) where the coefficients $a_i(n)$ and $h(n)$ are given in terms of indefinite nested product-sum expressions. *Find* all solutions that are expressible in terms of indefinite nested product sum expressions.

This solver finds, if possible, a particular solution of (16) in terms of indefinite nested product-sum expressions; and it finds a linear independent set of expressions in terms of indefinite nested product-sums with the following property: their linear combinations produce all solutions of the homogeneous version of (16) that can be expressed in terms of indefinite nested product-sum expressions. The sequences generated by these solutions are called d'Alembertian solutions, a subclass of Liouvillian solutions [69]. For details dealing with the rational case see [70].

Our recurrence `rec=Out[9]` can be solved with the following command of **Sigma**:

$$\text{In}[11]:= \text{recSol} = \text{SolveRecurrence}[\text{rec}, \text{SUM}[n], \text{IndefiniteSummation} \rightarrow \text{False}]$$

$$\text{Out}[11]= \left\{ \{0, 2^n\}, \left\{0, 2^n \sum_{i=1}^n \frac{1}{i}\right\}, \left\{0, 2^n \sum_{i=1}^n \frac{1}{i} \sum_{j=1}^i \frac{1}{2^j}\right\}, \left\{0, 2^n \sum_{i=1}^n \frac{1}{i} \sum_{j=1}^i \frac{1}{2^j} \sum_{k=1}^j \frac{2^k}{k}\right\}, \left\{1, 2^n \sum_{i=1}^n \frac{1}{i} \sum_{j=1}^i \frac{1}{2^j} \sum_{k=1}^j \frac{2^k}{k} \sum_{l=1}^k \frac{1}{2^l}\right\} \right\}$$

More precisely, we obtain the general solution

$$2^n \left[c_1 + c_2 \sum_{i=1}^n \frac{1}{i} + c_3 \sum_{i=1}^n \frac{1}{i} \sum_{j=1}^i \frac{1}{2^j} + c_4 \sum_{i=1}^n \frac{1}{i} \sum_{j=1}^i \frac{1}{2^j} \sum_{k=1}^j \frac{2^k}{k} + \sum_{i=1}^n \frac{1}{i} \sum_{j=1}^i \frac{1}{2^j} \sum_{k=1}^j \frac{2^k}{k} \sum_{l=1}^k \frac{1}{2^l} \right], \quad c_1, c_2, c_3, c_4 \in \mathbb{Q}.$$

By construction the solutions are highly nested: in particular the depth of the particular solution equals usually the recurrence order plus the nesting depth of the inhomogeneous part of the recurrence. It is therefore a crucial (and often the most challenging) task to simplify these solutions further by solving Problem EAR and applying the refined telescoping algorithms from Subsection 2.2. For the simplification of the particular solution see Out[3] from above. With **Sigma** the recurrence `rec` given in Out[9] is solved (see Problem RS) and the found solutions are simplified (with the default options) by the following function call:

$$\text{In}[12]:= \text{recSol} = \text{SolveRecurrence}[\text{rec}, \text{SUM}[n]]$$

$$\text{Out}[12]= \left\{ \{0, 2^n\}, \left\{0, 2^n \sum_{i=1}^n \frac{1}{i}\right\}, \left\{0, 2^n \left(2 \sum_{i=1}^n \frac{1}{i} - 2 \sum_{i=1}^n \frac{1}{2^i i}\right)\right\}, \left\{0, 2^n \left(\left(\sum_{i=1}^n \frac{1}{i}\right)^2 + \sum_{i=1}^n \frac{1}{i^2} - \sum_{i=1}^n \frac{1}{2^i i} \sum_{j=1}^i \frac{2^j}{j}\right)\right\}, \left\{1, 2^n \left(\left(\sum_{i=1}^n \frac{1}{i}\right)^2 + \sum_{i=1}^n \frac{1}{i^2} + \left(\sum_{i=1}^n \frac{1}{i}\right) \sum_{i=1}^n \frac{1}{2^i i} + \sum_{i=1}^n \frac{1}{2^i i^2} - \sum_{i=1}^n \frac{1}{2^i i} \sum_{j=1}^i \frac{2^j}{j} - 3 \sum_{i=1}^n \frac{1}{i} \sum_{j=1}^i \frac{1}{2^j j}\right)\right\} \right\}$$

Example. In [45, 46] we computed a large amount of initial values (up to 3500) of the massless Wilson coefficients to 3-loop order for individual color coefficients by taking the result given

in [71]. Next, we guessed recurrences of minimal order for these coefficients by using Kauer's package `Guess` [72]. Then, e.g., the largest recurrence of order 35 could be solved completely in about 1 day. This yielded 35 linearly independent solutions in terms of sums up to nesting depth 34. To this end, their simplifications in terms of harmonic sums took about 5 days.

2.3.3. Combining the solutions Finally, we take the linear combination of the homogeneous solutions (the entries with a 0) plus the particular solution (the entry with a 1) such that it agrees with $A(n)$ for $n = 1, 2, 3, 4$. This combination can be calculated by calling the function

```
In[13]:= sol = FindLinearCombination[recSol, mySum, n, 4]
Out[13]= 2^n ((Sum[1/i, {i, 1, n}]^2 + Sum[1/i^2, {i, 1, n}] + (Sum[1/i, {i, 1, n}] Sum[1/2^i, {i, 1, n}] + Sum[1/2^i, {i, 1, n}] - Sum[1/2^i, {i, 1, n}] Sum[2^j/j, {j, 1, i}] - 3 Sum[1/i, {i, 1, n}] Sum[1/2^j, {j, 1, i}])
```

Since the sum (13) and the derived expression in `Out[13]` agree for $n = 1, 2, 3, 4$ and since both are solutions of the recurrence `Out[9]`, they evaluate to the same sequence for all $n \in \mathbb{N}$.

