Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Combinatorial Theory, Series A 112 (2005) 1–43

Journal of Combinatorial Theory Series A

www.elsevier.com/locate/jcta

Effective scalar products of D-finite symmetric functions☆

Frédéric Chyzak[a], Marni Mishna[b], Bruno Salvy[a]

[a] *Projet Algorithmes, INRIA Rocquencourt, France*
[b] *LaCIM, Dépt. de Mathématiques Université du Québec à Montréal CP 8888, succ. Centre-ville Montréal, QC, Canada H2X 3Y7*

Received 9 October 2003

Abstract

Many combinatorial generating functions can be expressed as combinations of symmetric functions, or extracted as sub-series and specializations from such combinations. Gessel has outlined a large class of symmetric functions for which the resulting generating functions are D-finite. We extend Gessel's work by providing algorithms that compute differential equations, these generating functions satisfy in the case they are given as a scalar product of symmetric functions in Gessel's class. Examples of applications to k-regular graphs and Young tableaux with repeated entries are given. Asymptotic estimates are a natural application of our method, which we illustrate on the same model of Young tableaux. We also derive a seemingly new formula for the Kronecker product of the sum of Schur functions with itself.

MSC: 05E05; 05E10; 13N10; 13P10

Keywords: Symmetric functions; Differentiably finite functions; Non-commutative Groebner bases; Hammond series; Holonomic D-modules; Kronecker products; Regular graphs; Uniform Young tableaux

0. Introduction

A power series in one variable is called differentiably finite, or simply D-finite, when it is solution of a linear differential equation with polynomial coefficients. This differential

☆ This article completes the extended abstract published in the proceedings of FPSAC'02 under the title "Effective D-Finite Symmetric Functions".

E-mail addresses: frederic.chyzak@inria.fr (F. Chyzak), marni.mishna@inria.fr (M. Mishna), bruno.salvy@inria.fr (B. Salvy).

0097-3165/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcta.2005.01.001

equation turns out to be a convenient data structure for extracting information related to the series and many algorithms operate directly on this differential equation. In particular, the class of univariate D-finite power series is closed under sum, product, Hadamard product, and Borel transform, among other operations, and algorithms computing the corresponding differential equations are known (see for instance [34]). Moreover, the coefficient sequence of a univariate D-finite power series satisfies a linear recurrence, which makes it possible to compute many terms of the sequence efficiently. These closure properties are implemented in computer algebra systems [24,31]. Also, the mere knowledge that a series is D-finite gives information concerning its asymptotic behavior. Thus, whether it be for algorithmic or theoretical reasons, it is often important to know whether a given series is D-finite or not, and it is useful to compute the corresponding differential equation when possible.

D-finiteness extends to power series in several variables: a power series is called D-finite when the vector space spanned by the series and its derivatives is finite-dimensional. Again, this class enjoys many closure properties and algorithms are available for computing the systems of linear differential equations generating the corresponding operator ideals [4,5]. Algorithmically, the key tool is provided by Gröbner bases in rings of linear differential operators and an implementation is available in Chyzak's `Mgfun` package. [1] An additional, very important closure operation on multivariate D-finite power series is definite integration. It can be computed by an algorithm called *creative telescoping*, due to Zeilberger [46]. Again, this method takes as input (linear) differential operators and outputs differential operators (in fewer variables) satisfied by the definite integral. It turns out that the algorithmic realization of creative telescoping has several common features with the algorithms we introduce here.

Beyond the multivariate case, Gessel considered the case of infinitely many variables and laid the foundations of a theory of D-finiteness for symmetric functions [9]. He defines a notion of D-finite symmetric series and obtains several closure properties. The motivation for studying D-finite symmetric series is that new closure properties occur and can be exploited to derive the D-finiteness of usual multivariate or univariate power series. Thus, the main application of [9] is a proof of the D-finiteness for several combinatorial counting functions. This is achieved by describing the counting functions as combinations of coefficients of D-finite symmetric series, which can then be computed by way of a scalar product of symmetric functions. Under certain conditions, the scalar product of symmetric functions depending on extra parameters is D-finite in those parameters, where D-finiteness is that of (usual) multivariate power series. Most of Gessel's proofs are not constructive. In this article, we give algorithms that compute the resulting systems of differential equations for the scalar product operation. Besides Gessel's work, these algorithms are inspired by methods used by Goulden, Jackson, and Reilly in [12,13]. Finally, Gröbner bases are used to help make these methods into algorithms. One outcome is a simplification of the original techniques of [12,13].

Considering some enumerative combinatorial problem of a symmetric flavor and parameterized by a discrete parameter (denoted by k in the examples below), it is often so that the enumeration is solved by first forming a scalar product of two symmetric functions in k variables. Moreover, in the examples envisioned (the enumeration of k-regular graphs, of

[1] This package is part of the `algolib` library available at `http://algo.inria.fr/packages/`.

k-uniform tableaux, etc.), this scalar product is the specialization to k variables of a scalar product between two "closed-form" symmetric functions in infinitely many variables. Both symmetric functions are sufficiently well-behaved that nice "closed forms" are obtained under specialization, leading to descriptions in terms of linear differential operators that are easy to derive. This nice behavior is well exemplified by Eqs. (5) and (8) below and is what delimits the scope of our method in applications.

Additionally, our method extends to other scalar products whose associated adjunctions satisfy a certain condition of preservation of degree (see Section 9.1), as well as to the Kronecker product of symmetric functions (see Section 9.2).

A very basic example of application of our method is the enumeration of labeled graphs. A finite graph on n vertices labeled with non-negative integers $i_1, \ldots, i_n$, of respective valencies $v_1, \ldots, v_n$, is given as a weight the monomial $x_{i_1}^{v_1} \cdots x_{i_n}^{v_n}$. This encoding leads to generating functions that are symmetric series: the set of all finite simple graphs is enumerated by the product

$$G(x) = \sum_{G \in \mathcal{G}} \prod_{(i,j) \in E(G)} x_i x_j = \prod_{i<j} (1 + x_i x_j),$$

as each edge $(i, j) \in E(G)$ is either in the graph or not. This series is obviously invariant under renamings of the x_i's, which motivates the involvement of symmetric function theory in the application. Finite simple graphs whose vertices all have valency two are called *2-regular graphs*. Such a graph contributes to G by a term of the form $x_{i_1}^2 \cdots x_{i_n}^2$. Therefore, extracting the sub-series of G with same monomials as in the series expansion of $\prod_{i \in \mathbb{N}\setminus\{0\}} (1 + x_i^2)$, another symmetric series, results in the generating series of 2-regular graphs according to the same encoding. By symmetry, monomials based on different sets of indices $i_1, \ldots, i_n$ of cardinality n share the same coefficient in this extracted series. In this spirit, it will be shown in Section 3 that the number of 2-regular graphs on n vertices is given as the coefficient of t^n in the series

$$G_2(t) = \left\langle \exp\big((p_1^2 - p_2)/2 - p_2^2/4\big), \exp\big(t(p_1^2 + p_2)/2\big) \right\rangle.$$

Here, the scalar product is a scalar product for symmetric functions, to be defined in the next section; it implements the coefficient extraction. The variables t, p_1, and p_2 can be viewed as standard variables, although p_1 and p_2 will be assigned the symmetric function interpretation $p_1 = x_1 + x_2 + \cdots$, $p_2 = x_1^2 + x_2^2 + \cdots$. Our purpose in the present paper is to describe scalar products of symmetric functions like $G_2(t)$ by a linear differential equation. By our method, Algorithm 1 below calculates that $G_2(t)$ satisfies the differential equation

$$2(1-t)G_2'(t) - t^2 G_2(t) = 0,$$

which is easily solved to recover the classical series $G_2(t) = e^{-\frac{1}{4}t(t+2)}/\sqrt{1-t}$. More details on this calculation as well as similar examples will be given in Section 3. In general, the derived differential equation will not admit of such a closed form solution. However, it is possible to extract asymptotic information on the sequence being enumerated directly from this differential equation. This will be exemplified in Section 8.

This article is organized as follows. After recalling the necessary part of Gessel's work in Section 1, we start by focusing on the special situation when a single argument of the scalar product depends on extra parameters. We present an algorithm for computing the differential equations satisfied by the scalar product in this case in Section 2. The application to the example of k-regular graphs is detailed in Section 3. Then a special case where the algorithm can be further refined is described in Section 4. We treat a variant of Young tableaux where each element is repeated k times in Section 5. (These are in bijection with a generalization of involutions [19].) The general form of the main algorithm, when both arguments depend on extra parameters, is given in Section 6. Termination and correctness of the main algorithms are proved in Section 7. Next, in Section 8 we employ our algorithms to derive asymptotic estimates of the enumerating sequences of k-regular graphs for $k = 1, 2, 3, 4$. Following this approach of experimental mathematics, we state a conjecture for general k. A discussion on several extensions and applications of the method closes the paper in Section 9, including the calculation of a seemingly new formula for the Kronecker product of the sum of all Schur functions with itself.

1. Symmetric D-finite functions

In this section, we recall the facts we need about symmetric functions, D-finite functions, and symmetric D-finite functions.

1.1. Symmetric functions

We first collect basic definitions, notation, and results of the theory of symmetric functions. We refer to [21,34] for further results.

Symmetric functions are series in the infinite set of variables $x_1, x_2, \dots$ over a field K of characteristic 0, subject to a certain invariance under renumberings of the variables. The K-algebra Λ of symmetric functions is formally defined as follows. For each positive integer m, the K-vector space consisting of the polynomials of $K[x_1, \dots, x_m]$ that are fixed under any permutation of the variables is a graded K-algebra G_m, the algebra of symmetric polynomials in m variables. Here the grading is with respect to the total degree in the m variables and it induces a chain of graded surjective homomorphisms π_m from G_{m+1} onto G_m defined by setting x_{m+1} to 0. Taking the inverse limit (a.k.a. projective limit) of the system $(\{G_m\}, \{\pi_m\})$ results in the graded K-algebra Λ of symmetric functions. By restriction of the algebras G_m and the maps π_m to homogeneous polynomials in a fixed degree n, the inductive limit becomes a vector subspace Λ_n of Λ. We have the relation $\Lambda = \bigoplus_{n \geqslant 0} \Lambda_n$.

We now recall the definitions of the most frequently used bases of the ring Λ and vector spaces Λ_n. Denote by $\lambda = (\lambda_1, \dots, \lambda_k)$ a partition of the integer n. This means that $n = \lambda_1 + \cdots + \lambda_k$ and $\lambda_1 \geqslant \cdots \geqslant \lambda_k > 0$, which we also denote $\lambda \vdash n$. Alternatively, the power notation $\lambda = 1^{r_1} \cdots k^{r_k}$ for partitions indicates that i occurs r_i times in λ, for $i = 1, 2, \dots, k$. Partitions serve as indices for the five principal symmetric function families that we use:

- the homogeneous symmetric functions $h_\lambda = h_{\lambda_1} \cdots h_{\lambda_k}$, for h_n defined as the sum of all monomials of degree n in $x_1, x_2, \dots$, with possible repetition (i.e., with any non-negative exponents),

- the elementary symmetric functions $e_\lambda = e_{\lambda_1} \cdots e_{\lambda_k}$, for e_n defined as the sum of all monomials of degree n in $x_1, x_2, \dots$, with no possible repetition (i.e., with exponents 0 or 1, exclusively),
- the power symmetric functions $p_\lambda = p_{\lambda_1} \cdots p_{\lambda_k}$, for p_n defined as the sum of the nth power of all variables,
- the monomial symmetric functions $m_\lambda = \sum_\sigma (r_1!\, r_2! \dots)^{-1} x_{\sigma(1)}^{\lambda_1} \cdots x_{\sigma(k)}^{\lambda_k}$, where σ ranges over all permutations of the non-negative integers,
- the Schur symmetric functions s_λ, whose intuitive definition is in terms of the representations of the permutation group S_n, and that can alternatively be defined as the limit symmetric function when n tends to infinity of the determinant of the $n \times n$-matrix with (i, j)-entry $h_{\lambda_i - i + j}$.

When the indices are restricted to all partitions of the same positive integer n, any of the five families forms a basis for the vector space of symmetric polynomials of degree n in $x_1, x_2, \dots$. On the other hand, any of the three families indexed by the integers $i \in \mathbb{N}$, (p_i), (h_i), and (e_i), is algebraically independent over $\mathbb{Q}$ and generates the algebra Λ of symmetric functions over $K : \Lambda = K[p_1, p_2, \dots] = K[h_1, h_2, \dots] = K[e_1, e_2, \dots]$. In this work, we shall focus on the basis (p_i), as we shall endow Λ with a differential structure will regard to the variables p_i.

Generating series of symmetric functions live in the larger ring of symmetric series, $K[t][[p_1, p_2, \dots]]$. There, we have the generating series of homogeneous and elementary functions:

$$H(t) = \sum_n h_n t^n = \exp\left(\sum_i p_i \frac{t^i}{i}\right), \quad E(t) = \sum_n e_n t^n = \exp\left(\sum_i (-1)^i p_i \frac{t^i}{i}\right).$$

1.2. Scalar product and coefficient extraction

The ring of symmetric series is endowed with a scalar product defined as a bilinear symmetric form such that the bases (h_λ) and (m_λ) are dual to each other:

$$\langle m_\lambda, h_\mu \rangle = \delta_{\lambda,\mu}, \tag{1}$$

where $\delta_{\lambda,\mu}$ is 1 if $\lambda = \mu$ and 0 otherwise.

For a partition in power notation, $\lambda = 1^{n_1} \cdots k^{n_k}$, the normalization constant

$$z_\lambda := 1^{n_1} n_1! \cdots k^{n_k} n_k!$$

plays the role of the square of a norm of p_λ in the following important formula:

$$\langle p_\lambda, p_\mu \rangle = \delta_{\lambda,\mu} z_\lambda. \tag{2}$$

The scalar product is a basic tool for coefficient extraction. Indeed, if we write $F(x_1, x_2, \dots)$ in the form $\sum_\lambda f_\lambda m_\lambda$, then the coefficient of $x_1^{\lambda_1} \cdots x_k^{\lambda_k}$ in F is $f_\lambda = \langle F, h_\lambda \rangle$, by (1). Moreover, when $\lambda = 1^n$, the identity $h_{1^n} = p_{1^n}$ yields a simple way to compute this coefficient when F is written in the basis of the p's:

Theorem 1 (*Gessel; Goulden and Jackson*). *Let θ be the K-algebra homomorphism from the algebra of symmetric functions over K to the algebra $K[[t]]$ of formal power series in t defined by $\theta(p_1) = t$, $\theta(p_n) = 0$ for $n > 1$. Then if F is a symmetric function,*

$$\theta(F) = \sum_{n=0}^{\infty} a_n \frac{t^n}{n!},$$

where a_n is the coefficient of $x_1 \cdots x_n$ in F.

Gessel also provides an analogue for this theorem when $\lambda = 1^n 2^m$ and $\lambda = 1^n 3^m$ [9, Theorems 2–4]. Combinations of other degree patterns quickly become arduous to write explicitly.

1.3. Plethysm

Plethysm is a way to compose symmetric functions, which in the simplest case, amounts to simply scaling the indices on the power sums. This inner law of Λ, denoted $u[v]$ for u, v in Λ, is, for $w = \sum_\lambda c_\lambda p_\lambda$, defined by the rules [34]

$$p_n[w] = \sum_\lambda c_\lambda p_{n\times\lambda_1} p_{n\times\lambda_2} \cdots,$$

$$(\alpha u + \beta v)[w] = \alpha u[w] + \beta v[w], \qquad (uv)[w] = u[w]v[w],$$

where α, β in K. For example, consider that $w[p_n] = p_n[w]$, and in particular that $p_n[p_m] = p_{n\times m}$. Thus, we see that when we write $w \in \Lambda$ in the power sum basis as $w = w(p_1, p_2, \ldots, p_k, \ldots)$, the scaling effect appears on the indices as

$$w[p_n] = w(p_{1\times n}, p_{2\times n}, \ldots, p_{k\times n}, \ldots).$$

1.4. D-finiteness of multivariate series

Recall that a series $F \in K[[x_1, \ldots, x_n]]$ is *D-finite* in $x_1, \ldots, x_n$ when the set of all partial derivatives and their iterates, $\partial^{i_1+\cdots+i_n} F/\partial x_1^{i_1} \cdots \partial x_n^{i_n}$, spans a finite-dimensional vector space over the field $K(x_1, \ldots, x_n)$. A *D-finite description* of a series F is a set of differential equations whose solutions in any $K(x_1, \ldots, x_n)$-vector space share this property. A typical example of such a set is a system of n differential equations of the form

$$q_1(x) f(x) + q_2(x) \frac{\partial f}{\partial x_i}(x) + \cdots + q_k(x) \frac{\partial^k f}{\partial x_i^k}(x) = 0,$$

where i ranges over $1, \ldots, n$, each q_j is in $K(x_1, \ldots, x_n)$ for $1 \leqslant j \leqslant k$, and k and q_j depend on i. Observe that by a theorem of Stafford [2, Chapter 5], any D-finite series F admits a D-finite description consisting of only two differential equations. However, we do not know how to benefit from this theoretical result in our computational setting, and it will be more efficient to compute in a systematic way with non-minimal sets.

The properties of D-finite series we need here are summarized in the following theorem.