In order to rewrite the found expression `Out[13]` in terms of harmonic sums and their generalized versions, we load in the `HarmonicSums` package and execute the following function

```
In[14]:= << HarmonicSums.m
HarmonicSums by Jakob Ablinger -- © RISC
In[15]:= TransformToSSums[sol]
Out[15]= 2^n (S[1, n]^2 + S[2, n] + S[1, n] S[1, {1/2}, n] + S[2, {1/2}, n] - S[1, 1, {1/2, 2}, n] - 3 S[1, 1, {1, 1/2}, n])
```

To sum up, using the summation paradigms given in Fig. 1, we computed for the definite sum (13) the closed form in terms of generalized harmonic sums:

$$A(n) = 2^n [S_1(n)^2 + S_2(n) + S_1(n)S_1(\tfrac{1}{2}; n) + S_2(\tfrac{1}{2}; n) - S_{1,1}(\tfrac{1}{2}, 2; n) - 3S_{1,1}(1, \tfrac{1}{2}; n)]. \quad (17)$$

Remark. We highlight that all the calculation steps can be verified independently of the way how the (complicated) algorithms work. In this way, we obtain rigorous computer proofs.

3. Automatic simplification of multiple sums: the `EvaluateMultiSums` package

Using `Sigma`'s summation tools (see Fig. 1) the derivation of the right hand side of (17) can be done completely automatically with the package

```
In[16]:= << EvaluateMultiSums.m
EvaluateMultiSums by Carsten Schneider -- © RISC
```

Namely, our sum (13) can be simplified to indefinite nested sums by executing

```
In[17]:= EvaluateMultiSum[Sum[S[1, k]^2 Binomial[n, k], {k, 0, n}], {{k, 0, n}}, {n}, {0}, {Infinity}]
Out[17]= 2^n (S[1, n]^2 + S[2, n] + S[1, n] S[1, {1/2}, n] + S[2, {1/2}, n] - S[1, 1, {1/2, 2}, n] - 3 S[1, 1, {1, 1/2}, n])
```

The underlying method applies systematically the tools of `Sigma` (taking care of all the different options to find the optimal treatment) until the simplification is accomplished.

The key point is that this method can be applied iteratively to multiple sums. E.g., in recent calculations we succeeded in deriving the massive 3-loop OMEs $A_{qq,Q}^{(3),NS}$ and $A_{qq,Q}^{(3),NS,TR}$ for general values of N , in particular we obtained the Wilson coefficient $L_{qq,Q}^{(3),NS}$; for further comments we refer to [40]. Here one (of many multi-sums) is the hypergeometric quadruple sum

$$\mathcal{I}(\varepsilon, n) = \sum_{m=0}^n \sum_{j=0}^m \sum_{k=0}^{\infty} \sum_{i=0}^k f(\varepsilon, n, m, j, k, i) \quad (18)$$

where the summand is given in terms of Gamma functions and binomial coefficients:

$$\text{In[18]}:= \mathbf{f} = \frac{(-1)^{i+m+1} e^{-\frac{3\epsilon\gamma}{2}} \binom{k}{i} \binom{m}{j} \binom{n}{m} \Gamma(1-\frac{\epsilon}{2}) \Gamma(\frac{\epsilon}{2}) \Gamma(i-\frac{\epsilon}{2}) \Gamma(-\epsilon+i+j+1) \Gamma(k-\frac{3\epsilon}{2}) \Gamma(\frac{\epsilon}{2}+j+k+1) \Gamma(m+2) \Gamma(-\epsilon-j+m+1)}{k! \Gamma(-\frac{\epsilon}{2}+i+1) \Gamma(-\frac{3\epsilon}{2}+i+j+2) \Gamma(\frac{\epsilon}{2}+m+2) \Gamma(-\frac{\epsilon}{2}+k+m+2)}$$

In this instance, the first 5 coefficients $I_{-3}(n), \dots, I_1(n)$ of its Laurent series expansion (2) with $t = -3$ were needed. For this task (see Section 1) we first expand the summand

$$f(\epsilon, n, m, j, k, i) = F_{-3}(n, m, j, k, i) \epsilon^{-3} + \dots + F_1(n, m, j, k, i) \epsilon^1 + O(\epsilon^2). \quad (19)$$

Then the coefficients $I_i(n)$ are given by applying the sums to the $F_i(n, m, j, k, i)$. E.g., we compute $F_{-1}(n, m, j, k, i)$ by using the following function of the package `EvaluateMultiSums`:

$$\text{In[19]}:= \mathbf{F_{-1}} = \text{SeriesForProduct}[\mathbf{f}, \{\epsilon, -1, -1\}]$$

$$\text{Out[19]}= \frac{(-1)^{i+m+1} 2 \binom{k}{i} \binom{m}{j} \binom{n}{m} (i+j)! (k-1)! (j+k)! (m-j)!}{i(i+j+1)! k!(k+m+1)!}$$

In other words, the main task is to simplify the definite multi-sum

$$\sum_{m=0}^{(d)} \sum_{j=0}^{(c)} \sum_{k=0}^{(b)} \sum_{i=1}^{(a)} \frac{(-1)^{i+m+1} 2 \binom{k}{i} \binom{m}{j} \binom{n}{m} (i+j)! (k-1)! (j+k)! (m-j)!}{i(i+j+1)! k!(k+m+1)!}; \quad (20)$$

note that the inner sum (a) starts with $i = 1$ since there is a pole at $i = 0$. Exactly here (as for all the other multi-sums in this context) the presented toolbox can be exploited. The sum (a) is a definite sum over an indefinite nested product-sum. Therefore we can activate our definite summation technology (Fig. 1): try to transform it to an indefinite nested expression w.r.t. k (which is the summation index of the next sum (b)). Namely, we execute the function⁷:

$$\text{In[20]}:= \mathbf{sumA} = \text{EvaluateMultiSum}[\mathbf{F_{-1}}, \{\{i, 1, k\}\}, \{k, j, m, n\}, \{0, 0, 0, 2\}, \{\infty, m, n, \infty\}]$$

$$\text{Out[20]}= -2 \frac{(-1)^m m!}{(k+m+1)!} \left(\left(\frac{1}{(j+1)^{2k}} - \frac{S[1, k]}{(j+1)^k} \right) \frac{\binom{n}{m} (j+k)!}{j!} - \frac{k! \binom{n}{m}}{(j+1)k(j+k+1)} \right)$$