Theorem 2. (1) *The set of D-finite power series forms a K-subalgebra of $K[[x_1, \dots, x_n]]$ for the usual product of series.*

(2) *If F is D-finite in $x_1, \dots, x_n$ then for any subset of variables $x_{i_1}, \dots, x_{i_k}$ the specialization of F at $x_{i_1} = \dots = x_{i_k} = 0$ is D-finite in the remaining variables.*

(3) *If P is a polynomial in $x_1, \dots, x_n$, then $\exp P(x)$ is D-finite in $x_1, \dots, x_n$.*

(4) *If F and G are D-finite in the variables $x_1, \dots, x_{m+n}$, then the Hadamard product $F \odot G$ with respect to the variables $x_1, \dots, x_n$ is D-finite in $x_1, \dots, x_{m+n}$.*

(Recall that the Hadamard product of two series $\sum_{\alpha\in\mathbb{N}^k} a_\alpha u^\alpha \odot \sum_{\beta\in\mathbb{N}^k} b_\beta u^\beta$ is $\sum_{\alpha\in\mathbb{N}^k} a_\alpha b_\alpha u^\alpha$, where $u^\alpha = u_1^{\alpha_1} \cdots u_k^{\alpha_k}$.)

These properties are classical [34]. The first three are elementary, the last one relies on more delicate properties of dimension and is due to Lipshitz [20].

We note at this point that it is usually simple in applications to provide a D-finite description for a D-finite function, as the latter is most often given as a polynomial expression in "atomic" D-finite functions, usually well-known special functions. Given a table of atomic D-finite descriptions, one bases on the closure properties of Theorem 2 above and uses algorithms described in [5] in order to derive a D-finite description for the whole expression. In our examples, doing this will be straightforward since our functions will be exponentials of polynomials.

1.5. D-finite symmetric functions

The definition of D-finiteness for series in an infinite number of variables is achieved by generalizing property (2) in Theorem 2: $F \in K[[x_1, x_2, \dots]]$ is called *D-finite* in the infinitely many variables x_i if, for any choice of a finite set S of positive integers, the specialization to 0 of each x_i for i not in S results in a power series that is D-finite, in the classical sense, in the variables x_i for i in S. In this case, all the properties in Theorem 2 hold in the infinite multivariate case.

The definition is then tailored to symmetric series by considering the algebra of symmetric series as generated over K by the set $\{p_1, p_2, \dots\}$: a symmetric series is called *D-finite* when it is D-finite in the p_i's.

Property (4) in Theorem 2 has the following very important consequence:

Theorem 3 (*Gessel*). *Let f and g be elements of $K[[t_1, \dots, t_k]][[p_1, p_2, \dots]]$, D-finite in the p_i's and t_j's, and suppose that g involves only finitely many of the p_i's. Then $\langle f, g\rangle$ is D-finite in the t_j's provided it is well-defined as a power series.*

We return to the example of regular graphs given in the introduction. We shall see in Section 3 that the exponential generating series G_2 of 2-regular graphs is given as an extraction of coefficients from the generating series G of all finite simple graphs in the form $G_2 = \langle G, \exp(h_2 t)\rangle$ and we shall provide the explicit representations

$$G = \exp\left(\sum_i (-1)^i \frac{p_i^2 - p_{2i}}{2i}\right) \qquad \text{and} \qquad h_2 = \frac{p_1^2 + p_2}{2}.$$

Both G and $\exp(h_2 t)$ are clearly D-finite symmetric by the definition above. Now, G_2 is equal to the scalar product

$$\left\langle \exp\left(\sum_i (-1)^i (p_i^2 - p_{2i})/2i\right), \exp\bigl(t(p_1^2 + p_2)/2\bigr)\right\rangle,$$

and thus by Theorem 3 the resulting power series is D-finite in t. Note the effect of the requirement that g be dependent on finitely many p_i's only in the theorem—here $\exp h_2 t$ depends on p_1 and p_2 only. As a consequence, the scalar product extracts those terms from G that are supported by monomials in t, p_1, and p_2 only. In other words, we can set all p_i's to 0 in G when $i > 2$, which yields

$$G_2(t) = \Bigl\langle \exp\bigl((p_1^2 - p_2)/2 - p_2^2/4\bigr), \exp\bigl(t(p_1^2 + p_2)/2\bigr)\Bigr\rangle.$$

This scalar product is between symmetric functions in finitely many p_i's.

1.6. Effective D-finite symmetric closures

Our work consists in making Theorem 3 effective by giving algorithms for producing linear differential equations annihilating $\langle f, g\rangle$. The input to our algorithms consists of closed forms for g and the specialization of f in the finite number of p_i's appearing in g, from which generators of ideals of differential operators which annihilate them can then be computed.

Providing algorithms to manipulate linear differential equations amounts to making the closure properties of univariate D-finite series effective; similarly, algorithms operating on systems of linear differential operators make the closure properties of multivariate D-finite series effective. Our title is thus motivated by the fact that our algorithm makes it possible to compute all the information on a scalar product that can be predicted from D-finiteness. Note that we do not check that the resulting power series is well-defined: our algorithm merely computes equations that the scalar product series must satisfy if it is well-defined.

In our examples, we make use of symmetric series that are built by plethysm. Closure properties are given by Gessel, but our applications require only a simple consequence of property (3) in Theorem 2, namely that if g is a polynomial in the p_i's, then $h[g]$ and $e[g]$ are D-finite for $h = H(1)$ and $e = E(1)$.

2. Algorithm for scalar product: the simple case

We proceed to give a new algorithm to compute the differential equation satisfied by a scalar product of two D-finite symmetric series under the hypotheses of Theorem 3 and with the additional simplifying condition that only one of the symmetric series depends on t. When the number of t variables is 1, the output is a single differential equation for which existing computer algebra algorithms might find a closed-form solution. In most cases however, no such solution exists and we are content with a differential equation from which useful information can be extracted.

The basic tool we use here is non-commutative Gröbner bases in extensions of Weyl algebras. An introduction to this topic can be found in [30]. By W_t, we denote the Weyl algebra

$$\begin{aligned} W_t = K\big\langle & t_1, \dots, t_k, \partial_{t_1}, \dots, \partial_{t_k}; \\ & [\partial_{t_i}, t_j] = \delta_{i,j}, [t_i, t_j] = [\partial_{t_i}, \partial_{t_j}] = 0,\ 1 \leqslant i, j \leqslant k\big\rangle, \end{aligned}$$

where the bracket $[a, b]$ denotes $ab - ba$ and $\delta_{i,j}$ is the Kronecker notation. This algebra can be identified with the algebra of linear differential operators with coefficients that are polynomial in $t = t_1, \dots, t_k$. We correspondingly denote W_p for variables $p = p_1, \dots, p_n$, as well as ∂_t for $\partial_{t_1}, \dots, \partial_{t_k}$, ∂_p for $\partial_{p_1}, \dots, \partial_{p_n}$, etc. For the algorithm, we work in the extension

$$W_{p,t}(t) = K(t) \otimes_{K[t]} W_{p,t}$$

of the Weyl algebra $W_{p,t} = W_p \otimes_K W_t$ in which the coefficients of the differential operators are still polynomial in p but rational in t. Suppose F and G belong to $K[t][[p]]$ and are D-finite symmetric series as in Theorem 3. In particular, they both satisfy systems of linear differential equations with polynomial coefficients from $K(t)[p]$. We can write these equations as elements of $W_{p,t}(t)$ acting on F and G. The set $\mathcal{I}_F = \operatorname{ann}_{W_{p,t}(t)} F$ (resp. $\mathcal{I}_G$) of all operators of $W_{p,t}(t)$ annihilating F (resp. G) is then a *left* ideal of $W_{p,t}(t)$. Given as input Gröbner bases of $\mathcal{I}_F$ and $\mathcal{I}_G$, our algorithm outputs non-zero elements of the annihilating left ideal $\operatorname{ann}_{W_t(t)} \langle F, G\rangle$.

To combine elements of $\mathcal{I}_F$ and $\mathcal{I}_G$ in a meaningful way we use the adjunction map, denoted $\diamond$ here,[2] defined for an operator $P \in W_p$ by imposing the relation $\langle P \cdot F, G\rangle = \big\langle F, P^\diamond \cdot G\big\rangle$ for all series F and G. As a consequence, we have the relation $(PQ)^\diamond = Q^\diamond P^\diamond$ and the adjoint $P^\diamond$ is computed formally from $p_i^\diamond = i\partial_{p_i}$ and $\partial_{p_i}^\diamond = p_i/i$; in particular $(p_i\partial_{p_i})^\diamond = p_i\partial_{p_i}$ [21]. This makes the adjunction map an involution as well as an algebra anti-automorphism of W_p. Note that, although adjunction extends to $W_p(t)$ by setting $t_i^\diamond = t_i$, no adjoint for the ∂_{t_i} can be defined in any consistent way. Assume that an adjoint $\partial_{t_i}^\diamond$ existed. For reasons to be explained later, this adjoint has to be of the form $\alpha\partial_{t_i} + \beta t_i + \gamma$ for complex constants α, β, γ, with $\alpha\beta \neq 0$. Now, for any series F and G we have $\big\langle \partial_{t_i} \cdot F, G\big\rangle = \big\langle F, \partial_{t_i}^\diamond \cdot G\big\rangle$. Choose any non-zero series F independent of t_i; then by the method of variation of parameters for series, one finds a series G satisfying $\partial_{t_i}^\diamond \cdot G = F$. Upon evaluation, we obtain $0 = \langle F, F\rangle \neq 0$, a contradiction.

We now proceed to outline the algorithm for the simple case, meaning that from this point on we elect to have $F \in K[[p]]$, i.e., F independent of t. The condition on F that it does not involve t implies that $\partial_{t_i} \cdot F = 0$ for i from 1 to k. We can use this fact to simplify our calculations. In this case, we consider a different annihilator, $\operatorname{ann}_{W_p} F$, hereafter denoted J_F. Note that $J_F = \mathcal{I}_F \cap W_p$.

This allows us to determine the action of combinations of $P \in J_F^\diamond$ and $Q \in \mathcal{I}_G$. For example, given any $S \in W_p$, $T \in W_{p,t}(t)$, and $U \in W_t(t)$,

$$\big\langle F, (P^\diamond SU + TQ) \cdot G\big\rangle = \big\langle S^\diamond P \cdot F, U \cdot G\big\rangle + \langle F, TQ \cdot G\rangle = 0.$$

[2] Macdonald denotes the adjunction operator by $\perp$.

It follows that, if we can find a combination such that $\sum_j P_j^\diamond S_j U_j + \sum_j T_j Q_j = R \in W_t$, we have $0 = \langle F, R \cdot G\rangle = R \cdot \langle F, G\rangle$. Note that each $P_j^\diamond S_j$ is an element of $J_F^\diamond$ while each $T_j Q_j$ is an element of $\mathcal{I}_G$. Therefore, we conduct our search for an element of $\operatorname{ann}_{W_t} \langle F, G\rangle$ by determining a non-zero element of $\left(J_F^\diamond W_t(t) + \mathcal{I}_G\right) \cap W_t$. We shall prove in Section 7.1 that such an element exists. Basically, the goal of our algorithms is to compute sufficiently many non-zero elements of $\left(J_F^\diamond W_t(t) + \mathcal{I}_G\right) \cap W_t$ so as to generate a D-finite description of the scalar product.

Note, however, that while $\mathcal{I}_G$ is a left $W_{p,t}(t)$ ideal, $J_F^\diamond W_t(t)$ is a *right* $W_{p,t}(t)$-ideal and the sums $P + Q$ for $P \in J_F^\diamond W_t(t)$ and $Q \in \mathcal{I}_G$ do not form an ideal. This problem is very similar to the problem of creative telescoping: given an ideal $\mathcal{I} \subset W_{p,t}(t)$, the aim in the first step of this method is to determine an element of $\partial_p W_{p,t}(t) + \mathcal{I}$ that does not involve p. There also, $\partial_p W_{p,t}(t) := \sum_j \partial_{p_j} W_{p,t}(t)$ is a right ideal. The algorithm we present thus bears a non-fortuitous resemblance with that of [37]: in this reference, truncations of the left ideal $\mathcal{I}$ and of the right ideal $\partial_p W_{p,t}(t)$ at a given total degree in $p, \partial_p, \partial_t$ are recombined linearly, this for higher and higher truncation degrees until the corresponding truncation of the intersection $\left(\partial_p W_{p,t}(t) + \mathcal{I}\right) \cap W_t$ is non-trivial. In our situation, we determine truncations of the left ideal $\mathcal{I}_G$ and the right ideal $J_F^\diamond W_t(t)$ at a given truncation order, recombine those two vector spaces linearly, and iterate over higher and higher truncation orders until the corresponding truncation of $\left(J_F^\diamond W_t(t) + \mathcal{I}_G\right) \cap W_t$ is a D-finite description.

To some extent, the approach of the present paper also shares features with that in [27]. However, this reference focuses on determining a bound on a truncation order that permits to compute generators of an intersection $L = \left(\partial_p W_{p,t} + I\right) \cap W_t$ for a given ideal I of $W_{p,t}$, and also generators for a whole free resolution of L. From there, the cohomology groups of the module-theoretic integral W_t/L of the quotient module $W_{p,t}/I$ are derived. Roughly speaking, we are not concerned here with more than the first cohomology group, and furthermore, we treat the similar but different problem for ideals of $W_{p,t}(t)$ and intersections in $W_t(t)$.

Being a module over $W_t(t)$, the sum $J_F^\diamond W_t(t) + \mathcal{I}_G$ is a vector space over $K(t)$. It is this second structure that is adapted to our method. We could try using the module structure in this section, but this would not generalize to the case when also F depends on t. The idea is to use $K(t)$-linear algebra in the vector space structure to eliminate the ∂_{p_i} and p_i. Roughly speaking, we incrementally generate lines in a matrix corresponding to generators of $J_F^\diamond W_t(t) + \mathcal{I}_G$, and perform Gaussian elimination to remove the monomials involving p and ∂_p.

The main loop of the algorithm considers monomials of increasing degree with respect to any ordering on the monomials in $p, \partial_p, \partial_t$. We use the notation $\preccurlyeq$ to denote the monomial comparison associated with this ordering. We reduce each monomial α with respect to (the Gröbner bases for) $\mathcal{I}_F^\diamond$ and $\mathcal{I}_G$. Note that the chosen monomial ordering is the same for both $\mathcal{I}_G$ and $\mathcal{I}_F^\diamond$. Equivalently, the remainder of the reduction of a monomial α with respect to $\mathcal{I}_F^\diamond$ can be viewed as the adjoint of the remainder of the reduction of $\alpha^\diamond$ with respect to $\mathcal{I}_F$. However, to reflect the fact that adjunction modifies the variables, when reducing with respect to $\mathcal{I}_F$ we need to use a different order, specifically, the ordering $\preccurlyeq_\diamond$ defined by $\beta_1 \preccurlyeq_\diamond \beta_2$ on W_p if and only if $\beta_1^\diamond \preccurlyeq \beta_2^\diamond$. In our implementation, we use the ordering

DegRevLex($\partial_p > p > \partial_t$) which sorts by total degree first, breaking ties by a reverse lexicographic order on the variables. The order $\preccurlyeq_\diamond$ is then DegRevLex($p > \partial_p$).

Once we have computed these values, we add two rows (and for sufficiently large α only one column) in a matrix where we perform Gaussian elimination to cancel entries corresponding to monomials involving p and ∂_p.

We now state the algorithm more formally as Algorithm 1, followed by an example in the next section. After this example, we describe the modifications necessary to handle specific cases more efficiently, and how to treat the general case. The proofs that these algorithms work and terminate are delayed until Section 7.

Algorithm 1 (*Scalar product*).

Input: *Symmetric functions $F \in K[[p]]$ and $G \in K[t][[p]]$, both D-finite in p, t, given by D-finite descriptions in W_p and $W_{p,t}(t)$, respectively.*

Output: *A D-finite description of $\langle F, G\rangle$ in $W_t(t)$.*

```
(1) Determine a Gröbner basis $\mathcal{G}_G$ for the left ideal $\mathrm{ann}_{W_{p,t}(t)} G$ with respect to any monomial
    ordering $\preccurlyeq$, as well as a Gröbner basis $\mathcal{G}_{F^\diamond}$ for the right ideal $\mathrm{ann}_{W_p} F^\diamond$ with respect to
    the monomial ordering induced by $\preccurlyeq$ on $W_p$.
(2) $B := \{\}$.
(3) Iterate through each monomial $\alpha$ in $p, \partial_p, \partial_t$.
    (a) Write $\alpha = \beta\gamma$ with $\beta \in W_p$ and $\gamma \in K[\partial_t]$.
    (b) $\alpha_F := \big(\beta - (\beta \,\mathrm{red}_{\preccurlyeq}\, \mathcal{G}_{F^\diamond})\big)\gamma$.
    (c) $\alpha_G := \alpha - (\alpha \,\mathrm{red}_{\preccurlyeq}\, \mathcal{G}_G)$.
    (d) Introduce $\alpha_F$ and $\alpha_G$ as two new elements into B and reduce so as to eliminate $p, \partial_p$.
    (e) Compute the dimension of the ideal generated by $B \cap W_t(t)$. If this dimension is 0,
        break and output $B \cap W_t(t)$.
```

Notice, if $m = 1$, as is the case in our examples, there is only one variable t, and the dimension condition in (3e) is simplified to:

$$\text{If } B \text{ contains a non-zero element } P \text{ from } W_t(t), \text{ break and return } P.$$

Note that Step (1) requires to determine both ideals $\mathrm{ann}_{W_{p,t}(t)} G$ and $\mathrm{ann}_{W_p} F$, not just $\mathrm{ann}_{W_{p,t}(p,t)} G$ and $\mathrm{ann}_{W_p(p)} F$. In other words, one generally needs to pass from a D-finite description $\{P_i\}$ generating the ideal $\mathrm{ann}_{W_p(p)} F$ as $\sum_i W_p(p) P_i$ to a set $\{Q_i\}$ generating the ideal $\mathrm{ann}_{W_p} F = W_p \cap \mathrm{ann}_{W_p(p)} F$ as $\sum_i W_p Q_i$, and similarly for G. The operation of computing such intersections is called *Weyl closure,* in the terminology of [40,41]. It is a non-obvious task, owing to the change of module structure (coefficients in $W_p(p)$ are replaced with coefficients in W_p). Algorithms are provided in [40,41].