Given this form, the next sum (b) fits again to our summation paradigm: it is a definite sum over an indefinite nested product sum-expression w.r.t. k . Hence with the function call

$$\text{In[21]}:= \mathbf{sumB} = \text{EvaluateMultiSum}[\mathbf{sumA}, \{\{k, 0, \infty\}\}, \{j, m, n\}, \{0, 0, 2\}, \{m, n, \infty\}]$$

$$\text{Out[21]}= \frac{(-1)^m \binom{n}{m}}{(j+1)(m+1)} \left(\sum_{i_1=1}^j \frac{1}{1+m-i_1} \right)^2 - 2 \frac{(-1)^m \binom{n}{m}}{(j+1)^2(m+1)} \sum_{i_1=1}^j \frac{1}{1+m-i_1} + \frac{2(-1)^m \binom{n}{m}}{(j+1)^3(m+1)}$$

$$+ (-1)^m \left(2 \frac{\binom{n}{m} (-m)_j}{(j+1)^2 j!} - \frac{2 \binom{n}{m}}{(j+1)(m+1)} \right) S[2, m] + \frac{(-1)^m \binom{n}{m}}{(j+1)(m+1)} \sum_{i_1=1}^j \frac{1}{(1+m-i_1)^2} - 2 \frac{(-1)^m \binom{n}{m} (-m)_j}{(j+1)^2 j!} \sum_{i_1=1}^j \frac{i_1!}{(-m)_{i_1} i_1^2}$$

$$+ 2 \frac{(-1)^m \binom{n}{m}}{(j+1)(m+1)} \sum_{i_1=1}^j \frac{\sum_{i_2=1}^{i_1} \frac{1}{1+m-i_2}}{i_1} + (-1)^m \left(\frac{2 \binom{n}{m}}{(j+1)(m+1)} - 2 \frac{\binom{n}{m} (-m)_j}{(j+1)^2 j!} \right) z_2$$

we obtain an indefinite nested product-sum expression w.r.t. j (if sums are free of j , they are indefinite nested w.r.t. m , etc.). Again we are ready to apply our summation toolkit:

$$\text{In[22]}:= \mathbf{sumC} = \text{EvaluateMultiSum}[\mathbf{sumB}, \{\{j, 0, m\}\}, \{m, n\}, \{0, 2\}, \{n, \infty\}]$$

$$\text{Out[22]}= (-1)^m \left(2 \frac{(-1)^m (-n)_m}{(m+1)^4 m!} + 2 \frac{(-1)^m (-n)_m}{(m+1)^3 m!} S[1, m] - 2 - \frac{(-1)^m (-n)_m}{(m+1) m!} S[2, 1, m] \right)$$

This yields an indefinite nested product sum expression w.r.t. m . To this end, the outermost sum (d) is transformed to an indefinite nested product-sum expression

$$\text{In[23]}:= \mathbf{sumD} = \text{EvaluateMultiSum}[\mathbf{sumC}, \{\{m, 0, n\}\}, \{n\}, \{2\}, \{\infty\}]$$

$$\text{Out[23]}= \frac{2S[2, n]}{(n+1)^2} + \frac{2}{(n+1)^4}$$

which is nothing else than the simplification of (20). The full power of this machinery comes into action if the function is applied in one stroke by the following function call:

⁷ The free integer parameters k, j, m, n are given explicitly where their lower values are 0, 0, 0, 2 and their upper values are ∞, m, n, ∞ , respectively. I.e., $0 \leq k \leq \infty$, $0 \leq j \leq m$, $0 \leq m \leq \infty$ and $2 \leq n \leq \infty$. In particular, the given order k, j, m, n specifies how the sums should be transformed: if a sum depends on k , it should be transformed to indefinite sums w.r.t. k ; if it is free of k , but depends on j , it should be transformed w.r.t. j , etc.

In[24]:= EvaluateMultiSum[F₋₁, {{i, 1, k}, {k, 0, ∞}, {j, 0, m}, {m, 0, N}}, {n}, {2}, {∞}]
 Out[24]= $\frac{2S[2, n]}{(n+1)^2} + \frac{2}{(n+1)^4}$

More generally, we can deal with the following problem.

Problem EMS: EvaluateMultiSum. *Given*

$$F(\vec{m}) = \sum_{k_1=l_1}^{L_1(\vec{m})} \dots \sum_{k_v=l_v}^{L_v(\vec{m}, k_1, \dots, k_{v-1})} f(\vec{m}, k_1, \dots, k_v) \quad (21)$$

with an indefinite nested product-sum expression f w.r.t. k_v , integer parameters $\vec{m} = (m_1 \dots, m_r)$; $l_i \in \mathbb{N}$ and $L_i(\dots)$ stands for ∞ or a linear combination of the involved parameters with integer coefficients. *Find* an indefinite nested product-sum expression⁷ $\bar{F}(\vec{m})$ which evaluates to the same expression as $F(\vec{m})$.

⁷ If a sum depends on m_r , it should occur only in the outermost bound. If it is free of m_r , but depends on m_{r-1} , the parameter m_{r-1} should only occur in the outermost bound, etc.

Moreover, if one uses for instance the option `ExpandIn` → { $\varepsilon, -3, 1$ } also the expansion feature is applied. More precisely, the summand (19) is expanded and afterwards the summation machinery is applied. More precisely, with the following function call we arrive at the coefficients $I_{-3}(n), \dots, I_1(n)$ in terms of harmonic sums and the Riemann Zeta values $z_i = \zeta(i) = \sum_{k=1}^{\infty} 1/k^i$:

In[25]:= EvaluateMultiSum[f, {{i, 0, k}, {k, 0, ∞}, {j, 0, m}, {m, 0, N}}, {n}, {2}, {∞}, ExpandIn → { $\varepsilon, -3, 1$ }]
 Out[25]= $\left\{ -\frac{8}{3(n+1)^2}, -\frac{8S[1, n]}{3(n+1)^2}, -\frac{2SS[1, n]^2}{3(n+1)^2} - \frac{2S[2, n]}{3(n+1)^2} - \frac{z_2}{(n+1)^2} - \frac{8}{3(n+1)^4}, \right.$
 $\left. -\frac{S[1, n]^3}{9(n+1)^2} + \left(-\frac{S[2, n]}{(n+1)^2} - \frac{8}{3(n+1)^4} \right) S[1, n] - \frac{z_2 S[1, n]}{(n+1)^2} - \frac{8S[3, n]}{9(n+1)^2} - \frac{2S[2, 1, n]}{3(n+1)^2} - \frac{5z_3}{3(n+1)^2}, \right.$
 $\left. -\frac{S[1, n]^4}{72(n+1)^2} + \left(-\frac{S[2, n]}{4(n+1)^2} - \frac{2}{3(n+1)^4} \right) S[1, n]^2 + \left(\frac{S[2, n]}{(n+1)^3} + \frac{2S[3, n]}{9(n+1)^2} - \frac{S[2, 1, n]}{(n+1)^2} \right) S[1, n] - \frac{3S[2, n]^2}{8(n+1)^2} - \frac{23z_2^2}{16(n+1)^2} - \right.$
 $\left. \frac{2S[2, n]^4}{3(n+1)^4} + \frac{S[3, n]}{(n+1)^3} + \frac{S[3, n]}{12(n+1)^2} - \frac{S[2, 1, n]}{(n+1)^3} - \frac{5S[3, 1, n]}{3(n+1)^2} + \frac{5S[2, 1, 1, n]}{3(n+1)^2} + \left(-\frac{S[1, n]^2}{4(n+1)^2} - \frac{S[2, n]}{4(n+1)^2} - \frac{1}{(n+1)^4} \right) z_2 + \right.$
 $\left. \left(\frac{2}{(n+1)^3} - \frac{5S[1, n]}{3(n+1)^2} \right) z_3 - \frac{8}{3(n+1)^6} \right\}$

We remark that during these calculations exceptional points at the summation borders are carefully treated (like the point $i = 0$ that does not hold for the sum representation (20)).

As demonstrated above, the definite summation spiral in Fig. 1 (finding a recurrence, solving the recurrence, combining the recurrence using initial values⁸) is applied iteratively from inside to outside, and the corresponding sums are transformed stepwise to indefinite nested product-sum expressions. For a description of the full method we refer to [20, 58]. Here we want to stress the following aspect: For an arbitrary input, the method might fail. First, there might not exist a recurrence for a certain subproblem. However, for Feynman integrals as given in Section 1 the arising multi-sums have the appropriate shape to guarantee that the recurrence finder is always successful; this follows by ideas from [14, 27, 30, 31]. Here only time and space resources might be the bottleneck. Second, our recurrence solver might fail to find sufficiently many solutions in terms of indefinite nested product-sum expressions. And exactly here the miracle happens –at least for the classes of Feynman integrals that we considered so far: Problem RS produces usually the full solution space for the occurring recurrences. Consequently, the solutions can be always combined to an alternative representation for the input sum. In summary, the presented method works very well for big classes of Feynman integrals.

⁸ The initial values (e.g., $n = 1, 2, 3$) can be obtained by the same method (with one parameter less); see [73]. As a consequence, we find indefinite nested product-sum expressions which usually simplify to multiple zeta values, infinite versions of S -sums or cyclotomic sums. Then techniques from [35, 38, 39, 74] are heavily needed to rewrite the constants in this special form and to express them in terms of constants such that no further algebraic relations are known. In order to deal with such problems, Ablinger's `HarmonicSums` package is used.

4. Crunching sums and mass production: the package SumProduction

So far we presented symbolic summation technologies and the related packages that enables one to simplify multi-sums to expressions in terms of indefinite nested product-sum expressions. For various situations, in particular for our 2-loop calculations [44] (based on a careful preparation of my cooperation partners) and case studies of massive 3-loop scalar ladder integrals [29], this toolbox was sufficient to perform the necessary calculations.

However, in 3-loop calculations for complete physical problems in QCD the number of the occurring sums grows substantially. Here several thousands, even up to several hundred thousands of sums have to be simplified. Typical examples are, e.g., the calculations of the first two complete Wilson coefficients $L_{qq,Q}^{PS}$ and L_g^S for general values of the Mellin variable n ; see [49]. Further examples are the current calculations of 3-loop graphs with two fermionic lines of equal mass and diagrams with two massive lines of different mass (for charm and bottom quarks); for examples see [40].