Sometimes, the input set $\{P_i\}$ already constitutes a generating set for the Weyl closure. In this case, one can skip Step (1) of the algorithm. This is the case in our examples.

The remainder of the reduction with respect to the Gröbner basis $\mathcal{G}_G$ is a multivariate analogue of the remainder of the Euclidean division. It is such that for any α, $\alpha_{\mathcal{G}} = \alpha - (\alpha \,\mathrm{red}\, \mathcal{G})$ belongs to the ideal generated by $\mathcal{G}$. A similar statement holds for $\mathcal{G}_F$.

For this description we have assumed that Gröbner bases could be computed for both left and right ideals. If they can only be computed on one side, say for left ideals, then the operators α_F can be obtained as follows: first, determine the monomial ordering $\preccurlyeq_\diamond$ induced by adjunction on W_p viewed as a left structure from the ordering $\preccurlyeq$ on W_p viewed as a right structure; then, replace the Gröbner basis $\mathcal{G}_{F^\diamond}$ with the Gröbner basis $\mathcal{G}_F$ for the left ideal $\operatorname{ann}_{W_p} F$ with respect to $\preccurlyeq_\diamond$; α_F is then computed as $\big(\beta - (\beta^\diamond \operatorname{red}_{\preccurlyeq_\diamond} \mathcal{G}_F^\diamond)\big)\gamma$. This way we get $\mathcal{G}_{F^\diamond} = (\mathcal{G}_F)^\diamond$.

We represent the basis B as a matrix, with columns indexed by monomials in the p_i's, the ∂_{p_i}'s, and the ∂_{t_i}'s. Each row in the matrix corresponds to the row vector of the coefficients of some element of B with regard to the indexing monomial basis. Introducing an element into the basis consists of adding a new row at the bottom of the matrix, performing row reduction (also known as Gaussian elimination), and then returning the new matrix as the updated basis. In practice, B can be handled (not inefficiently) by a Gröbner basis computation with respect to a monomial ordering that eliminates the p_i's and the ∂_{p_i}'s, performing calculations in the free $K[t]$-module with a basis the list of indexing monomials.

Finally, some remembering can be done at Step (3b) to avoid reducing the same β again and again, for different α's involving the same β.

3. Example: *k*-regular graphs

The enumeration of regular graphs has been treated by a number of authors [6,9,13,29]. We present it here because of its expository value and as it is the simplest in a family of examples. After expressing the problem as a scalar product, we describe in detail how our algorithm treats it. We conclude this section with an indication of how the scenario may be generalized.

3.1. A generating series for graphs as a scalar product

Recall from the introduction that a generating series for the set of all finite simple graphs labeled with integers from $\mathbb{N} \setminus \{0\}$ is

$$G(x) = \sum_{G \in \mathcal{G}} \prod_{(i,j) \in E(G)} x_i x_j = \prod_{i<j} (1 + x_i x_j),$$

under the encoding that a graph on n vertices $i_1, \dots, i_n$ of respective valencies $v_1, \dots, v_n$ contributes a monomial $x_{i_1}^{v_1} \cdots x_{i_n}^{v_n}$. We can similarly make a generating function for graphs with multiple edges (multigraphs) by

$$M(x) = \prod_{i<j} \frac{1}{(1 - x_i x_j)},$$

for an edge (i, j) of a graph with multiplicity m contributes a monomial $x_i^m x_j^m$ and any non-negative multiplicity is allowed.

Clearly both G and M are symmetric functions, and in fact, we have the relations $G = e[e_2]$ and $M = h[e_2]$, as determined by a method that we discuss in Section 3.4. Both are easily rewritten in terms of the p_i's:

$$G = \exp\left(\sum_i (-1)^i (p_i^2 - p_{2i})/2i\right) \quad \text{and} \quad M = \exp\left(\sum_i \left(p_i^2 + p_{2i}\right)/2i\right). \tag{3}$$

In any given term, the degree of x_k gives the valency of vertex k. So, for example, the coefficient g_n of $x_1 \cdots x_n$ in G, hereafter denoted $[x_1 \cdots x_n]G$, gives the number of 1-regular graphs, or perfect matchings on the complete graph on n vertices, and in general the coefficient $g_n^{[k]} = [x_1^k \cdots x_n^k]G$, also given as $[m_{k^n}]G$, gives the number of k-regular graphs on n vertices. By virtue of Eq. (1), coefficient extraction amounts to a scalar product, and the generating function $G_k(t)$ of k-regular graphs is given by

$$G_k(t) := \sum_n g_n^{[k]} \frac{t^n}{n!} = \langle G, H_k \rangle ,$$

where

$$H_k(t) := \sum_n h_{k^n} \frac{t^n}{n!} = \sum_n \frac{(h_k t)^n}{n!} = \exp(h_k t). \tag{4}$$

Now, since $h_k = \sum_{\lambda \vdash k} p_\lambda / z_\lambda$ (where the sum is over all partitions λ of k), the exponential generating function $H_k(t)$ is also $\exp\bigl(t \sum_{\lambda \vdash n} p_\lambda / z_\lambda\bigr)$, an exponential in a finite number of p_i's. By property (3) in Theorem 2, this is D-finite. Further, as a result of scalar product property (2), we can rewrite Eq. (4) as

$$G_k(t) = \left\langle \exp\left(\sum_{i \text{ even},\, i \leqslant k} (-1)^{i/2} \frac{p_i^2}{2i} + \frac{p_i}{i} + \sum_{i \text{ odd},\, i \leqslant k} \frac{p_i^2}{2i} \right),\ \exp\left(t \sum_{\lambda \vdash k} \frac{p_\lambda}{z_\lambda} \right) \right\rangle \tag{5}$$

and now by Theorem 3 this generating function $G_k(t)$ is D-finite.

Note how the closed form for G in (3), in infinitely many variables, and the closed form for $H_k(t)$ in (4), in terms of the h's, have led to the scalar product (5) between two closed forms, explicitly written in terms of finitely many p_i for each k. This reduction is what has made the algorithm applicable.

3.2. *Effective computation for $k = 2$*

To illustrate a typical calculation, we calculate $G_2(t)$, the generating function for 2-regular graphs which, according to Eq. (5), is determined by

$$G_2(t) = \left\langle \exp\bigl((p_1^2 - p_2)/2 - p_2^2/4\bigr), \exp\bigl(t(p_1^2 + p_2)/2\bigr) \right\rangle .$$

Algorithm 1 calculates that $G_2(t)$ satisfies the differential equation

$$2(1-t)G_2'(t) - t^2 G_2(t) = 0,$$

which is easily solved to find $G_2(t) = e^{-\frac{1}{4}t(t+2)}/\sqrt{1-t}$.

In order to appeal to Algorithm 1, set $F = \exp((p_1^2 - p_2)/2 - p_2^2/4)$ and $G = \exp(t(p_1^2 + p_2)/2)$ and determine the Gröbner bases $\mathcal{G}_F$ and $\mathcal{G}_G$ of their annihilating ideals, respectively:

$$\mathcal{G}_F = \{\underline{p_2} + 2\partial_{p_2} + 1, \underline{p_1} - \partial_{p_1}\} \text{ and } \mathcal{G}_G = \{2\underline{\partial_{p_2}} - t, \underline{\partial_{p_1}} - tp_1, \underline{p_1^2} + p_2 - 2\partial_t\},$$

where $\mathcal{G}_F$ is a Gröbner basis with respect to the degree reverse lexicographical monomial ordering such that $p_1 > p_2 > \partial_{p_1} > \partial_{p_2}$ and $\mathcal{G}_G$ is a Gröbner basis with respect to the degree reverse lexicographical monomial ordering such that $\partial_{p_1} > \partial_{p_2} > p_1 > p_2 > \partial_t$. (Leading monomials with respect to the monomial ordering are underlined.) Before proceeding, the set $\mathcal{G}_F$ is converted by adjunction into a Gröbner basis $\mathcal{G}_F^\diamond$ with respect to the degree reverse lexicographical monomial ordering such that $\partial_{p_1} > \partial_{p_2} > p_1 > p_2$:

$$\mathcal{G}_F^\diamond = \{2\underline{\partial_{p_2}} + p_2 + 1, \underline{\partial_{p_1}} - p_1\}.$$

(The reader should not get confused by the peculiar situation of this example: here, adjunction has not changed the polynomials, except for signs, but this is only a coincidence.)

The initial value of B is the empty set. For the sake of the example, we shall iterate on monomials α according to the degree reverse lexicographical order such that $\partial_t > \partial_{p_2} > p_2 > \partial_{p_1} > p_1$, and perform reductions when inserting into the basis according to the elimination order sorting first by the degree reverse lexicographical order such that $\partial_{p_2} > p_2 > \partial_{p_1} > p_1$, and breaking ties by the degree in ∂_t.

We now briefly sketch the run of the algorithm until α becomes $p_1\partial_{p_1}$ and then illustrate the steps of the main loop in more details.

For $\alpha = 1$ and p_1, the algorithm inserts no polynomial into the basis B. The next iteration of the loop, for $\alpha = \partial_{p_1}$, produces $\alpha_F = \underline{\partial_{p_1}} - p_1$, which is inserted into B as is, and $\alpha_G = \underline{\partial_{p_1}} - tp_1$, whose insertion puts p_1 into B. Next, the case $\alpha = p_2$ inserts no polynomial before, for $\alpha = \partial_{p_2}$, $\alpha_F = 2\underline{\partial_{p_2}} + p_2 + 1$ gets inserted as is, and the insertion of $\alpha_G = 2\underline{\partial_{p_2}} - t$ puts $\underline{p_2} + (t+1)$ into B. The iteration for $\alpha = \partial_t$ has no effect on B. For $\alpha = p_1^2$, $\alpha_F = 0$ is not inserted, and $\alpha_G = \underline{p_1^2} + p_2 - 2\partial_t$ gets inserted in the form $\underline{p_1^2} - 2\partial_t - (t+1)$.

At this point, the algorithm is about to treat $\alpha = p_1\partial_{p_1}$ and the value of B is

$$B = \left\{\underline{\partial_{p_1}} - p_1, \underline{p_1}, 2\underline{\partial_{p_2}} + p_2 + 1, \underline{p_2} + (t+1), \underline{p_1^2} - 2\partial_t - (t+1)\right\}, \tag{6}$$

where we have written elements in the order of introduction into the set. In matrix notation, the column vector of elements of B reads

$$\begin{pmatrix} 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & t+1 \\ 1 & 0 & 0 & 0 & 0 & -2 & -(t+1) \end{pmatrix} \begin{pmatrix} p_1^2 \\ \partial_{p_2} \\ p_2 \\ \partial_{p_1} \\ p_1 \\ \partial_t \\ 1 \end{pmatrix}.$$

Here, we have chosen to keep the rows in the order of creation by the algorithm and to sort the column according to the monomial order used by the elimination step. Observe that in this way, no two rows have their left-most non-zero entry on the same column: simply reordering rows would put the matrix in row echelon form.

Then, the algorithm computes

$$\alpha_F = \alpha - (\alpha \operatorname{red}_{\preccurlyeq} \mathcal{G}_F^{\diamond}) = \alpha - (\alpha^{\diamond} \operatorname{red}_{\preccurlyeq_{\diamond}} \mathcal{G}_F)^{\diamond} = \underline{p_1 \partial_{p_1}} - p_1^2 + 1$$

and

$$\alpha_G = \alpha - (\alpha_{\operatorname{red}_{\preccurlyeq}} \mathcal{G}_G) = \underline{p_1 \partial_{p_1}} + t p_2 - 2t\partial_t.$$

(Note that α_F is really $(\partial_{p_1} - p_1)p_1$, an element of the *right* ideal generated by $\mathcal{G}_F^{\diamond}$.) Next, we update B to include these two values. We insert α_F into B after one reduction, leading to

$$B := B \cup \{\underline{p_1 \partial_{p_1}} - 2\partial_t - t\}.$$

In matrix notation, this insertion adds a new column to the left of the matrix, corresponding to the new monomial $p_1 \partial_{p_1}$, and one more row at the bottom of the matrix, $\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & -2 & -t \end{pmatrix}$. Then the algorithm inserts α_G. Its leading monomial $p_1\partial_{p_1}$ is already present in B, leading to an initial reduction to $t\underline{p_2} + 2(1-t)\partial_t + t$. One final reduction by t times the pre-last element in Eq. (6) results in the step

$$B := B \cup \{2(1-t)\underline{\partial_t} - t^2\}.$$

The intersection of this and $W_t(t)$ is non-trivial, and the algorithm outputs $2(1-t)\partial_t - t^2$. We conclude that $G_2(t)$ satisfies the differential equation

$$2(1-t)G_2'(t) - t^2 G_2(t) = 0.$$

Table 1 summarizes the results by the same algorithm for $k = 2, 3, 4$. These match with the results in [13].

3.3. *Efficient enumeration of k-regular graphs*

An efficient procedure for the enumeration of k-regular graphs is immediately derived from the differential equations for the generating series of k-regular graphs collected in Table 1. Indeed, one simply needs to convert the differential equation for $G_k(t)$ into a

Table 1
Differential equation $\phi_2 G_k'' + \phi_1 G_k' + \phi_0 G_k = 0$ satisfied by $G_k(t)$, $k = 2, 3, 4$

2-*Regular graphs*	
ϕ_0	$-t^2$
ϕ_1	$-2t + 2$
ϕ_2	0
3-*Regular graphs*	
ϕ_0	$t^3(t^4 + 2t^2 - 2)^2$
ϕ_1	$-3(t^{10} + 6t^8 + 3t^6 - 6t^4 - 26t^2 + 8)$
ϕ_2	$-9t^3(t^4 + 2t^2 - 2)$
4-*Regular graphs*	
ϕ_0	$-t^4(t^5 + 2t^4 + 2t^2 + 8t - 4)^2$
ϕ_1	$-4(t^{13} + 4t^{12} - 16t^{10} - 10t^9 - 36t^8 - 220t^7 - 348t^6$ $-48t^5 + 200t^4 - 336t^3 - 240t^2 + 416t - 96)$
ϕ_2	$16t^2(t-1)^2(t^5 + 2t^4 + 2t^2 + 8t - 4)(t+2)^2$

recurrence relation for its coefficients $g_n^{[k]}$ and to determine sufficiently many starting values $g_0^{[k]}, g_1^{[k]}, \ldots$. Then, one can efficiently compute $g_n^{[k]}$ for any n by unrolling the recurrence.

Implementations are available to help with this approach. For example, the Maple package `gfun`[3] by Salvy and Zimmerman [31] contains commands dedicated to the conversion step and the iterative calculations based on a linear recurrence. Computations in the case $k = 4$ result in a recurrence relation of order 15 already published by Read and Wormald [29] and can be found as a formula accompanying sequence number A005815 in Sloane's encyclopedia of integer sequences [32]. From this recurrence relation and initial terms, it is then a matter of seconds to compute the exact integer values for hundreds of terms in the sequence.

It should be stressed that this method proves much more efficient than the direct computation of the scalar product based on a termwise expansion and application of formula (2). For example, Stembridge's implementation in the package SF for symmetric function manipulation in Maple [35] already requires several minutes to compute the $g_n^{[4]}$ for n up to 15, and becomes unsuitable to handle the symmetric functions that would be necessary to obtain $g_{20}^{[4]}$. Far from showing any weakness of SF's general approach, this illustrates the computational progress provided by our techniques in the specific setting of differentiably finite series.

3.4. Generalization

The series given by Eq. (3) is determined combinatorially in a direct fashion using the theory of species [1]. This can be extended naturally to handle a wider family of combinato-

[3] This package is part of the `algolib` library, which is available at `http://algo.inria.fr/packages/`.

rial structures, such as hypergraphs, set covers, latin rectangles. For an in-depth treatment, consult [26].

4. Hammond series

In the example above, it turned out that except for monomials of degree 1, we needed only examine the two monomials p_1^2 and $p_1\partial_{p_1}$ in order to reach the solution. However, depending on the monomial ordering, the algorithm might well consider many monomials before it adds the ones that eliminate the p_i's and ∂_{p_i}'s. The problem becomes far more serious as the number of variables and the degree of the monomials increase. It turns out that in the common case when the scalar product is of the type $\langle F, H_k(t)\rangle$ it is possible to modify the approach and eliminate the p_i and the ∂_{p_i} in a more efficient manner using the *Hammond series*[4] (or H-series) introduced by Goulden, Jackson, and Reilly in [13]: for $F \in K[[p_1, p_2, \ldots]]$, the Hammond series of F is defined as

$$\mathcal{H}(F)(t_1, t_2, \ldots) = \left\langle F, \sum_\lambda h_\lambda t^\lambda / m(\lambda)! \right\rangle,$$

where the sum is over all partitions, and if $\lambda = 1^{m_1} \cdots k^{m_k}$ then $t^\lambda = t_1^{m_1} \cdots t_k^{m_k}$ and $m(\lambda)! = m_1! m_2! \cdots m_k!$. These are very closely related to the Hammond operators, defined by Hammond [15] and used extensively by MacMahon [22]. A Hammond operator can be described as $h_\lambda^\diamond$, and thus the Hammond series of F with all of the t variables set to 1 results essentially in a sum of Hammond operators acting on F.

Observe that the generating function for k-regular graphs is

$$G_k(t) = \mathcal{H}(G)(0, \ldots, 0, t, 0, \ldots),$$

where the t occurs in position k. This is true for any generating function which takes the form $\langle F, H_k(t)\rangle$ for some F.