For all these problems the additional package `SumProduction` was heavily used to perform these large scale problems. Subsequently, the feature and usage of the package will be illustrated by the 3-loop corrections of $O(n_f T_F^2 C_{A,F})$ to the massive OMEs with local operator insertions on the gluonic lines, $A_{gq,Q}$ and $A_{gg,Q}$ at general values of the Mellin variable n [52]. One of the larger expressions (actually, it is one of the smallest examples in comparison to the other examples mentioned above) was produced with the help of FORM [75, 76]. I.e., it is a 2 GB expression of 2419 multi-sums. Each of them can be treated by `EvaluateMultiSum`, like e.g.,

$$\begin{aligned} \text{In[26]} := & \text{EvaluateMultiSum}\left[\frac{\pi 2^{\epsilon+3} e^{-\frac{3\gamma\epsilon}{2}} (-1)^{j_1} (j_2+1) \Gamma(2-\epsilon) \Gamma(\frac{\epsilon}{2}+2) \Gamma(-\frac{3\epsilon}{2}) \Gamma(-\frac{\epsilon}{2}+j_1+4) \Gamma(-j_1+n-2) \Gamma(\epsilon-j_1-j_2+n-5)}{(\epsilon-10)(\epsilon-8)(\epsilon-2)\epsilon \Gamma(\frac{\epsilon}{2}-\epsilon) \Gamma(\frac{\epsilon+5}{2}) \Gamma(\frac{\epsilon}{2}+n+1) \Gamma(-j_1-j_2+n-4)}, \right. \\ & \left. \{\{j_2, 0, n-j_1-6\}, \{j_1, 0, n-5\}\}, \{n\}, \{5\}, \text{ExpandIn} \rightarrow \{\epsilon, -3, -1\}\right] \\ \text{Out[26]} = & \left\{0, \frac{16(-1)^n (3n^2+12n+11)}{135(n+1)(n+2)^2(n+3)^2} - \frac{16(n^8+6n^7-6n^6-80n^5-81n^4+178n^3+274n^2-4n-96)}{45(n-2)(n-1)^2n^2(n+1)(n+2)^2(n+3)^2}\right. \\ & \frac{16(n^2-n-8)}{45(n-1)n(n+2)(n+3)} S[1, n], - \frac{8(n^2-n-8)}{45(n-1)n(n+2)(n+3)} S[2, n] + \frac{2(-1)^n (187n+127)(3n^2+12n+11)c}{2025(n+1)^2(n+2)^2(n+3)^2} \\ & + \left(\frac{2(17n^6-231n^5+121n^4+2063n^3-1458n^2-2432n+960)}{675(n-2)(n-1)^2n^2(n+1)(n+2)(n+3)} - \frac{16(-1)^n (3n^2+12n+11)}{135(n+1)(n+2)^2(n+3)^2}\right) S[1, n] \\ & \left. + \frac{2(43n^{12}+112n^{11}+263n^{10}-216n^9-11309n^8-16476n^7+55837n^6+78164n^5-95178n^4-116688n^3+51784n^2+30624n-23040)}{675(n-2)^2(n-1)^3n^3(n+1)^2(n+2)^2(n+3)^2}\right\} \end{aligned}$$

For details on the calculation steps for this particular sum we refer to [20]. Similarly, all the other sums could be treated step by step with a lot of computer resources. However, as worked out in [20] we can do it much better by using the toolbox of the package

```
In[27] := << SumProduction.m
SumProduction - AsummationpackagebyCarstenSchneider© RISC - Linz
```

4.1. Reduction to master sums.

First, we reduce the 2 GByte expression (stored in `expr` and being valid for $n \geq 6$) to master sums (resp. key sums) with the function call

```
In[28] := compactExpr = ReduceMultiSums[expr, {n}, {6}, {∞}];
```

The reduced expression `compactExpr` is only 7.6 MByte large and it required 6 hours and 53 minutes to obtain this reduction.

Problem RMS: ReduceMultiSums. *Given* a linear combination of definite multi-sums⁸ over indefinite nested product-sum expressions as given in (21). *Compactify* the expression, i.e., express the summands with objects such that no algebraic relations remain. In particular, synchronize the summation bounds and merge the sums to master sums.

⁸ The sums can be of the form (3) like in our concrete example `In[26]`, or the summands might also involve, e.g., harmonic sums in the numerators; cf. (6). In addition, further regulators (like ϵ) might be involved.

In this routine the merging can be done in different ways. By default, the summands are tried to be given in the form $r t_1^{m_1} \dots t_r^{m_r}$ where the t_i are indefinite nested sums or products, $m_i \in \mathbb{Z}$ and r is a rational function where the numerator and denominator are co-prime.

In short, we reduce the sum expressions to a minimal set of key sums/master sums where no algebraic relations occur within the summand objects. Given this reduced expression, only these master sums have to be treated for further simplification.

In our concrete example the 2419 sums are synchronized w.r.t. the occurring summation ranges (taking for each class the maximum of the lower bounds and the minimum of the upper bounds). As result, we obtained only 4 sums with synchronized ranges

$$\sum_{i_2=5}^{n-5} \sum_{i_1=0}^{i_2} h_1(\varepsilon, n, i_2, i_1), \quad \sum_{i_2=0}^{n-5} \sum_{i_1=0}^{n-i_2-5} h_2(\varepsilon, n, i_2, i_1), \quad \sum_{i_1=5}^{n-5} h_3(\varepsilon, n, i_1), \quad \sum_{i_1=0}^{\infty} h_4(\varepsilon, n, i_1)$$

plus a large term free of summation quantifiers. Next, all the occurring Pochhammer symbols, factorials/ Γ -functions, and binomials are written in a basis of algebraically independent objects plus the extra object $(-1)^n$ (if necessary); for details see Problem EAR in Subsection 2.2. Finally, the expressions are split further to get the form $\sum h(n, (i_2,)i_1, \varepsilon) * r(n, (i_2,)i_1, \varepsilon)$ or $h(n, \varepsilon) * r(n, \varepsilon)$ where h stands for a (proper) hypergeometric term in n (and i_1, i_2). More precisely, it is a product of binomials/factorials/Pochhammers in the numerator and denominator, and $r(n, (i_2, i_1), \varepsilon)$ is a rational function in n, ε (and i_1, i_2); note that r might fill several pages. As final result we obtain an expression with only 29 sums and 15 terms being free of sums.

4.2. Automatic computation of the ε -expansions (in parallel)

Finally, the sums are simplified with `EvaluateMultiSums`. In particular, the coefficients of the required Laurent series expansion can be derived. In order to perform this calculation automatically, the following function call can be applied:

```
In[29]:= ProcessEachSum[compactExpr, {n}, {6}, {∞}, ExpandIn → {ε, -3, 0}]
```

It sequentially applies `EvaluateMultiSum` with the corresponding input parameters to the occurring multi-sums in `compactExpr`. In our concrete example this step took in total 2 hours and 35 minutes.

Often the evaluation of one sum (in particular, for triple and quadruple sums) takes several hours. In order to utilize the benefit of the available computers, we emphasize that this function can be executed simultaneously with different Mathematica kernels, in particular, on different machines within a network. Here the following mechanism is applied. Internally, the function takes the first multi-sum and generates a file with the name SUM1. If the result is computed, the file is updated with the result. Then the routine continues with the second sum provided the file SUM2 is not existent on the hard disk. In this way, `ProcessEachSum` can be executed in parallel for mass productions.

4.3. Combination to the final result

Finally, the result of the sums (or the expansion of the sums) are read from the hard disk and are summed up to the final result. In particular, the expressions are reduced further by applying Problem EAR given in Subsection 2.2 to the occurring sums and products. As shortcut also the available algebraic relations for harmonic sums and their generalized versions [38, 39, 60, 61] are utilized. This last step can be carried out with the following function call

```
In[30]:= result = CombineExpression[compactExpr, {n}, {6}, {∞}];
```

This calculation took about 21 seconds. The final result can be expressed in terms of $\zeta_2, \zeta_3, (-1)^n, S_1(n), S_2(n), S_3(n), S_{2,1}(n), S_{3,1}(n), S_{2,1,1}(n)$ and requires about 100 KByte memory. In summary, the total calculation took around 9 hours and 30 minutes.

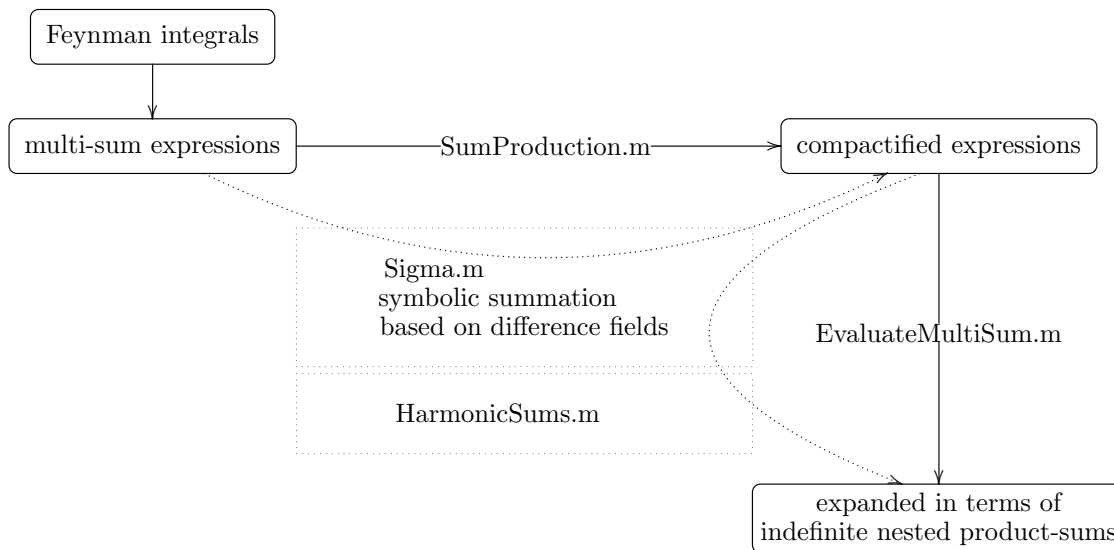


Figure 2. The packages in interaction

5. Conclusion

Summarizing, the whole interaction of the presented packages can be visualized in Fig. 2. Here the Feynman integrals are transformed to huge multi-sum expressions using mostly the computer algebra system FORM. These expressions are then loaded into the computer algebra system Mathematica and our machinery is activated. Using the package `SumProduction` the multi-sum expressions are crunched to expressions in terms of master sums/key sums. In addition, the package supports the user to perform the simplification completely automatically on distributed systems. In order to perform these simplifications, in particular to calculate the coefficients of the Laurent series expansion, the package `EvaluateMultiSums` is called accordingly. This summation technology relies heavily on the summation toolkit of the `Sigma` package that is based on difference field theory. In addition, special function algorithms are needed if infinite summations arise in the given expressions. Here we rely on Ablinger’s `HarmonicSums` package [26, 38, 39, 61] that utilizes and generalizes ideas of [22, 35, 74, 77].

All the packages and underlying algorithms are steadily extended, improved and optimized to deal with more and more complicated Feynman integrals.

Acknowledgments

This work is supported by the Austrian Science Fund (FWF) grants P20347-N18 and SFB F50 (F5009-N15) and by the EU Network LHCPheNet PITN-GA-2010-264564.