A theorem from [13] is specially useful: Goulden, Jackson, and Reilly's H-series theorem states that $\mathcal{H}(\partial_{p_n} \cdot F)$ and $\mathcal{H}(p_n F)$ can be expressed in terms of the $\partial_{t_i} \cdot \mathcal{H}(F)$'s. In terms of Gröbner bases, this corresponds to introducing the additional variables $t_1, \ldots, t_k$ (instead of $t = t_k$ alone) and work with the series $\mathcal{H}_k(t_1, \ldots, t_k) = \sum_\lambda h_\lambda t^\lambda / m(\lambda)!$ with sum over partitions λ whose largest part is k (instead of working with the univariate $H_k(t)$). The H-series theorem therefore implies that for an appropriate monomial order, there is a Gröbner basis of the ideal $I_{\mathcal{H}_k}$ of all operators of $W_{p,t}$ annihilating $\mathcal{H}_k$, with elements of the form

$$p_i - P_i(t, \partial_t), \quad \partial_{p_i} - Q_i(t, \partial_t), \qquad i = 1, \ldots, k, \tag{7}$$

where all the P_i and Q_i are polynomials in t, ∂_t.

[4] In [12, Section 3.5] this is referred to as the *Gamma series* of F.

The algorithm in this case is as follows.

Algorithm 2 (*Hammond series*).
Input: *An integer k, and* $F \in K[[p_1, \ldots, p_n]]$.
Output: *A differential equation satisfied by*

$$\left\langle F, \sum_i h_{k^i} t_k^i \right\rangle = \mathcal{H}(F)(0, \ldots, 0, t_k, 0, \ldots),$$

where t_k *is in position k.*

(1) *Compute* $\mathcal{G}_F$, *a Gröbner basis for the left ideal* J_F *annihilating F in* W_p.
(2) *Compute* $\mathcal{G}_{\mathcal{H}_k}$, *a Gröbner basis of the form* (7).
(3) *For each* $U \in \mathcal{G}_F$, *compute* $r_U \in W_t$ *as the reduction of* $U^\diamond$ *by* $\mathcal{G}_{\mathcal{H}_k}$ *for an order which eliminates* p, ∂_p. *Let* R_0 *be the set of* r_U*'s.*
(4) *For i from* 1 *to* $k-1$ *eliminate* ∂_{t_i} *from* R_{i-1} *and set* $t_i = 0$ *in the resulting polynomials*; *call* R_i *the new set.*
(5) *Return* R_{k-1}.

As with Algorithm 1, the first step is to determine an annihilating ideal in W_p. Again, one can possibly first determine a D-finite description and use Weyl closure [40,41] to obtain the annihilating ideal.

After Step (3), all the p_i's and ∂_{p_i}'s have been eliminated and R_0 contains a set of generators of a D-finite $W_t(t)$-ideal annihilating $\langle F, \mathcal{H}_k\rangle$. Then, in order to obtain differential equations satisfied by the specialization at $t_1 = \cdots = t_{k-1} = 0$, Step (4) proceeds in order by eliminating differentiation with respect to t_i and then setting $t_i = 0$ in the remaining operators.

Note that the Gröbner basis of Step (2) can be precomputed for the required k's (although most of the time is actually spent in Step (4)).

In order to compute the elimination in Step (4), one should not compute a Gröbner basis for an elimination order, since this would in particular perform the unnecessary computation of a Gröbner basis of the eliminated ideal. Instead, one can modify the main loop in the Gröbner basis computation so that it stops as soon as sufficient elimination has been performed or revert to skew elimination by the non-commutative version of the extended Euclidean algorithm as described in [5]. This is the method we have adopted in the example session given in Appendix B. [5]

This calculation is comparatively rapid since the size of the basis is greatly reduced. Further, the basis grows smaller as the algorithm progresses, on account of setting variables to 0. We can compute the case of 4-regular graphs in a second, instead of a couple of minutes using the general algorithm. The 5-regular expression requires significantly more computation time, and we could not compute it.

[5] An implementation of the algorithms presented here is available in the Maple package ScalarProduct available at http://algo.inria.fr/mishna.

A mathematically equivalent but slightly faster way of performing Step (3) is to compute r_U by simply replacing each monomial $p_1^{\alpha_1} \cdots p_n^{\alpha_n} \partial_{p_1}^{\beta_1} \cdots \partial_{p_n}^{\beta_n}$ in U with the product $Q_n^{\beta_n} \cdots Q_1^{\beta_1} P_n^{\alpha_n} \cdots P_1^{\alpha_1}$.

In order to explain the relative speed of Algorithm 2, compared to Algorithm 1, it needs to be said that the Hammond series approach searches a smaller space, which can well result in a differential equation of order higher than that obtained by Algorithm 1. This occurs, for instance, in the case of 4-regular graphs: Algorithm 2 returns a differential equation of order 3 only when that returned by Algorithm 1 is of order 2.

In the same vein, note that the order in which the eliminations are done in Step (4) could be changed, possibly leading to a different (but correct) output.

4.1. *Proof of termination and correctness*

Termination of Algorithm 2 is obvious. On the other hand, the full proof of correctness requires technical results to be proved in Section 7. The following corollary articulates a property of D-finite functions in the simple language of symmetric functions and D-finite descriptions, and is a corollary of Proposition 9 that will be proved independently.

Corollary 4. *Let F and G be D-finite symmetric series in $K[[p_1, \ldots, p_n]]$ and $K[t_1, \ldots, t_k][[p_1, \ldots, p_n]]$, respectively, with corresponding annihilators $J_F \subset W_p$ and $\mathcal{I}_G \subset W_{p,t}(p, t)$. Under these conditions, the vector space*

$$\left(J_F^{\diamond} W_t(t) + \mathcal{I}_G\right) \cap W_t(t)$$

is non-trivial and contains a D-finite description of $\langle F, G\rangle$.

Proposition 5. *Algorithm* 2 *terminates and is correct.*

Proof. First, we remark that for fixed k,

$$\mathcal{H}_k(t_1, \ldots, t_k) = \exp\left(\sum_{j=1}^{k} h_j t_j\right)$$

is a D-finite symmetric series by Theorem 2 since each h_j is a finite combination of $p_1, \ldots, p_n$. Thus, $f = H(F)(t_1, \ldots, t_k) = \langle \mathcal{H}_k(t_1, \ldots, t_k), F\rangle$ is a D-finite function of $t_1, \ldots, t_k$, by Theorem 3.

We proceed by proving the following invariant of the main loop: the set R_{i-1} generates a D-finite description of $\mathcal{H}(F)(0, \ldots, 0, t_i, t_{i+1}, \ldots, t_k)$. This establishes the result since it implies that R_{k-1} contains a D-finite description of $\mathcal{H}(F)(0, \ldots, 0, t_k)$, in this case, a single differential equation. This is precisely what the algorithm claims to determine.

To prove the base case of this invariant, note that R_0 contains the generators of the intersection $\left(J_F^{\diamond} W_t(t) + \mathcal{I}_{\mathcal{H}_k}\right) \cap W_t(t)$. We appeal to Corollary 4, to conclude that R_0 contains a D-finite description of $\mathcal{H}(F)(t_1, \ldots, t_k)$.

The general case is proven with the known result [5] that given a D-finite description of a function $F(x_1, x_2, \ldots, x_n)$, one can compute the D-finite description of $F(x_1, \ldots, x_{n-1}, 0)$, for example, by first eliminating ∂_{x_n}, removing factors of x_n in the remaining polynomials, and finally, setting $x_n = 0$ in the equations, precisely the process outlined in Algorithm 2. □

5. Example: *k*-uniform tableaux

Another family of combinatorial objects whose generating function can be resolved with our method is a certain class of Young tableaux, namely k-uniform Young tableaux.

For a partition $\lambda = (\lambda_1, \dots, \lambda_k) \vdash n$, a Young tableau of shape λ is an array $T = (T_{i,j})$ of positive integers $T_{i,j}$ defined when $1 \leqslant i \leqslant k$ and $1 \leqslant j \leqslant \lambda_i$. When a Young tableau is strictly increasing on each of its rows and columns ($T_{i,j} < T_{i+1,j}$ and $T_{i,j} < T_{i,j+1}$, whenever this makes sense) and the n integers $T_{i,j}$ are all integers from 1 to n, it is called standard.

Standard Young tableaux are in direct correspondence with many different combinatorial objects. For example, Stanley [34] has studied the link between standard tableaux and paths in Young's lattice, the lattice of partitions ordered by inclusion of diagrams. This link was generalized by Gessel [10] to tableaux with repeated entries. Gessel remarks that such paths have arisen in the work of Sundaram on the combinatorics of representations of symplectic groups [36].

The weight of a tableau is $\mu = (\mu_1, \dots, \mu_k)$ where μ_1 is the number of 1's, μ_2 is the number of 2's, etc., in the tableau entries. Here we consider Young tableaux that are column strictly increasing and row weakly increasing, and with weight $\mu = 1^k 2^k \cdots n^k$: each entry appears k times. We call Young tableaux with these properties *k-uniform*. These correspond to paths in Young's lattice with steps of length k. The set of k-uniform tableaux of size kn is also in bijection with symmetric $n \times n$ matrices with non-negative integer entries with each row sum equal to k. Gessel notes that for fixed k, the generating series of the number of k-uniform tableaux is D-finite [9]. Our method makes this effective.

Two observations from [21] are essential. First, $[x_1^{\mu_1} \cdots x_k^{\mu_k}] s_\lambda$ is the number of (column strictly increasing, row weakly increasing) tableaux with weight μ. Secondly,

$$\sum_\lambda s_\lambda = h[e_1 + e_2] = \exp\left(\sum_i p_i^2/2i + \sum_{i \text{ odd}} p_i/i\right),$$

which is D-finite. Define $y_n^{[k]}$ to be the number of k-uniform tableaux of size kn, and let Y_k be the generating series of these numbers. The previous two observations imply

$$Y_k(t) = \sum_n y_n^{[k]} t^k = \left\langle \exp\left(\sum_{i=1}^k p_i^2/2i + \sum_{i \text{ odd}}^k p_i/i\right), \sum_n h_{k^n} t^n \right\rangle. \tag{8}$$

This problem is well-suited to our methods since again we treat an exponential of a polynomial in the p_i's, with an explicit closed form in terms of k for this polynomial.

Calculating the equations for $k = 1, 2, 3, 4$ is fast with either Algorithm 1 or Algorithm 2. The resulting differential equations are listed in Table 2. For $k = 1, 2$ these results agree with known results [14,34], and are the entries A000085, and A000985, respectively in Sloane's encyclopedia of integer sequences [32]. The first few values of $y_n^{[k]}$ are summarized in Table 3. For $k = 3, 4$ these appear to be new.

Table 2
Differential equation $\phi_2 Y_k'' + \phi_1 Y_k' + \phi_0 Y_k = 0$ satisfied by $Y_k(t)$, $k = 1, \dots, 4$

1-*Uniform tableaux*	
ϕ_0	$-(t-1)$
ϕ_1	1
ϕ_2	0
2-*Uniform tableaux*	
ϕ_0	$t^2(t-2)$
ϕ_1	$-2(t-1)^2$
ϕ_2	0
3-*Uniform tableaux*	
ϕ_0	$(t^{11} + t^{10} - 6t^9 - 4t^8 + 11t^7 - 15t^6 + 8t^5 - 2t^3 + 12t^2 - 24t - 24)$
ϕ_1	$-3t(t^{10} - 2t^8 + 2t^6 - 6t^5 + 8t^4 + 2t^3 + 8t^2 + 16t - 8)$
ϕ_2	$9t^3(-t^2 - 2 + t + t^4)$
4-*Uniform tableaux*	
ϕ_i	(see Appendix A)

Table 3
The number, $y_n^{[k]}$, of k-uniform tableaux of size kn

k	$y_0^{[k]}, y_1^{[k]}, y_2^{[k]}, \dots$
1	1, 1, 2, 4, 10, 26, 76, 232, 764, 2620, 9496, 35696, 140152, 568504
2	1, 1, 3, 11, 56, 348, 2578, 22054, 213798, 2313638, 27627434, 360646314, 5107177312, 77954299144
3	1, 1, 4, 23, 214, 2698, 44288, 902962, 22262244, 648446612, 21940389584, 849992734124
4	1, 1, 5, 42, 641, 14751, 478711, 20758650, 1158207312, 80758709676, 6877184737416, 701994697409136

Concerning the dual problem, where instead n is fixed and k varies, the sequences $\left(y_n^{[k]}\right)_k$ appear, respectively, as A019298, A053493, and A053494 for $n = 3, 4, 5$. Stanley [33, Proposition 4.6.21] reports that the generating functions $G_n(x) = \sum_k y_n^{[k]} x^k$ are rational with denominator of the form $(1-x)^s(1-x^2)^t$ where s and t are positive integers.

6. Algorithm for scalar product: the general situation

So far, we have limited the scope of the algorithms to pairs of D-finite symmetric functions where only one of the two functions depends on the variables $t_1, \dots, t_k$. While this is sufficient in many applications, it is possible to modify Algorithm 1 in order to accommodate the t_i's in both functions and thus make the full power of Theorem 3 effective. While no additional ideas are to be used, the description of the algorithm is more technical.

Algorithm 1 manipulates monomials α and reduces them modulo the ideals $\mathcal{I}_F$ and $\mathcal{I}_G$ in order to determine equations of the form

$$\left\langle F, \left(\alpha - (\alpha \operatorname{red}_{\preccurlyeq} \mathcal{I}_F^{\diamond})\right) \cdot G\right\rangle = 0 \quad \text{and} \quad \left\langle F, \left(\alpha - (\alpha \operatorname{red}_{\preccurlyeq} \mathcal{I}_G)\right) \cdot G\right\rangle = 0, \tag{9}$$

where on the left, α supposedly does not involve any of the ∂_{t_i}'s. What makes the situation of Algorithm 1 and the left-hand identity in (9) simple is the assumption that F does not depend on t, making the action of W_t on $\langle F, G\rangle$ act on the right-hand argument only. The difficulty in generalizing lies in that now, the action of ∂_{t_i} on F may be non-trivial and must be considered in the differentiation rule for scalar products,

$$\partial_{t_i} \cdot \langle F, G\rangle = \left\langle \partial_{t_i} \cdot F, G\right\rangle + \left\langle F, \partial_{t_i} \cdot G\right\rangle, \tag{10}$$

which itself stems from the differentiation rule for usual products on the level of coefficients.

The idea is therefore to manipulate operators in *three* sets of ∂_{t_i}'s: one which acts on the full scalar product $\langle F, G\rangle$, and one for each of its components, acting directly on the component. To facilitate the description of this situation, we denote the former by ∂_{t_i}, the one acting on the left component by ∂_{ℓ_i}, and the one acting on the right component ∂_{r_i}. Using this notation, we wish to view Eq. (10) as

$$\partial_{t_i} = \partial_{\ell_i} + \partial_{r_i}. \tag{11}$$

We thus modify Algorithm 1 by enlarging the family of monomials over which we iterate, and use Eq. (11) to eliminate the ∂_{ℓ_i}'s before we begin Gaussian elimination. Here, we iterate over monomials $\alpha\partial_\ell^\beta\partial_r^\gamma$ of the free commutative monoid $[p, \partial_p, \partial_\ell, \partial_r]$ with $\alpha \in [p, \partial_p]$ to examine the following generalizations of Eq. (9):

$$\left\langle \left(\alpha^\diamond \partial_t^\beta - (\alpha^\diamond \partial_t^\beta \operatorname{red} \mathcal{G}_F)\right) \cdot F, \partial_t^\gamma \cdot G \right\rangle = 0 \tag{12}$$

and

$$\left\langle \partial_t^\beta \cdot F, \left(\alpha\partial_t^\gamma - (\alpha\partial_t^\gamma \operatorname{red} \mathcal{G}_G)\right) \cdot G \right\rangle = 0,$$

or, with a change of notation,

$$\left(\alpha^\diamond \partial_\ell^\beta - (\alpha^\diamond \partial_\ell^\beta \operatorname{red} \mathcal{G}_F)\right)\partial_r^\gamma \cdot \langle F, G\rangle = 0$$

and

$$\partial_\ell^\beta\left(\alpha\partial_r^\gamma - (\alpha\partial_r^\gamma \operatorname{red} \mathcal{G}_G)\right) \cdot \langle F, G\rangle = 0.$$

Upon making use of Eq. (11) and applying adjunction to the first equation in Eq. (12), we get a linear combination of terms of the form $\partial_t^{\beta'} \cdot \left\langle F, \alpha' \cdot G\right\rangle$ with coefficients in $K[t]$, where $\beta' \in \mathbb{N}^k$, and $\alpha' \in W_{p,t}(t)$. The algorithm proceeds as before by performing Gaussian elimination over $K(t)$ to eliminate p, ∂_p, and ∂_r. In our implementation, the monomial order $\preccurlyeq$ is DegRevLex$(\partial_r > \partial_\ell > \partial_p > p)$. The method is summarized in Algorithm 3.

Algorithm 3 (*General scalar product*).

Input: *$F \in K[t][[p]]$ and $G \in K[t][[p]]$, both D-finite in p, t, given by D-finite descriptions in $W_{p,t}(t)$.*

Output: *A D-finite description of $\langle F, G\rangle$ in $W_t(t)$.*

```
(1) Determine a Gröbner basis $\mathcal{G}_G$ for the left ideal $\operatorname{ann}_{W_{p,t}(t)} G$ with respect to any monomial
    ordering $\preccurlyeq$, as well as a Gröbner basis $\mathcal{G}_{F^\diamond}$ for the right ideal $\operatorname{ann}_{W_{p,t}} F^\diamond$ with respect
    to the same ordering.
(2) $B := \{\}$.
(3) Iterate through each monomial $\alpha$ in $p, \partial_p, \partial_\ell, \partial_r$ in any order.
    (a) $\alpha_l := \alpha|_{\partial_\ell=\partial_t, \partial_r=1}$.
    (b) $\alpha_F := \alpha_l - (\alpha_l \operatorname{red}_{\preccurlyeq} \mathcal{G}_{F^\diamond})$.
    (c) $\alpha_r := \alpha|_{\partial_r=\partial_t, \partial_\ell=1}$.
    (d) $\alpha_G := \alpha_r - (\alpha_r \operatorname{red}_{\preccurlyeq} \mathcal{G}_G)$.
    (e) Introduce $(\alpha_F|_{\partial_\ell=\partial_t-\partial_r})(\alpha|_{p=\partial_p=\partial_\ell=1})$ and $(\alpha|_{p=\partial_p=\partial_r=1})\alpha_G$ into B and reduce so
        as to eliminate $p, \partial_p, \partial_r$.
    (f) Compute the dimension of the ideal generated by $B \cap W_t(t)$. If this dimension is 0,
        break and output $B \cap W_t(t)$.
```

As in Algorithm 1, if $m = 1$, there is only one variable t, and the condition in (3f) is simplified to:

If B contains a non-zero element P from $W_t(t)$, break and return P.

The same remarks as those made after Algorithm 1 at the end of Section 2 also apply here.

7. Termination and correctness of Algorithms 1 and 3

7.1. Sketch of the proof

The common goal of Algorithms 1 and 3 is to find differential equations satisfied by $\langle F, G\rangle$, which is equivalent to non-zero elements in W_t which annihilate $\langle F, G\rangle$. Although Algorithm 1 is a specialization of Algorithm 3, parts of the proof would become artificially more involved if restricted to the simple case. We thus treat both algorithms simultaneously. The discussion at the beginning of Section 2 has illustrated how to manipulate the annihilators of F and G to determine a combination $P^\diamond S + TQ \in W_t$ with $P \in \mathcal{I}_F^\diamond$, $Q \in \mathcal{I}_G$, $S \in W_p(t)$, $T \in W_{p,t}(t)$, which annihilates $\langle F, G\rangle$. Not all of the elements in $\operatorname{ann}_{W_t} \langle F, G\rangle$ are of this form, however, as the following simple example illustrates. If $F = p_1 - p_2$ and $G = p_1 + p_2/2$, then $\langle F, G\rangle = 1 - 1 = 0$ and thus $1 \in \operatorname{ann}_{W_t} \langle F, G\rangle$. However, it can be established that 1 can not be written as a combination of the form $P^\diamond S + TQ$ for those F and G. Nonetheless, we show that the annihilating elements that can be written this way form a non-trivial subideal of $\operatorname{ann}_{W_t} \langle F, G\rangle$, which we generate with the algorithms.

Although the problem of finding differential equations appears at first inherently analytic in nature, we rephrase it algebraically into a question amenable to the theory of D-modules.

The adjunction properties of the scalar product are naturally accommodated by tensor products. Specifically, the proof below centers around a certain W_t-module S whose elements are tensors, and where, for example,

$$(i^{-1}p_i \cdot u) \otimes v = (u \cdot \partial_{p_i}) \otimes v = u \otimes (\partial_{p_i} \cdot v),$$

which corresponds to the equivalence $\big\langle (i^{-1}p_i) \cdot F, G \big\rangle = \big\langle F, \partial_i \cdot G \big\rangle$. (See also Eqs. (13–16)) below.) On the other hand, the ∂_{ℓ_i} and ∂_{r_i} that are involved in the description of Algorithm 3 really are the operators $\partial_{t_i} \otimes 1$ and $1 \otimes \partial_{t_i}$ acting on S, respectively, where 1's denote identity maps.

The module S can be expressed in terms of the ideal $\operatorname{ann}_{W_t}(F^\diamond \otimes G)$, itself contained in $\operatorname{ann}_{W_t} \langle F, G \rangle$. The former ideal is non-trivial and in fact, is sufficient to describe the scalar product as holonomic, a property whose definition is recalled shortly and which implies D-finiteness. In fact, we show that the algorithms calculate a Gröbner basis for $\operatorname{ann}_{W_t(t)}(F^\diamond \otimes G)$, in other words a D-finite description of the scalar product $\langle F, G \rangle$.

The main result is summarized by the following theorem.

Theorem 6. *Suppose F and G are symmetric functions subject to the conditions of Algorithm* 1 (*resp. Algorithm* 3). *Then, Algorithm* 1 (*resp. Algorithm* 3) *determines, in finite time, a Gröbner basis for a non-zero D-finite ideal contained in* $\operatorname{ann}_{W_t(t)} \langle F, G \rangle$.

The notion of holonomy to be used in the proof follows [2,7]. Introduce a filtration of W_t by the K-vector spaces F_d of all operators in W_t of total degree at most d in t, ∂_t. These spaces are finite-dimensional, of dimension $\binom{d+2k}{2k} = O\big(d^{2k}\big)$ as d tends to infinity. A W_t-module $M = \sum_i W_t \cdot g_i$ generated by a finite family of generators g_i is holonomic whenever the K-vector spaces $\sum_i F_d \cdot g_i$ have dimension growing like $O\big(d^k\big)$. A function of t that is an element of a holonomic W_t-module is called holonomic. From the definition, it is a basic result that a holonomic function is D-finite; the converse is a more difficult result to be found in [38, Theorem 2.4 and Appendix 6]. Similar definitions apply to $W_{p,t}$-modules, with a dimension growth of $O\big(d^{k+n}\big)$ in place of $O\big(d^k\big)$.

The discussion so far has not relied on the definition of the scalar product. Rather, remark that Algorithms 1 and 3 are essentially parameterized by the adjunction property of the scalar product of symmetric functions, and can easily be redefined and adapted to other adjunctions. It suits our needs for the proof to consider adjoints for the usual scalar product of functions, $\langle f | g \rangle := \int f(x) g(x)\, dx$. To avoid confusion, we notationally distinguish $\langle f | g \rangle$ from $\langle F, G \rangle$ for the two scalar products, as well as $\star$ from $\diamond$ for the respective adjunction operations.

Indeed, guided by existing results concerning the preservation of holonomy under operations involving the usual scalar product, we link the symmetric case to the usual one with a map from one adjunction to the other. This reduction also demonstrates how algorithms analogous to Algorithms 1 and 3 for other scalar products could be shown to terminate with the correct output. (See Section 9.1.)

To make this comparison more intuitive, we could identify $\langle F, G \rangle$ with the integral

$$\int_{\mathbb{R}^n} \mathcal{L}\big(q \mapsto F(q_1, 2q_2, \ldots, nq_n)\big)(p)\, G(p)\, dp_1 \cdots dp_n,$$

where $\mathcal{L}$ is the modified Laplace transform

$$\mathcal{L}(F)(p) = \int_{\mathbb{R}^n} F(q)e^{-(p_1q_1+\cdots+p_nq_n)}\,dq,$$

which satisfies

$$\mathcal{L}\big(q \mapsto q_i F(q)\big)(p) = -(\partial_{p_i} \circ \mathcal{L})(F)(p).$$

Note, for example

$$\begin{aligned}\left\langle i^{-1}p_i \cdot F, G\right\rangle &= \int_{\mathbb{R}^n} \mathcal{L}\big(q \mapsto q_i F(q_1, \dots, nq_n)\big)(p)\, G(p)\, dp_1 \cdots dp_n \\ &= -\int_{\mathbb{R}^n} (\partial_{p_i} \circ \mathcal{L})(F)(p)\, (\partial_{q_i} \cdot G)(p)\, dp_1 \cdots dp_n \\ &= \int_{\mathbb{R}^n} \mathcal{L}\big(q \mapsto F(q_1, \dots, nq_n)\big)(p)\, (\partial_{q_i} \cdot G)(p)\, dp_1 \cdots dp_n \\ &= \left\langle F, \partial_{p_i} \cdot G\right\rangle. \end{aligned} \tag{13}$$

Formally, we must work on the level of abstract modules, however. This avoids situations where the integral is not convergent or the Laplace transform is not defined as a function.

Thus, to prove Theorem 6, we show Corollary 10 below which states that $\mathrm{ann}_{W_t}\big(F^\diamond \otimes G\big)$ is a non-zero subideal of $\mathrm{ann}_{W_t}\langle F, G\rangle$ such that the quotient $W_t/\mathrm{ann}_{W_t}\big(F^\diamond \otimes G\big)$ is a holonomic module. This is done in several stages. First, in Section 7.2, we define S, the algebraic structure in which our calculations take place, and prove that it is holonomic by reducing the problem to the usual scalar product analogue, where similar results are known. This analogue is detailed in Section 7.3. Next, in Section 7.4 we express S as a quotient. Corollary 10 follows from this discussion. Finally, to conclude that the algorithm terminates, we relate S to the algorithm in more detail and prove in Section 7.5 that all of the generators are determined in finite time. Together, these results prove Theorem 6 and thus the correctness and termination of Algorithms 1 and 3.

7.2. *The scalar product of symmetric functions*

We now formally define the W_t-module S. Begin with $U = W_{p,t} \cdot F$ and $V = W_{p,t} \cdot G$, two holonomic $W_{p,t}$-modules. We shall denote by $U^\diamond$ the adjoint module of U: as K-vector spaces, $U = U^\diamond$, and a right $W_p[t]$-action is defined on $U^\diamond$ by $u \cdot P = P^\diamond \cdot u$ for any $u \in U^\diamond$ and $P \in W_p[t]$, where the last operation is taken for the left structure of U. Set S as the tensor product $U^\diamond \otimes_{W_p[t]} V$, which makes it a $K[t]$-module. This has the desirable effect of encoding the scalar product adjunction relations: for all $u \in U$ and all $v \in V$,

$$(\partial_{p_i} \cdot u) \otimes v = (u \cdot \partial_{p_i}^\diamond) \otimes v = (u \cdot i^{-1}p_i) \otimes v = u \otimes (i^{-1}p_i \cdot v), \tag{14}$$

$$(p_i \cdot u) \otimes v = (u \cdot p_i^\diamond) \otimes v = (u \cdot i\partial_{p_i}) \otimes v = u \otimes (i\partial_{p_i} \cdot v), \tag{15}$$

$$t_i \cdot (u \otimes v) = (t_i \cdot u) \otimes v = (u \cdot t_i) \otimes v = u \otimes (t_i \cdot v). \tag{16}$$

To endow S with a W_t-module structure, let ∂_{t_i} act on a pure tensor $u \otimes v$ by

$$\partial_{t_i} \cdot (u \otimes v) = (\partial_{t_i} \cdot u) \otimes v + u \otimes (\partial_{t_i} \cdot v), \tag{17}$$

and extend to S by K-linearity. In other words, $\partial_{t_i} = \partial_{\ell_i} + \partial_{r_i}$ after defining $\partial_{\ell_i} = \partial_{t_i} \otimes 1$ and $\partial_{r_i} = 1 \otimes \partial_{t_i}$, where 1's are identity maps.

Armed with this definition and Theorem 7 (formally stated and proven independently in Section 7.3), we prove that S is holonomic. Theorem 7 is an analogous result for the usual scalar product, corresponding adjunction, and corresponding adjoint module $M^\star$ of a module M. It states that for holonomic M and N, $M^\star \otimes_{W_p[t]} N$ is a holonomic W_t-module under the action of ∂_{t_i} given by (17). We shall appeal to this theorem with an appropriate choice for M and N.

To determine the relationship between the two scalar products and make our choice for M and N, we compare both adjunction operations. In the symmetric case, adjunction is defined as the anti-automorphism $\diamond$ which maps p_i to $i\partial_{p_i}$ and ∂_{p_i} to $i^{-1}p_i$, for all i, and the usual scalar product adjunction is defined as the anti-automorphism $\star$ which maps ∂_{p_i} to $-\partial_{p_i}$, and leaves the p_i variables unchanged. One way to connect both adjunctions is to factor $\diamond$ into the composition of three algebra morphisms:

(1) the automorphism τ mapping (p_i, ∂_i) to $(ip_i, i^{-1}\partial_i)$. This corresponds to the dilation which maps a function F to $p \mapsto F(p_1, 2p_2, \dots, np_n)$;
(2) the automorphism $\mathcal{F}$ mapping (p_i, ∂_i) to $(-\partial_i, p_i)$ and named 'Fourier transform' in D-module theory (see [2, proof of Theorem 3.1.8] or [7, p. 39]). Informally speaking, this corresponds to mapping a function F to its Laplace transform $\mathcal{L}(F)$;
(3) the anti-automorphism $\star$ mapping (p_i, ∂_i) to $(p_i, -\partial_i)$.

The important property to note is that each of these three maps preserves holonomy since they preserve total degree, hence are filtration-preserving bijections. A direct calculation on p_i and ∂_i verifies that $\diamond = \star \circ \mathcal{F} \circ \tau$, so that the composite $\diamond$ also is a holonomy-preserving linear bijection. Thus, we introduce two holonomic modules, $M = (\mathcal{F} \circ \tau)(U)$ also denoted $U^{\mathcal{F}\circ\tau}$, and $N = V$, so as to appeal to Theorem 7. One concludes that

$$S = U^\diamond \otimes_{W_p[t]} V = \left(U^{\mathcal{F}\circ\tau}\right)^\star \otimes_{W_p[t]} V = M^\star \otimes_{W_p[t]} N \tag{18}$$

is a holonomic W_t-module. After we have described the quotient structure of S in Section 7.4, this information will be used to prove that $\mathrm{ann}_{W_t}(F^\diamond \otimes G)$ is non-trivial and that the quotient module $W_t/\mathrm{ann}_{W_t}(F^\diamond \otimes G)$ is holonomic, a fact we use to show that the algorithms terminate.

7.3. Preservation of holonomy under the usual scalar product

In the previous section, we reduced the proof of the holonomy of $S = U^\diamond \otimes_{W_p[t]} V$ to an analogous result in terms of the usual scalar product, to be proven in this section: the module $T = M^\star \otimes_{W_p[t]} N$ is holonomic when M and N are.

The following notion will be used in the proof: the integral of a $W_{p,t}$-module P, denoted $\int P = \int P \, dp_1 \cdots dp_n$, is defined as $P\big/\big(\sum_i \partial_{p_i} \cdot P\big)$. It is the image of composed maps: the Fourier transform $\mathcal{F}$, the inverse image π_* under the projection π from K^{n+m} to K^n defined by $\pi(p,t) = t$, and the inverse Fourier transform. Specifically we have, $\int P = \mathcal{F}^{-1}\pi_*\mathcal{F}(P)$. These maps preserve holonomy (see [2, Theorem 3.3.4] or [7, Theorem

18.2.2 and Section 20.3]), so that the integral of a holonomic $W_{p,t}$-module is a holonomic W_t-module. (See also [2, Theorem 3.1.8].)

The module T fits naturally in between an existing holonomy-preserving surjection from the W_t-module $\int M \otimes_{K[p,t]} N$ to the space $\langle M|N\rangle$. Factoring this map to pass through $T = M^\star \otimes_{W_p[t]} N$ yields

$$\int M \otimes_{K[p,t]} N \xrightarrow{\phi} M^\star \otimes_{W_p[t]} N \xrightarrow{\psi} \langle M|N\rangle\,, \tag{19}$$

where ψ surjectively maps $m \otimes n$ to $\langle m|n\rangle$, and ϕ is a natural W_t-linear surjection that we are about to define in the course of the next theorem. After proving that the first module in (19) is holonomic, the surjectivity of ϕ implies the holonomy of T.

Theorem 7. *Suppose that M and N are two holonomic $W_{p,t}$-modules, and define T as $M^\star \otimes_{W_p[t]} N$. Then, T is a holonomic W_t-module under the action of ∂_{t_i} given by*

$$\partial_{t_i} \cdot (m \otimes n) = (\partial_{t_i} \cdot m) \otimes n + m \otimes (\partial_{t_i} \cdot n).$$

Proof. First, we focus our attention on the module $\int M \otimes_{K[p,t]} N$ in (19). Consider the $W_{p,t}$-module $P := M \otimes_{K[p,t]} N$, with action of ∂_{p_i} defined by $\partial_{p_i} \cdot (m \otimes n) = (\partial_{p_i} \cdot m) \otimes n + m \otimes (\partial_{p_i} \cdot n)$, and action of ∂_{t_i} defined similarly. We can also write this as the inverse image $\iota^* (M \otimes_K N)$, where ι is the map from K^{m+n} to $K^{(n+m)+(n+m)}$ which sends (p, t) to (p, t, p, t). The advantage of the second presentation is that the holonomy of P is obtained from the holonomic closure under inverse image under embeddings (see [2, Theorem 3.2.3] or [7, Section 15.3 and Example 15.4.5]) and the holonomic closure under tensor product over K [7, Corollary 13.4.2]. Therefore, $\int P$ is also holonomic.

Next, we define a W_t-linear surjection to T. Define a map from $M \times N$ to T which sends (m, n) to $m \otimes n$. This map is $K[p, t]$-balanced, $K[p, t]$-bilinear, and surjective. By the universality of the tensor product, this induces a surjective map ϕ from $P = M \otimes_{K[p,t]} N$ to T. Observe that each derivation ∂_{p_i} maps P into the kernel of ϕ, as the following calculation indicates:

$$\begin{aligned}\phi\big(\partial_{p_i} \cdot (m \otimes n)\big) &= \phi\big((\partial_{p_i} \cdot m) \otimes n + m \otimes (\partial_{p_i} \cdot n)\big)\\ &= (\partial_{p_i} \cdot m) \otimes n + m \otimes (\partial_{p_i} \cdot n)\\ &= m \otimes (-\partial_{p_i} \cdot n) + m \otimes (\partial_{p_i} \cdot n) = 0.\end{aligned}$$

In other words, $\sum_i \partial_{p_i} \cdot P \subset \ker \phi$, and thus ϕ also induces a well-defined surjective map from $\int P$ to T. Any good filtration of $\int P$ will induce a good filtration for T (see [2, Proposition 1.11] or [7, Lemma 7.5.1]). Thus, T is finitely generated with dimension bounded by that of $\int P$. Therefore, T is holonomic. $\square$

7.4. *The quotient structure of S*

Subsequent developments to express S as a quotient involve modules over $W_{p,t}$ and ideals of $W_{p,t}$, rather than $W_{p,t}(t)$. We therefore introduce the annihilators $I_F = \operatorname{ann}_{W_{p,t}} F$ and $I_G = \operatorname{ann}_{W_{p,t}} G$, to be used in place of $\mathcal{I}_F = \operatorname{ann}_{W_{p,t}(t)} F$ and $\mathcal{I}_G = \operatorname{ann}_{W_{p,t}(t)} G$, respectively. Note that $I_F = \mathcal{I}_F \cap W_{p,t}$ and $\mathcal{I}_F = K(t) \otimes_{K[t]} I_F$, and similarly for G.