References

- [1] Karr M 1981 *J. ACM* **28** 305–350
- [2] Schneider C 2005 *Proc. ISSAC’05* ed Kauers M (ACM) pp 285–292
- [3] Schneider C 2008 *J. Symbolic Comput.* **43** 611–644 [arXiv:0808.2543v1]
- [4] Schneider C 2010 *Motives, Quantum Field Theory, and Pseudodifferential Operators (Clay Mathematics Proceedings vol 12)* ed Carey A, Ellwood D, Paycha S and Rosenberg S (Amer. Math. Soc) pp 285–308 arXiv:0808.2543
- [5] Schneider C 2004 *Proc. ISSAC’04* ed Gutierrez J (ACM Press) pp 282–289
- [6] Schneider C 2007 *J. Algebra Appl.* **6** 415–441
- [7] Schneider C 2010 *Ann. Comb.* **14** 533–552 [arXiv:0808.2596]
- [8] Schneider C 2013 *Computer Algebra in Quantum Field Theory: Integration, Summation and Special Functions* Texts and Monographs in Symbolic Computation ed Schneider C and Blümlein J (Springer) pp 325–360 arXiv:1304.4134 [cs.SC]

- [9] Schneider C 2011 *Symbolic Summation in Difference Fields* Ph.D. thesis RISC, Johannes Kepler University, Linz technical report 01-17
- [10] Schneider C 2004 *Discrete Math. Theor. Comput. Sci.* **6** 365–386
- [11] Schneider C 2007 *Sém. Lothar. Combin.* **56** 1–36 article B56b
- [12] Zeilberger D 1991 *J. Symbolic Comput.* **11** 195–204
- [13] Petkovšek M 1992 *J. Symbolic Comput.* **14** 243–264
- [14] Petkovšek M, Wilf H S and Zeilberger D 1996 *A = B* (Wellesley, MA: A. K. Peters)
- [15] Paule P and Schneider C 2003 *Adv. in Appl. Math.* **31** 359–378
- [16] Andrews G, Paule P and Schneider C 2005 *Advances in Applied Math.* **34** 709–739
- [17] Driver K, Prodinger H, Schneider C and Weideman J A C 2006 *Ramanujan J.* **12** 299–314
- [18] Osburn R and Schneider C 2009 *Math. Comp.* **78** 275–292 arXiv:math/0610281 [math.NT]
- [19] Ablinger J, Blümlein J, Round M and Schneider C 2012 *Loops and Legs in Quantum Field Theory 2012* PoS(2012)50 pp 1–14 arXiv:1210.1685 [cs.SC]
- [20] Blümlein J, Hasselhuhn A and Schneider C 2012 *Proceedings of RADCOR 2011* vol PoS(RADCOR2011)32 pp 1–9 arXiv:1202.4303 [math-ph]
- [21] Ablinger J, Blümlein J, Klein S and Schneider C 2010 *Nucl. Phys. B (Proc. Suppl.)* **205-206** 110–115 arXiv:1006.4797 [math-ph]
- [22] Blümlein J 2009 *Comput. Phys. Commun.* **180** [arXiv:0901.3106 [hep-ph]]
- [23] Blümlein J, Klein S, Schneider C and Stan F 2012 *J. Symbolic Comput.* **47** 1267–1289 arXiv:1011.2656 [cs.SC]
- [24] Bogner C and Weinzierl S 2010 *Int. J. Mod. Phys. A* **25** 2585–2618 [arXiv:1002.3458 [hep-ph]]
- [25] Weinzierl S 2013 *Computer Algebra in Quantum Field Theory: Integration, Summation and Special Functions* Texts and Monographs in Symbolic Computation ed Schneider C and Blümlein J (Springer) pp 381–406
- [26] Ablinger J 2012 *Computer Algebra Algorithms for Special Functions in Particle Physics* Ph.D. thesis J. Kepler University Linz
- [27] Apagodu M and Zeilberger D 2006 *Adv. Appl. Math.* **37** 139–152
- [28] Brown F 2009 *Math. Phys.* **287** 925–958
- [29] Ablinger J, Blümlein J, Hasselhuhn A, Klein S, Schneider C and Wissbrock F 2012 *Nuclear Physics B* **864** 52–84 arXiv:1206.2252v1 [hep-ph]
- [30] Wilf H and Zeilberger D 1992 *Invent. Math.* **108** 575–633
- [31] Wegschaider K 1997 *Computer generated proofs of binomial multi-sum identities* Master’s thesis RISC, J. Kepler University
- [32] Schneider C 2005 *Adv. in Appl. Math.* **34** 740–767
- [33] Schneider C 2005 *J. Differ. Equations Appl.* **11** 799–821
- [34] Chyzak F 2000 *Discrete Math.* **217** 115–134
- [35] Vermaseren J A M 1999 *Int. J. Mod. Phys. A* **14** 2037–2976 arXiv:hep-ph/9806280
- [36] Blümlein J and Kurth S 1999 *Phys. Rev.* **D60** arXiv:hep-ph/9810241
- [37] Moch S O, Uwer P and Weinzierl S 2002 *J. Math. Phys.* **43** 3363–3386
- [38] Ablinger J, Blümlein J and Schneider C 2013 *J. Math. Phys.* **54** 1–74 arXiv:1302.0378 [math-ph]
- [39] Ablinger J, Blümlein J and Schneider C 2011 *J. Math. Phys.* **52** 1–52 [arXiv:1007.0375 [hep-ph]]
- [40] Ablinger J, Blümlein J, Freitas A D, Hasselhuhn A, von Manteuffel A, Raab C, Round M, Schneider C and Wissbrock F 2013 *XXI International Workshop on Deep-Inelastic Scattering and Related Subjects - DIS2013* arXiv:1307.7548 [hep-ph]
- [41] Ablinger J and Blümlein J 2013 *Computer Algebra in Quantum Field Theory: Integration, Summation and Special Functions* Texts and Monographs in Symbolic Computation ed Schneider C and Blümlein J (Springer) pp 1–32
- [42] Weinzierl S 2004 *J. Math. Phys.* **45** 2656–2673 arXiv:hep-ph/0402131
- [43] Bierenbaum I, Blümlein J, Klein S and Schneider C 2007 *Proc. ACAT 2007* vol PoS(ACAT)082 pp 1–15 arXiv:0707.4659 [math-ph]
- [44] Bierenbaum I, Blümlein J, Klein S and Schneider C 2008 *Nucl. Phys. B* **803** 1–41 [arXiv:hep-ph/0803.0273]
- [45] Blümlein J, Kauers M, Klein S and Schneider C 2008 *Proc. of ACAT 2008* vol PoS(ACAT08)106 pp 1–7 arXiv:0902.4095 [hep-ph]
- [46] Blümlein J, Kauers M, Klein S and Schneider C 2009 *Comput. Phys. Commun.* **180** 2143–2165 arXiv:0902.4091 [hep-ph]
- [47] Ablinger J, Bierenbaum I, Blümlein J, Hasselhuhn A, Klein S, Schneider C and Wissbrock F 2010 *Nucl. Phys. B (Proc. Suppl.)* **205-206** 242–249 arXiv:1007.0375 [hep-ph]
- [48] Ablinger J, Blümlein J, Klein S, Schneider C and Wissbrock F 2011 *19th International Workshop On Deep-Inelastic Scattering And Related Subjects (DIS 2011)* (American Institute of Physics (AIP)) arXiv:1106.5937 [hep-ph]
- [49] Ablinger J, Blümlein J, Klein S, Schneider C and Wissbrock F 2011 *Nucl. Phys. B* **844** 26–54 arXiv:1008.3347