Finally, although adjunction has not been defined for ∂_t, we use the notation $W^\diamond_{p,t}$ to denote $W_{p,t}$ endowed with both a structure of W_t-module on the left and a structure of $W_p[t]$-module on the right.

Proposition 8. *The module $S = (W_{p,t} \cdot F)^\diamond \otimes_{W_p[t]} (W_{p,t} \cdot G)$ is isomorphic to*

$$(W^\diamond_{p,t} \otimes_{W_p[t]} W_{p,t})/(I^\diamond_F \otimes_{W_p[t]} W_{p,t} + W^\diamond_{p,t} \otimes_{W_p[t]} I_G).$$

Proof. The W_t-module $S = U^\diamond \otimes_{W_p[t]} V$ is also a $W^\diamond_{p,t} \otimes_{W_{p[t]}} W_{p,t}$-module. As such, it is generated by $F^\diamond \otimes G$. Consider the two exact sequences of respectively right and left $W_p[t]$-modules

$$\begin{array}{ccccccccc} 0 & \to & I^\diamond_F & \overset{\rho}{\to} & W^\diamond_{p,t} & \overset{\alpha}{\to} & U^\diamond & \to & 0, \\ 0 & \to & I_G & \overset{\sigma}{\to} & W_{p,t} & \overset{\beta}{\to} & V & \to & 0, \end{array}$$

where $\alpha(P) = F^\diamond \cdot P$, $\beta(Q) = Q \cdot G$, and ρ and σ are inclusions. (Here, F and $F^\diamond$ denote the same element of the set U, but we write $F^\diamond$ when viewed as an element of the right module $U^\diamond$, F when viewed as in the left module U.) We combine them to make a third exact sequence:

$$\begin{array}{ccccccc} \ker(\alpha\otimes\beta) & \to & W^\diamond_{p,t} \otimes_{W_p[t]} W_{p,t} & \overset{\alpha\otimes\beta}{\longrightarrow} & S & \to & 0, \\ & & P \otimes Q & \longmapsto & (F^\diamond \cdot P) \otimes (Q \cdot G), & & \end{array} \tag{20}$$

where, by Bourbaki [3, II.59, Proposition 6],

$$\ker(\alpha\otimes\beta) = \mathrm{im}(\rho \otimes 1_{W_{p,t}}) + \mathrm{im}(1_{W^\diamond_{p,t}} \otimes \sigma) = I^\diamond_F \otimes_{W_p[t]} W_{p,t} + W^\diamond_{p,t} \otimes_{W_p[t]} I_G$$

as $K[t]$-modules. We conclude that, as W_t-modules,

$$\begin{aligned} S &\simeq (W^\diamond_{p,t} \otimes_{W_p[t]} W_{p,t})/\ker(\alpha\otimes\beta) \\ &\simeq (W^\diamond_{p,t} \otimes_{W_p[t]} W_{p,t})/(I^\diamond_F \otimes_{W_p[t]} W_{p,t} + W^\diamond_{p,t} \otimes_{W_p[t]} I_G). \qquad \square \end{aligned}$$

To be more explicit, note that this isomorphism maps the class of $1 \otimes 1$ in the quotient to $F^\diamond \otimes G \in S$. Remark also that, as W_t-modules,

$$\begin{aligned} \ker(\alpha\otimes\beta) &= \left\{P \otimes Q \in W^\diamond_{p,t} \otimes W_{p,t} : (\alpha\otimes\beta)(P\otimes Q) = 0\right\} \\ &= \left\{P \otimes Q \in W^\diamond_{p,t} \otimes W_{p,t} : (F^\diamond \cdot P) \otimes (Q \cdot G) = 0\right\} \\ &= \left\{P \otimes Q \in W^\diamond_{p,t} \otimes W_{p,t} : (P \otimes Q) \cdot (F^\diamond \otimes G) = 0\right\} \\ &= \mathrm{ann}_{W^\diamond_{p,t} \otimes_{W_p[t]} W_{p,t}}(F^\diamond \otimes G), \end{aligned}$$

so that we also have

$$\mathrm{ann}_{W^\diamond_{p,t} \otimes_{W_p[t]} W_{p,t}}(F^\diamond \otimes G) = \ker(\alpha\otimes\beta) = I^\diamond_F \otimes_{W_p[t]} W_{p,t} + W^\diamond_{p,t} \otimes_{W_p[t]} I_G. \tag{21}$$

Proposition 9. *The W_t-module $S' = W_t \cdot (F^\diamond \otimes G)$ is a submodule of S, isomorphic to*

$$W'_t \Big/ \left((I^\diamond_F \otimes_{W_p[t]} W_{p,t} + W^\diamond_{p,t} \otimes_{W_p[t]} I_G) \cap W'_t\right),$$

where $W'_t \simeq W_t$ *is the smallest* K*-subalgebra of* $W^{\diamond}_{p,t} \otimes_{W_p[t]} W_{p,t}$ *generated by* $K[t]$, $1 \otimes \partial_{t_1} + \partial_{t_1} \otimes 1, \ldots, 1 \otimes \partial_{t_k} + \partial_{t_k} \otimes 1$. *In the simplified situation when* $I_F = \partial_t W_{p,t} + W_t J_F$ *for* $J_F = \operatorname{ann}_{W_p} F$, S' *is isomorphic to*

$$W_t \Big/ \big((W_t J_F^{\diamond} + I_G) \cap W_t\big).$$

We first prove this proposition, then in the next section we discuss how to connect the description of S' above directly to the algorithm and how to apply it to show that the algorithms terminate.

Proof. The annihilator of $F^{\diamond} \otimes G$ in $W'_t \cdot (F^{\diamond} \otimes G)$

$$\operatorname{ann}_{W'_t}(F^{\diamond} \otimes G) = \operatorname{ann}_{W^{\diamond}_{p,t} \otimes_{W_p[t]} W_{p,t}}(F^{\diamond} \otimes G) \cap W'_t.$$

In view of the action of W_t on S' through the isomorphism between W_t and W'_t, we thus have that S' is isomorphic to $W_t/\operatorname{ann}_{W_t}(F^{\diamond} \otimes G)$, itself isomorphic to

$$W'_t/\operatorname{ann}_{W'_t}(F^{\diamond} \otimes G) = W'_t/\big(\operatorname{ann}_{W^{\diamond}_{p,t} \otimes_{W_p[t]} W_{p,t}}(F^{\diamond} \otimes G) \cap W'_t\big).$$

Owing to (21), this proves the general quotient expression for S' in the proposition statement.

Now, to prove the formula in the simpler case, observe that when $I_F = \partial_t W_{p,t} + W_t J_F$,

$$\begin{aligned} I_F^{\diamond} \otimes_{W_{p[t]}} W_{p,t} &= \partial_t W^{\diamond}_{p,t} \otimes_{W_{p[t]}} W_{p,t} + W_t J_F^{\diamond} \otimes_{W_{p[t]}} W_{p,t} \\ &= \partial_t W_t \otimes_{K[t]} W_{p,t} + W_t \otimes_{K[t]} W_t J_F^{\diamond} \end{aligned}$$

while $W^{\diamond}_{p,t} \otimes_{W_p[t]} I_G = W_t \otimes_{K[t]} I_G$, whence the relation $\ker(\alpha \otimes \beta) = \partial_t W_t \otimes_{K[t]} W_{p,t} + W_t \otimes_{K[t]} (W_t J_F^{\diamond} + I_G)$. Since $W^{\diamond}_{p,t} \otimes_{W_p[t]} W_{p,t} = W_t \otimes_{K[t]} W_{p,t}$, we obtain

$$S \simeq W_{p,t}/(W_t J_F^{\diamond} + I_G),$$

as $(W_t \otimes_{K[t]} W_{p,t})/\ker(\alpha \otimes \beta) \simeq (K[t] \otimes_{K[t]} W_{p,t})/\big(K[t] \otimes_{K[t]} (W_t J_F^{\diamond} + I_G)\big) \simeq W_{p,t}/(W_t J_F^{\diamond} + I_G)$. Following these isomorphisms, W'_t can be identified as the copy of W_t included in $W_{p,t}$ in the last quotient above. Therefore, the submodule S' of S is isomorphic to the quotient announced in the proposition statement. $\square$

Corollary 10. *The ideal* $\operatorname{ann}_{W_t}(F^{\diamond} \otimes G)$ *is*

(1) *isomorphic to* $(I_F^{\diamond} \otimes_{W_p[t]} W_{p,t} + W^{\diamond}_{p,t} \otimes_{W_p[t]} I_G) \cap W'_t$ *as a* W_t*-module*;
(2) *a non-trivial ideal contained in* $\operatorname{ann}_{W_t} \langle F, G \rangle$ *and such that the quotient* $W_t/\operatorname{ann}_{W_t}(F^{\diamond} \otimes G) \simeq S'$ *is holonomic.*

Proof. From (21),

$$\begin{aligned} \operatorname{ann}_{W'_t}(F^{\diamond} \otimes G) &= \Big(\operatorname{ann}_{W^{\diamond}_{p,t} \otimes_{W_p[t]} W_{p,t}}(F^{\diamond} \otimes G)\Big) \cap W'_t \\ &= \Big(I_F^{\diamond} \otimes_{W_p[t]} W_{p,t} + W^{\diamond}_{p,t} \otimes_{W_p[t]} I_G\Big) \cap W'_t, \end{aligned} \tag{22}$$

and we have shown (1) in the corollary statement. The W_t-module $S' \simeq W_t/\operatorname{ann}_{W_t}(F^{\diamond} \otimes G)$ is a holonomic W_t-module, as it is a submodule of the holonomic W_t-module S. Now since

W_t is not holonomic, $\operatorname{ann}_{W_t}(F^\diamond \otimes G)$ must be non-trivial by a simple dimension argument. Finally, we recall that this non-trivial ideal is contained in $\operatorname{ann}_{W_t}\langle F, G\rangle$, since there is a surjection from S' to $W_t/\operatorname{ann}_{W_t}\langle F, G\rangle$ given by $\psi : (u \otimes v) \mapsto \langle u, v\rangle$. This proves (2) in the corollary statement. □

7.5. *Termination*

We now link the modules S and S' to the algorithms and prove their termination. The termination of Algorithm 3 is more technical to prove than that of Algorithm 1 since ∂_{t_i} can act separately on F and G. Thus, for ease of presentation, we consider Algorithms 1 and 3 in turn, to show that they eventually generate a Gröbner basis for $\operatorname{ann}_{W_t(t)}(F^\diamond \otimes G)$.

7.5.1. *Termination of Algorithm 1*

The basic idea of Algorithm 1 is to compute filtrations of $\mathcal{I}_F$ and $\mathcal{I}_G$ independently and incrementally and to recombine them at each step. The algorithm terminates when condition (3e) in the algorithm description is satisfied. We show that the algorithm will satisfy this condition by eventually producing a Gröbner basis for $\operatorname{ann}_{W_t(t)}(F^\diamond \otimes G)$. This subideal describes $F^\diamond \otimes G$ and $\langle F, G\rangle$ as D-finite.

Proof (*Theorem 6, Algorithm 1*). Algorithm 1 places a constraint on F that allows us to take advantage of the simpler W_t-structure of $U = W_{p,t} \cdot F$: since each $\partial_{t_i} \cdot F$ is 0, we have $U = K[t] \otimes_K (W_p \cdot F)$ and $I_F = \partial_t W_{p,t} + W_t J_F$. Taking the intersection with W_t' is then far more transparent: from the previous section, we obtain the following simplification of Eq. (22):

$$\operatorname{ann}_{W_t}(F^\diamond \otimes G) = \left(J_F^\diamond W_t + I_G\right) \cap W_t. \tag{23}$$

Considering the monoid of monomials generated by $p, \partial_p, \partial_t$, ordered by the monomial order $\preccurlyeq$ specified by the algorithm, we denote by $\mathcal{V}_\beta$ the filtration $\bigoplus_{\gamma \preccurlyeq \beta} K(t)\gamma$.

Assume that Algorithm 1 fails to terminate on some input F and G. For any β, Algorithm 1 thus eventually reaches a value for the main loop index α such that all the monomials that have been considered in the algorithm span a vector space containing $\mathcal{V}_\beta$. After Step (3d) in the main loop for this value α of the loop index, B generates a vector space containing

$$L_\beta := \left(J_F^\diamond W_t(t) \cap \mathcal{V}_\beta\right) + \left(\mathcal{I}_G \cap \mathcal{V}_\beta\right).$$

By our choice of elimination term order, $B \cap W_t(t)$ consists of generators of a vector space which contains the intersection $L_\beta \cap W_t(t)$.

Next, for each γ, $\left(J_F^\diamond W_t(t) + \mathcal{I}_G\right) \cap \mathcal{V}_\gamma$ is a subspace of L_β for some β. Indeed, since $\mathcal{V}_\gamma$ is finite dimensional, so is the intersection under consideration. Let us introduce a basis $b_1, \dots, b_d$ of it. Each b_i can be written in the form $f_i + g_i$ for $f_i \in \mathcal{I}_F^\diamond = J_F^\diamond W_t(t)$ and $g_i \in \mathcal{I}_G$, so that, provided $\beta = \max\{\max_i \deg f_i, \max_i \deg g_i\}$, the intersection

$$\left(J_F^\diamond W_t(t) + \mathcal{I}_G\right) \cap \mathcal{V}_\gamma = \bigoplus_{i=1}^{d} K(t)(f_i + g_i)$$

is a subspace of

$$\sum_{i=1}^{d} K(t) f_i + \sum_{i=1}^{d} K(t) g_i \subset \big(W_t(t) J_F^{\diamond} \cap \mathcal{V}_\beta\big) + \big(\mathcal{I}_G \cap \mathcal{V}_\beta\big) = L_\beta.$$

Since $\operatorname{ann}_{W_t(t)}(F^\diamond \otimes G)$ is finitely generated by noetherianity of $W_t(t)$, we can choose a finite set of generators for it, and set γ to their maximal leading monomial. Consequently, the chosen generators are in

$$\operatorname{ann}_{W_t(t)}(F^\diamond \otimes G) \cap \mathcal{V}_\gamma = \big(W_t(t) J_F^{\diamond} + \mathcal{I}_G\big) \cap W_t(t) \cap \mathcal{V}_\gamma.$$

By the reasoning above, the latter is a subspace of L_β for some β, and when the loop index reaches a sufficiently high α, $\operatorname{ann}_{W_t(t)}(F^\diamond \otimes G)$ is a subideal of the ideal generated in $W_t(t)$ by $B \cap W_t(t)$. Since, by Corollary 10, $W_t/\operatorname{ann}_{W_t}(F^\diamond \otimes G)$ is a holonomic module, $\operatorname{ann}_{W_t(t)}(F^\diamond \otimes G)$ is of dimension 0, and condition (3e) is satisfied. The algorithm terminates, a contradiction to our assumption. □

A limitation of the algorithm is that we cannot predict in advance how many monomials must be tested, and hence cannot estimate the running time.

7.5.2. Termination of Algorithm 3

The termination of Algorithm 3 can be proved similarly, but we must use greater care when treating the ∂_{t_i}.

Proof (*Theorem 6, Algorithm 3*). Since there is no adjoint action for ∂_{t_i}, we consider occurrences of ∂_{t_i} in the left argument of the scalar product differently from those on the right side. This is modelled in S by tensoring over $W_p[t]$, where ∂_t is absent and thus, $\partial_{t_i} \otimes 1$ differs from $1 \otimes \partial_{t_i}$. Both still obey the same commutation law with t_i as ∂_{t_i}. Denote the former by ∂_{ℓ_i} and the latter by ∂_{r_i}.

Having distinguished these two cases, we rewrite several of the important elements from the previous proof using this new notation. For example,

$$\begin{aligned} W_{p,t}^{\diamond} \otimes_{W_p[t]} W_{p,t} = K\big\langle & p, t, \partial_p, \partial_\ell, \partial_r; [\partial_{p_i}, p_j] = [\partial_{\ell_i}, t_j] = [\partial_{r_i}, t_j] = \delta_{i,j}, \\ & [p_i, p_j] = [p_i, t_j] = [t_i, t_j] = [\partial_{\ell_i}, p_j] = [\partial_{r_i}, p_j] \\ & = [\partial_{p_i}, t_j] = 0\big\rangle, \end{aligned}$$

and its subalgebra W_t' is generated by $K[t]$, $\partial_{\ell_1} + \partial_{r_1}, \ldots, \partial_{\ell_k} + \partial_{r_k}$. We can also rewrite $I_F^{\diamond} \otimes_{W_p[t]} W_{p,t} + W_{p,t}^{\diamond} \otimes_{W_p[t]} I_G$ in the form $I_F^{\diamond}\big|_{\partial_t = \partial_\ell} K[\partial_r] + K[\partial_\ell] I_G\big|_{\partial_t = \partial_r}$. Algorithm 3 actually computes with coefficients that are rational functions in t, and so with elements of $\mathcal{I}_F^{\diamond}\big|_{\partial_t = \partial_\ell} K[\partial_r] + K[\partial_\ell] \mathcal{I}_G\big|_{\partial_t = \partial_r}$.

In order to endow $W_{p,t}^{\diamond} \otimes_{W_p[t]} W_{p,t}$ with a filtration, let us extend the ordering $\preccurlyeq$ to monomials in $p, \partial_p, \partial_\ell, \partial_r$ by considering any ordering which, after setting $\partial_\ell = \partial_t$, $\partial_r = 1$ or $\partial_r = \partial_t$, $\partial_\ell = 1$, respectively, induces the ordering $\preccurlyeq$. We denote the extended ordering by $\preccurlyeq$ as well. Then, we let $\mathcal{U}_\beta$ denote the filtration $\bigoplus_{\gamma \preccurlyeq \beta} K(t)\beta$ for β, γ ranging over the monomials in the variables $p, \partial_p, \partial_r, \partial_\ell$. Turning our attention to $W_t'(t)$, let $\mathcal{V}_\beta'$ be the image of the $\mathcal{V}_\beta$ of the previous section, under the same transformation which takes $W_t(t)$ to $W_t'(t)$, that is,

$$\mathcal{V}_\beta' = \bigoplus_{p^a \partial_p^b \partial_t^c \preccurlyeq \beta} K(t) p^a \partial_p^b \left(\partial_\ell + \partial_r\right)^c .$$

For each β, there is β' such that $\mathcal{V}_\beta' \subset \mathcal{U}_{\beta'}$.

Assume that Algorithm 3 fails to terminate on some input F and G. Since the main loop enumerates all monomials in $p, \partial_p, \partial_\ell, \partial_r$ in some order, for any β there exists a value of the index loop α such that when the loop reaches it, all monomials that have been enumerated span a vector space containing $\mathcal{U}_\beta$. After the algorithm has introduced (variants of) α_F and α_G at Step (3e) for this value of α, let us call V_α the vector space generated by the set B. Setting $\partial_\ell = \partial_t - \partial_r$ maps V_α to a vector space which contains

$$H_\beta := \left(\mathcal{I}_F^\diamond\big|_{\partial_t=\partial_\ell} K[\partial_r]\right) \cap \mathcal{U}_\beta + \left(K[\partial_\ell]\mathcal{I}_G\big|_{\partial_t=\partial_r}\right) \cap \mathcal{U}_\beta.$$

We use this fact to conclude termination.

At this point we show that for each γ, the vector space $\mathcal{X} \cap \mathcal{V}_\gamma'$ where

$$\mathcal{X} = \mathcal{I}_F^\diamond \otimes_{W_p(t)} W_{p,t}(t) + W_{p,t}(t)^\diamond \otimes_{W_p(t)} \mathcal{I}_G$$

is a subspace of H_β for some β. Indeed, choose γ' such that $\mathcal{V}_\gamma' \subset \mathcal{U}_{\gamma'}$, so that $\mathcal{X} \cap \mathcal{V}_\gamma' \subset \mathcal{X} \cap \mathcal{U}_{\gamma'}$. The latter intersection is finite-dimensional, since $\mathcal{U}_{\gamma'}$ is so. Suppose it has for basis $b_1, \dots, b_d$, with each b_i of the form $b_i = f_i r_i + l_i g_i$, where $f_i \in \mathcal{I}_F^\diamond\big|_{\partial_t=\partial_\ell}$, $g_i \in \mathcal{I}_G\big|_{\partial_t=\partial_r}$, $r_i \in K[\partial_r]$, and $l_i \in K[\partial_\ell]$, and set $\beta = \max\{\max_i \deg f_i r_i, \max_i \deg l_i g_i\}$, where here deg extracts the leading monomial. Then,

$$\mathcal{X} \cap \mathcal{V}_\beta' \subset \bigoplus_{i=1}^{d} K(t)(f_i r_i + l_i g_i) \subset \sum_{i=1}^{d} K(t) f_i r_i + \sum_{i=1}^{d} K(t) l_i g_i \subset H_\beta.$$

By noetherianity, we can choose a finite set of generators for $\operatorname{ann}_{W_t(t)}(F^\diamond \otimes G)$, and set γ to their maximal leading monomial. The generators are thus elements of $\operatorname{ann}_{W_t(t)}(F^\diamond \otimes G) \cap \mathcal{V}_\gamma$, which is isomorphic to $\operatorname{ann}_{W_t'(t)}(F^\diamond \otimes G) \cap \mathcal{V}_\gamma'$. By (22) the latter is also $\mathcal{X} \cap \mathcal{V}_\gamma'$, and, as explained above, there is β such that this is a subspace of H_β.

By our earlier loop invariant, the same generators, after setting $\partial_\ell = \partial_t - \partial_r$, are contained in the space spanned by B when the loop index reaches a sufficiently high α'. Thus, it suffices to run the algorithm until this α and generators of $\operatorname{ann}_{W_t}(F^\diamond \otimes G)$ will be contained in B. At this point the termination conditions are satisfied, and the algorithm terminates. $\square$

8. Asymptotic estimates

We now illustrate how the differential equations computed by our algorithms may be exploited in order to derive asymptotic estimates of combinatorial quantities.

8.1. Outline of the method

A very general principle in asymptotic analysis is that the asymptotic behaviour of a sequence is governed by the local behavior of its generating series at its singularity of smallest modulus, see for instance [28, Section 10]. Our approach is thus based on applying the classical analysis of linear differential equations as presented in textbooks such as [17,43] in order to derive asymptotic estimates for the coefficients. Moreover, large parts of this analysis can be automated thanks to the algorithms described in [23,39,42], many of which have been implemented in computer algebra systems. [6] An alternative approach based on Birkhoff's work can be found in [44].

In the special case of solutions of linear differential equations, the possible location of singularities is restricted to the roots of the coefficient of the highest derivative. Then, the analysis depends on the nature of the singularity. The classical theory distinguishes two kinds of singular points: regular singular points, where the solutions have an algebraic–logarithmic behavior; and irregular singular points where the solutions have an essential singularity of the type exponential of a rational power. Accordingly, the asymptotic behavior of the coefficients is deduced either by singularity analysis [8,18], or by the saddle-point method [16,45]; both approaches are implemented in the `algolib` library.

This asymptotic analysis of D-finite generating series extends to the divergent case. Indeed, the coefficients u_n of a divergent D-finite series grow at most like a power of $n!$ with a rational exponent p/q which can be computed (see example below). Then one constructs an auxiliary differential equation satisfied by the convergent generating series of $u_n/(n(n-q)(n-2q)\cdots r)^p$ (where r denotes the remainder of the division of n by q), to which the previous method applies. This construction is achieved thanks to the closure properties of D-finite series, by multiplying u_n with the solution of the recurrence $(n+q)^p v_{n+q} = v_n$, which, up to a constant, grows like $n!^{p/q} n^{p(q-1)/2q}$. This operation is implemented in the `gfun` package.

8.2. k-Uniform Young tableaux

We now illustrate this method in the special case of the k-uniform Young tableaux of Section 5. We treat in detail the case $k = 3$; other cases are similar. To the best of our knowledge, these asymptotic estimates are new.

We start from the differential equation for $k = 3$ to be found in Table 2. This is a second-order differential equation and its leading coefficient vanishes at the origin. This indicates a possible singularity of $Y_3(t)$ at the origin, which would be reflected by the divergence of this series. Indeed, from this differential equation, a linear recurrence is readily computed for the coefficients $u_n := y_n^{[3]}$:

$$\begin{aligned}&u_n + u_{n+1} - (3n+12)u_{n+2} - 4u_{n+3} + (6n+35)u_{n+4} - 15u_{n+5}\\&\quad +(9n^2+93n+242)u_{n+6} + (18n+126)u_{n+7} - (9n^2+159n+698)u_{n+8}\\&\quad +(9n^2+147n+606)u_{n+9} - (18n^2+366n+1884)u_{n+10}\\&\quad -(48n+552)u_{n+11} + (24n+288)u_{n+12} = 0.\end{aligned}$$

[6] In Maple, this functionality is provided by `DEtools[formal_sol]`.

8.2.1. Divergence

From this recurrence it is easy to compute a couple hundred coefficients and observe their rapid growth. Simple experiments indicate that the growth of these coefficients is of order $\sqrt{n!}$. That this growth is the exact exponent of $n!$ in the behavior follows upon considering the degrees of the coefficients in the recurrence: the terms of order 12 and 11 have coefficients of degree 1, while the term of order 10 has a coefficient of degree 2 (the maximal degree). Thus, up to first order, the behavior is dictated by

$$24nu_{n+12} = 18n^2u_{n+10},$$

which leads to a growth of order $(\frac{3}{4})^{n/2}n!^{1/2}$. In order to derive a more precise estimate, we compute a linear differential equation satisfied by the *convergent* generating function of $y_n^{[3]}v_n$ where v_n satisfies $v_{n+2} = v_n/(n+2)$. This differential equation is obtained by first computing a linear recurrence for $y_n^{[3]}v_n$, which exists thanks to the closure properties of linear recurrent sequences. This closure operation produces a linear recurrence of order 24 with coefficients of degree 29. From there we obtain a linear differential equation of order 29 with coefficients of degree 37, which we now analyze.

8.2.2. Singular behavior

The leading coefficient of the previous equation is $t^{27}(3t^2-4)$, up to a constant factor. This reveals a dominant singularity at $\rho = 2/\sqrt{3}$, thus confirming the growth order $(3/4)^{n/2}$ expected from the previous stage.[7] The next step consists in analyzing the behavior of our convergent generating series in the neighborhood of ρ. A local analysis of the differential equation reveals that all solutions of this equation of order 29 behave like

$$g(u) + \lambda\frac{\exp\left(\frac{3}{4u}\right)}{\sqrt{u}}\left(1 - \frac{145}{144}u - \frac{8591}{41472}u^2 + O\left(u^3\right)\right), \qquad 1 - z/\rho = u \to 0,$$

where g is an analytic function at 0, and λ is a constant depending on the solution.

8.2.3. Asymptotic estimate

This behavior is typical of an irregular singular point and can thus be dealt with using the saddle-point method. Putting everything together, we finally obtain

$$y_n^{[3]} = C_3n!^{1/2}\left(\frac{\sqrt{3}}{2}\right)^n\frac{\exp\sqrt{3n}}{n^{3/4}}(1 + O(1/n)),$$

for some constant C_3, and where the O-term hides the beginning of an expansion in descending powers of n that could be computed with the same method.

The constant C_3 can then be approximated numerically by using Romberg's acceleration method, adapted to powers of $n^{-1/2}$, and we get

$$C_3 \approx 0.377200.$$

[7] We could also have incorporated this factor in the recurrence for v_n.

Table 4
Asymptotic number of k-uniform Young tableaux

1	$C_1 \frac{\exp\sqrt{n}}{\sqrt{n!}\,n^{1/4}}$	$C_1 \approx 0.347829$
2	$C_2 \frac{\exp\sqrt{2n}}{\sqrt{n}}$	$C_2 \approx 0.282094$
3	$C_3\sqrt{n!}\left(\frac{\sqrt{3}}{2}\right)^n \frac{\exp\sqrt{3n}}{n^{3/4}}$	$C_3 \approx 0.377200$
4	$C_4 n!\left(\frac{2}{3}\right)^n \frac{\exp 2\sqrt{n}}{n}$	$C_4 \approx 0.831565$

8.2.4. *Other values of k*

The computation of the asymptotic behavior of $y_n^{[k]}$ for other values of k is completely similar, provided one has computed the differential equation. We summarize our results in Table 4. This serves to illustrate a typical use of our techniques in experimental mathematics to obtain conjectures such as the following.

Conjecture 11. *The number $y_n^{[k]}$ of k-uniform Young tableaux of size n behaves asymptotically according to*

$$y_n^{[k]} \sim \frac{1}{\sqrt{2}}\left(\frac{e^{k-2}}{2\pi}\right)^{k/4} n!^{k/2-1}\left(\frac{k^{k/2}}{k!}\right)^n \frac{\exp(\sqrt{kn})}{n^{k/4}}, \qquad n \to \infty.$$

This conjecture is proved for $k = 1$ and 2: the constant is obtained from a closed form solution of the differential equation. For $k = 3$ and 4, only the value of the constant is conjectural. The proof of the general case of the conjecture requires techniques such as those of [11,25], which fall outside of the scope of this article.

8.3. *Conclusion*

The main advantages of our method are its general applicability, its ability to produce full asymptotic expansions up to *one* constant factor, the availability of computer algebra programs that automate many of its steps. The price to pay for this generality is that the method can only produce numerical estimates for the constant factor. In some special cases, specific approaches often exist that provide this constant term.

9. Conclusions and directions for future work

9.1. *Applying the method to other scalar products*

Let us note that the method of this article can be applied in the case of other scalar products, provided that the corresponding adjunction $\diamond$ (no longer denoting the symmetric adjunction) is a linear involution that preserves the total degree (in p, ∂_p) of the differential operators. In effect, one should simply set $M = (U^{\diamond})^{\star}$ and $N = V$ to obtain a suitable analogue to (18) and prove the holonomy, thus D-finiteness, of the scalar product: M is holonomic if and only if U is. Since the statement and proof of Algorithms 1 and 3 do not make use of any other special property of $\diamond$ than being a degree-preserving involution, correctness of the algorithms can then be established along the same lines as for the case of the scalar product of symmetric functions.

We use this idea in the next two sections by introducing various scalar products given by an adjunction relation involving a formal parameter.

9.2. Calculating the Kronecker product of symmetric functions

Another symmetric function operation, closely related to the scalar product, is the Kronecker product, also known as the tensor product. One can define it on the power basis as $p_\lambda * p_\mu = \langle p_\lambda, p_\mu \rangle p_\lambda$. Gessel showed in [9] that given two D-finite symmetric series F and G, the Kronecker product $F * G$ is also a D-finite symmetric series. Algorithm 1 can be used to make this fact effective via the following observation:

$$p_\lambda * p_\mu = \left\langle p_\lambda t^\lambda, p_\mu \right\rangle \big|_{t_i = p_i}.$$

More precisely, we rewrite a Kronecker product as a scalar product by multiplying each p_i in F by t_i. In the system which results we make the substitution $t_i = p_i$ and $\partial_{t_i} = \partial_{p_i}$.

We formalize this in the following algorithm, which merely calls Algorithm 1 on modified input systems.

Algorithm 4 (*Kronecker product*).

Input: *Symmetric functions $F \in K[[p]]$ and $G \in K[[p]]$, both D-finite in p, each given by a D-finite description in W_p.*

Output: *A D-finite description of $F * G$* in W_t.

(1) *Call $\mathcal{G}$ the system defining G and set $\mathcal{G}' = \{t_1\partial_{t_1} - p_1\partial_{p_1}, \dots, t_n\partial_{t_n} - p_n\partial_{p_n}\}$.*
  (a) *For each element in $\mathcal{G}$, replace p_i with $t_i p_i$, ∂_{p_i} with $t_i^{-1}\partial_{p_i}$ and add to $\mathcal{G}'$;*
  (b) *For each element in $\mathcal{G}$, replace p_i with $t_i p_i$, ∂_{p_i} with $p_i^{-1}\partial_{t_i}$, clear denominators, and add to $\mathcal{G}'$;*

(2) *Follow the steps of Algorithm* 1 *on the input system for F and the modified system $\mathcal{G}'$ for G.*

(3) *In the output of Algorithm* 1 *make the substitution $t_i = p_i$ and $\partial_{t_i} = \partial_{p_i}$ and return this value.*

Many interesting problems which use this operation require an infinite number of p_n, and are thus at first glance seemingly unsuitable for direct application of our algorithms. However, applying our algorithms for several truncations of a combinatorial problem can serve as a means to generate information upon which reasonable conjectures can be formulated. For example, Eq. (25) below was initially conjectured after a clear pattern emerged from a sequence of appeals to Algorithm 4. For each of these, we render the problem applicable by setting most p_n's to 0. In some cases, notably symmetric series arising from plethysms, there is sufficient symmetry and structure which can be exploited to verify these guesses by applying one of Algorithm 4 to well chosen subproblems. That is, in certain cases, such as the example that follows, the Kronecker product of two functions each with an infinite number of p_n variables can be reduced to a finite number of symbolic calculations.

For example, if two symmetric series F and G can be expressed, respectively, in the form

$$F(p_1, p_2, \ldots) = \prod_{n \geqslant 1} f_n(p_n) \qquad \text{and} \qquad G(p_1, p_2, \ldots) = \prod_{n \geqslant 1} g_n(p_n),$$

for functions f_n, g_n, then one can easily deduce that

$$F * G = \prod_{n \geqslant 1} f_n(p_n) * g_n(p_n). \tag{24}$$

Remark that series which arise as plethyms of the form $h[u]$ or $e[u]$, where u can be written as a sum $\sum_n u_n(p_n)$, for some functions u_n, are precisely of this form. For example, we can use this fact to compute the Kronecker product of the sum of all Schur functions

$$F(p_1, p_2, \ldots) = \sum_\lambda s_\lambda = h[p_1 + 1/2p_1^2 - 1/2p_2] = \exp\left(\sum_i \frac{p_i^2}{2i} + \frac{p_{2i-1}}{2i-1}\right),$$

and itself. Due to the patterns present, we can reduce the calculation of the entire product to two symbolic calculations. More precisely, in order to determine a system of differential equations satisfied by $G = F * F$ we consider only the even and odd cases, and set

$$f_{2n} = \exp(p_{2n}^2/4n) \quad \text{and} \quad f_{2n-1} = \exp((p_{2n-1}^2/2 + p_{2n-1})/(2n-1)).$$

All of the functions $g_{2n} = f_{2n} * f_{2n}$ are obtained from a single computation by our Algorithm 4, adapted to handle a formal parameter. This modification is of the same nature of that described in Section 9.1. Here we introduce the scalar product given by the adjunction formula $p^\diamond = n\partial$ for a *formal parameter n* from the field K. Thus computing $\exp(p^2/4n) * \exp(p^2/4n)$ with this variant algorithm results in a first-order operator in p and ∂, which, once interpreted back in terms of p_n becomes

$$(1 - p_n^2)\frac{\partial g_n(p_n)}{\partial p_n} + p_n g_n(p_n) = 0, \qquad \text{for even } n.$$

A second calculation for $g_{2n-1} = f_{2n-1} * f_{2n-1}$ results in

$$n(1 + p_n)(1 - p_n)^2 \frac{\partial g_n(p_n)}{\partial p_n} - \left(1 + (n+1)p_n - np_n^2\right) g_n(p_n) = 0, \quad \text{for odd } n.$$

These linear equations are satisfied respectively by the functions

$$g_{2n} = \left(1 - p_{2n}^2\right)^{-1/2} \quad \text{and}$$
$$g_{2n-1} = \exp\left(\frac{p_{2n-1}}{(2n-1)(1 - p_{2n-1})}\right)\left(1 - p_{2n-1}^2\right)^{-1/2}.$$

Applying Eq. (24) above, we get the following result.