- [hep-ph]
- [50] Ablinger J, Blümlein J, Freitas A D, Hasselhuhn A, Klein S, Raab C, Round M, Schneider C and Wissbrock F 2012 *Proc. Loops and Legs in Quantum Field Theory 2012* PoS(LL2012)033 pp 1–12 arXiv:1212.6823 [hep-ph]
- [51] Ablinger J, Blümlein J, Freitas A D, Hasselhuhn A, Klein S, Schneider C and Wissbrock F 2012 *Proceedings of the 36th International Conference on High Energy Physics* vol PoS(ICHEP2012)270 pp 1–9 arXiv:1212.5950 [hep-ph]
- [52] Blümlein J, Hasselhuhn A, Klein S and Schneider C 2013 *Nuclear Physics B* **866** 196–211 arXiv:1205.4184 [hep-ph]
- [53] Gosper R W 1978 *Proc. Nat. Acad. Sci. U.S.A.* **75** 40–42
- [54] Zeilberger D 1990 *J. Comput. Appl. Math.* **32** 321–368
- [55] Kauers M and Paule P 2011 *The concrete tetrahedron* Texts and Monographs in Symbolic Computation (Springer)
- [56] Koutschan C 2013 *Computer Algebra in Quantum Field Theory: Integration, Summation and Special Functions* Texts and Monographs in Symbolic Computation ed Schneider C and Blümlein J (Springer) pp 171–194 arXiv:1307.4554 [cs.SC]
- [57] Kauers M 2007 *J. of Symbolic Comput.* **42** 948–970
- [58] Schneider C 2013 *arXiv:1307.7887 [cs.SC]*
- [59] Karr M 1985 *J. Symbolic Comput.* **1** 303–315
- [60] Blümlein J 2004 *Comput. Phys. Commun.* **159** 19–54 [arXiv:hep-ph/0311046]
- [61] Ablinger J, Blümlein J and Schneider C 2013 *In preparation: Structural Relations of Harmonic Sums*
- [62] Hoffman M 2000 *J. Algebraic Combin.* **11** 49–68
- [63] Schneider C 2010 *Appl. Algebra Engrg. Comm. Comput.* **21** 1–32
- [64] Schneider C 2005 *Ann. Comb.* **9** 75–99
- [65] Abramov S A and Petkovšek M 2010 *J. Symbolic Comput.* **45** 684–708
- [66] Abramov S A, Bronstein M, Petkovšek M and Schneider C 2013 *In preparation*
- [67] Bronstein M 2000 *J. Symbolic Comput.* **29** 841–877
- [68] Abramov S A and Petkovšek M 1994 *Proc. ISSAC'94* ed von zur Gathen J (ACM Press) pp 169–174
- [69] Hendriks P A and Singer M F 1999 *J. Symbolic Comput.* **27** 239–259
- [70] Petkovšek M and Zakrajšek H 2013 *Computer Algebra in Quantum Field Theory: Integration, Summation and Special Functions* Texts and Monographs in Symbolic Computation ed Schneider C and Blümlein J (Springer) pp 259–284
- [71] Moch S O, Vermaseren J A M and Vogt A 2004 *Nucl. Phys. B* **688** 101 arXiv:hep-ph/0403192v1
- [72] Kauers M 2009 *Guessing Handbook* Tech. Rep. 09-07 RISC Report Series, University of Linz, Austria
- [73] Pemantle R and Schneider C 2007 *Amer. Math. Monthly* **114** 344–350
- [74] Blümlein J, Broadhurst D J and Vermaseren J A M 2010 *Comput. Phys. Commun.* **181** 582–625 [arXiv:0907.2557 [math-ph]]
- [75] Vermaseren J A M 2000 *arXiv:math-ph/0010025*
- [76] Vermaseren J A M 2013 *Computer Algebra in Quantum Field Theory: Integration, Summation and Special Functions* Texts and Monographs in Symbolic Computation ed Schneider C and Blümlein J (Springer) pp 361–379
- [77] Remiddi E and Vermaseren J A M 2000 *Int. J. Mod. Phys. A* **15** 725 arXiv:hep-ph/9905237v1