Proposition 12. *The Kronecker product of the sum of the Schur functions with itself is*

$$\left(\sum_\lambda s_\lambda\right) * \left(\sum_\lambda s_\lambda\right) = \exp\left(\sum_{n\geqslant 1} \frac{p_{2n-1}}{(2n-1)(1-p_{2n-1})}\right)\left(\prod_{n\geqslant 1}\left(1-p_n^2\right)\right)^{-1/2}. \tag{25}$$

9.3. A q-analogue

A q-calculus parameter can be incorporated in symmetric functions in several ways.

Apart from the scalar product defined by (1), several other ones are of interest in relation to symmetric functions, notably the following two, which lead to the definitions of Hall and Macdonald polynomials, respectively:

$$\langle p_\mu, p_\lambda\rangle = z_\lambda \delta_{\mu,\lambda}\prod_{i=1}^{l(\lambda)}(1-t^{\lambda_i}) \quad \text{and} \quad \langle p_\mu, p_\lambda\rangle = z_\lambda \delta_{\mu,\lambda}\prod_{i=1}^{l(\lambda)}\frac{(1-t^{\lambda_i})}{1-q^{\lambda_i}},$$

where $\ell(\lambda)$ is the length k of a partition $\lambda = (\lambda_1, \ldots, \lambda_k)$. The same approach as in this article works in this setting and our Maple code has been adapted very easily.[8]

As a related problem, the ring homomorphism $\theta_q : \Lambda \to K[q][[t]]$ defined as

$$\theta_q\big(f(x_1, x_2, \ldots)\big) = f\big((1-q)t, (1-q)qt, (1-q)q^2t, \ldots\big)$$

is useful for studying partitions and for counting permutations [34]. This is one possibility for a q-analogue to the map θ from Theorem 1 (named exponential specialization in [34]), since $\lim_{q\to 1}\theta_q(F) = \theta(F)(x)$. An algorithm to compute θ_q, possibly mapping differential equation to D_q equation should be of interest.

9.4. Other conditions for D-finite closure

Remark that Theorem 3 requires that g be a function of only a finite number of p_n. The necessity of this condition is evident in the following example. Find a sequence c_n such that $\sum c_n t^n$ is not D-finite. However, according to the given definition of D-finite symmetric series, $\sum_n c_n p_n$ is D-finite, as is $\sum_n p_n t^n/n$. The series $\left\langle \sum_n c_n p_n, \sum_n p_n t^n/n\right\rangle = \sum_n c_n t^n$ is not D-finite by construction.

On the other hand, the condition is not essential. We have that $\langle H(1), H(t)\rangle = \frac{1}{1-t}$, which is D-finite despite H being a function of *all* p_n. Perhaps a closer investigation on the level of modules could reveal a refined condition.

[8] This variant is also available at `http://algo.inria.fr/mishna`.

Acknowledgments

The authors wish to thank François Bergeron for promoting D-finite symmetric functions as an interesting area of study. M.M. also extends gratitude towards NSERC for funding, and to Projet Algo, Inria, for their generous invitations during which much of the work was completed. Finally, we thank the anonymous referees who read the work carefully and offered many useful suggestion to improve the clarity.

Appendix A. 4-Uniform Young tableaux

The differential equation satisfied by $Y_4(t)$ is

$$64t^4(t-2)^2(t+1)^4\alpha(t)Y_4^{(3)}(t) - 16t^2(t-2)(t+1)^2\beta(t)Y_4^{(2)}(t)$$
$$+4\gamma(t)Y_4'(t) - \delta(t)Y_4(t) = 0$$

where $\alpha(t)$, $\beta(t)$, $\gamma(t)$, $\delta(t)$ are irreducible polynomials given by

$$\begin{aligned}\alpha(t) = {} & t^{14} - t^{13} - 5t^{12} - 7t^{11} + 6t^{10} + 35t^9 + 39t^7 - 50t^6 - 162t^5 - 92t^4\\ & +228t^3 + 424t^2 + 248t + 48,\\ \beta(t) = {} & t^{29} - 3t^{28} - 16t^{27} + 24t^{26} + 147t^{25} + 14t^{24} - 770t^{23} - 666t^{22} + 1416t^{21}\\ & +3567t^{20} - 916t^{19} - 16598t^{18} + 17766t^{17} + 40678t^{16} - 102556t^{15}\\ & -53272t^{14} + 390656t^{13} + 364080t^{12} - 707936t^{11} - 1406336t^{10}\\ & -552544t^9 + 1397664t^8 + 2020864t^7 + 176256t^6 - 916864t^5\\ & +304896t^4 + 1283328t^3 + 877056t^2 + 253440t + 27648,\\ \gamma(t) = {} & t^{28} - t^{27} - 14t^{26} - 20t^{25} + 111t^{24} + 278t^{23} - 196t^{22} - 1216t^{21}\\ & -1384t^{20} + 2765t^{19} + 3170t^{18} - 3400t^{17} + 12140t^{16} + 15588t^{15}\\ & -70280t^{14} - 108946t^{13} + 121796t^{12} + 349056t^{11}\\ & +116992t^{10} - 481704t^9 - 706320t^8 + 3040t^7 + 581184t^6 + 158688t^5\\ & -297408t^4 - 173952t^3 + 22272t^2 + 35712t + 6912,\\ \delta(t) = {} & 2t^{21} - 3t^{20} - 17t^{19} - 2t^{18} + 74t^{17} + 105t^{16} - 108t^{15} - 172t^{14} - 252t^{13}\\ & +432t^{12} - 667t^{11} + 1500t^{10} + 7336t^9 - 3772t^8 - 23056t^7 - 20584t^6\\ & +15504t^5 + 38160t^4 + 17904t^3 - 4512t^2 - 5568t - 1152.\end{aligned}$$

Appendix B. Sample maple session for 3-regular graph computation

The following Maple session indicates the user-level routines required to program Algorithm 2. It requires the library `algolib`, which is available at `http:algo.inria.fr/packages/`.

```
# Load the packages.
with(Ore_algebra): with(Mgfun): with (Groebner):
# Determine the DE satisfied by the generating function
# for 3-regular graphs.
k:=3: Fp:= exp(1/2*p1ˆ2-1/4*p2ˆ2-1/2*p2+p3ˆ2/6):
Gp:=exp(1/6*t3*p1ˆ3+1/2*t2*p1ˆ2+t1*p1+1/2*t3*p2*p1
    +1/2*t2*p2+1/3*t3*p3):
# Define the variables.
vars:= seq(p||i, i=1..k):  dvars:= seq(d||i, i=1..k):
tvars:= seq(t||i, i=1..k): dtvars:= seq(dt||i, i=1..k):

# Define the algebra.
A:= diff_algebra(seq([dvars[i], vars[i]], i=1..k),
seq([dtvars[i], tvars[i]], i=1..k), polynom={vars}):
At:= diff_algebra(seq([dtvars[i], tvars[i]], i=1..k)):

# Define the monomial orders.
T[g]:=termorder(A, lexdeg([dvars, vars],[dtvars])):
T[f]:=termorder(A,tdeg(vars, dvars, dtvars)):

# Define the systems.
sys[g]:=dfinite_expr_to_sys(Gp, F(seq(p||i::diff, i=1..k),
        seq(t||i::diff, i=1..k))):
newsys[g]:=subs(
    [seq(diff(F(vars,tvars),vars[i])=dvars[i],i=1..k),
     seq(diff(F(vars, tvars), tvars[i])=dtvars[i], i=1..k),
     F(vars,tvars)=1], sys[g]):

# Find the Groebner basis for G.
GB[g]:=gbasis(newsys[g],T[g]);

# Do the same for F.
sys[f]:=dfinite_expr_to_sys(Fp, F(seq(p||i::diff, i=1..k))):
newsys[f]:=subs([seq(diff(F(vars),vars[i])=dvars[i],i=1..k),
F(vars)=1],sys[f]);
GB[f]:=gbasis(newsys[f],T[f]);

# Define the adjoint and reduction procedures.
star:= x->subs(
   [seq(d||i=1/i*p||i, i=1..k),seq(p||i=d||i*i, i=1..k)],x):
rdc[f]:=x->star(star(x)-map(normalf, star(x), GB[f], T[f]));
rdc[g] := x->normalf(x, GB[g], T[g]);
```

```
# Reduce the Groebner basis of F.
for pol in GB[f] do m[pol]:=rdc[g](pol) end do:

# Small optimization: we will always try to reduce
# with respect to a linear term when possible.
lpol:=[seq(m[i],i=subsop(1=NULL,GB[f])),m[GB[f][1]]]:

for indelim from k-1 by -1 to 1 do
    # eliminate dt.indelim
    for j from 2 to nops(lpol) do
       newpol[j]:=skew_elim(lpol[j],lpol[1],dt||indelim,At)
    end do;
    # set t.indelim = 0
    lpol:=map(primpart,subs(t||indelim=0,
        [seq(newpol[j],j=2..nops(lpol))]),[dtvars])
end do:

# The only term left is the correct one.
ode:=op(lpol):
# Convert to recurrence.
REC:=diffeqtorec(
    {applyopr(ode, F(t||k), At), F(0)=1}, F(t||k), a(n)):
# Calculate some terms.
GRAPH:=rectoproc(REC, a(n),list)(20):
[seq(GRAPH(10)[i]*(i-1)!,i=1..20)];


 [1,0,0,0,1,0,70,0,19355,0,11180820,0,11555272575,0,
   19506631814670,0,50262958713792825,0,
   187747837889699887800,0]
```

References

[1] F. Bergeron, G. Labelle, P. Leroux, Combinatorial Species and Tree-like Structures, Cambridge University Press, Cambridge, 1998.

[2] A. Borel, P.-P. Grivel, B. Kaup, A. Haefliger, B. Malgrange, F. Ehlers, Algebraic *D*-modules, Academic Press Inc., Boston, MA, 1987.

[3] N. Bourbaki, Éléments de mathématique. Algèbre, Hermann, Paris, 1970 (Chapitres 1 à 3).

[4] F. Chyzak, Fonctions holonomes en calcul formel, Thèse universitaire, École polytechnique, 1998; INRIA, TU 0531. 227pp.

[5] F. Chyzak, B. Salvy, Non-commutative elimination in Ore algebras proves multivariate identities, J. Symbolic Comput. 26 (2) (1998) 187–227.

[6] L. Comtet, The art of finite and infinite expansions, Advanced Combinatorics, enlarged ed., D. Reidel Publishing Co., Dordrecht, 1974.

[7] S.C. Coutinho, A Primer of Algebraic *D*-modules, Cambridge University Press, Cambridge, 1995.

[8] P. Flajolet, A.M. Odlyzko, Singularity analysis of generating functions, SIAM J. Discrete Math. 3 (2) (1990) 216–240.

[9] I.M. Gessel, Symmetric functions and P-recursiveness, J. Combin. Theory Ser. A 53 (2) (1990) 257–285.
[10] I.M. Gessel, Counting paths in Young's lattice, J. Statist. Plann. Inference 34 (1) (1993) 125–134.
[11] C.D. Godsil, B.D. McKay, Asymptotic enumeration of Latin rectangles, J. Combin. Theory, B 48 (1990) 19–44.
[12] I.P. Goulden, D.M. Jackson, Combinatorial Enumeration, Wiley, New York, 1983.
[13] I.P. Goulden, D.M. Jackson, J.W. Reilly, The Hammond series of a symmetric function and its application to *P*-recursiveness, SIAM J. Algebraic Discrete Methods 4 (2) (1983) 179–193.
[14] H. Gupta, Enumeration of symmetric matrices, Duke Math. J. 35 (1968) 653–659.
[15] J. Hammond, On the use of certain differential operators in the theory of equations, Proc. London Math. Soc. 14 (1883) 119–129.
[16] W.K. Hayman, A generalization of Stirling's formula, J. Reine Angew. Math. 196 (1956) 67–95.
[17] E.L. Ince, Ordinary Differential Equations, Dover Publications, New York, 1956 (reprint of the 1926 edition).
[18] R. Jungen, Sur les séries de Taylor n'ayant que des singularités algébrico-logarithmiques sur leur cercle de convergence, Comment. Math. Helv. 3 (1931) 266–306.
[19] D.E. Knuth, Permutations, matrices, and generalized Young tableaux, Pacific J. Math. 34 (1970) 709–727.
[20] L. Lipshitz, The diagonal of a *D*-finite power series is *D*-finite, J. Algebra 113 (2) (1988) 373–378.
[21] I.G. Macdonald, Symmetric Functions and Hall Polynomials, second ed., The Clarendon Press, Oxford University Press, New York, 1995.
[22] P.A. MacMahon, Combinatory Analysis, 2 vols. (bound as one), Chelsea Publishing Co., New York, 1960.
[23] B. Malgrange, Sur la réduction formelle des équations différentielles à singularités irrégulières, preprint, 1979.
[24] C. Mallinger, Algorithmic manipulations and transformations of univariate holonomic functions and sequences, Master's Thesis, RISC, Johannes Kepler Universität Linz, Austria, August 1996.
[25] B.D. McKay, The asymptotic numbers of regular tournaments, eulerian digraphs and eulerian oriented graphs, Combinatorica 10 (4) (1990) 367–377.
[26] M.J. Mishna, Une approche holonome à la combinatoire algébrique, Doctorat en mathématiques, UQÀM, Montreal, Canada, Nov 2003.
[27] T. Oaku, N. Takayama, An algorithm for de Rham cohomology groups of the complement of an affine variety via *D*-module computation, J. Pure Appl. Algebra 139 (1–3) (1999) 201–233 Effective Methods in Algebraic Geometry, Saint-Malo, 1998..
[28] A.M. Odlyzko, Asymptotic enumeration methods, in: R. Graham, M. Grötschel, L. Lovász (Eds.), Handbook of Combinatorics, vol. 2, Elsevier, Amsterdam, 1995, pp. 1063–1229.
[29] R.C. Read, N.C. Wormald, Number of labeled 4-regular graphs, J. Graph Theory 4 (2) (1980) 203–212.
[30] M. Saito, B. Sturmfels, N. Takayama, Gröbner deformations of hypergeometric differential equations, Algorithms and Computation in Mathematics, vol. 6, Springer, Berlin, 2000.
[31] B. Salvy, P. Zimmermann, Gfun: a Maple package for the manipulation of generating and holonomic functions in one variable, ACM Trans. Math. Software 20 (2) (1994) 163–177.
[32] N.J.A. Sloane, Ed. The On-Line Encyclopedia of Integer Sequences, 2003. http://www.research.att.com/~njas/sequences/.
[33] R.P. Stanley, Enumerative Combinatorics. vol. I, The Wadsworth & Brooks/Cole Mathematics Series, Wadsworth & Brooks/Cole Advanced Books & Software, Monterey, CA, 1986 (with a foreword by Gian-Carlo Rota).
[34] R.P. Stanley, Enumerative Combinatorics, vol. 2, Cambridge University Press, Cambridge, 1999.
[35] J.R. Stembridge, A Maple package for symmetric functions, J. Symbolic Comput. 20 (5–6) (1995) 755–768 Symbolic Computation in Combinatorics Δ_1 Ithaca, NY, 1993.
[36] S. Sundaram, The Cauchy identity for $\mathrm{Sp}(2n)$, J. Combin. Theory Ser. A 53 (2) (1990) 209–238.
[37] N. Takayama, An algorithm of constructing the integral of a module—an infinite dimensional analog of Gröbner basis, in: Proceedings of ISSAC'90, Kyoto, ACM, 1990, pp. 206–211.
[38] N. Takayama, An approach to the zero recognition problem by Buchberger algorithm, J. Symbolic Comput. 14 (2–3) (1992) 265–282.
[39] É. Tournier, Solutions formelles d'équations différentielles, Doctorat d'état, Université scientifique, technologique et médicale de Grenoble, 1987.
[40] H. Tsai, Weyl closure of a linear differential operator, J. Symbolic Comput. 29 (4–5) (2000) 747–775 Symbolic Computation in Algebra, Analysis, and Geometry, Berkeley, CA, 1998.

[41] H. Tsai, Algorithms for associated primes, Weyl closure, and local cohomology of *D*-modules, in: Local Cohomology and its Applications (Guanajuato, 1999), Lecture Notes in Pure and Applied Mathematics, vol. 226, Dekker, New York, 2002, pp. 169–194.
[42] M. van Hoeij, Formal solutions and factorization of differential operators with power series coefficients, J. Symbolic Comput. 24 (1) (1997) 1–30.
[43] W. Wasow, Asymptotic Expansions for Ordinary Differential Equations, Dover Publications Inc., New York, 1987 (Reprint of the John Wiley 1976 edition).
[44] J. Wimp, D. Zeilberger, Resurrecting the asymptotics of linear recurrences, J. Math. Anal. Appl. 111 (1985) 162–176.
[45] M. Wyman, The asymptotic behavior of the Laurent coefficients, Canad. J. Math. 11 (1959) 534–555.
[46] D. Zeilberger, The method of creative telescoping, J. Symbolic Comput. 11 (3) (1991) 195–204.