An automatic approach to the generating functions of some special sequences

Weiping Wang [1,2,*] Tianming Wang [2]

[1] School of Science, Zhejiang Sci-Tech University
Hangzhou 310018, P. R. China

[2] School of Mathematical Sciences, Dalian University of Technology
Dalian 116024, P. R. China

Abstract

In this paper, we give explicit algorithms to compute generating functions of some special sequences, based on the operations of differential operators and shift operators in the non-commutative context and Zeilberger's holonomic algorithm. It can be found that not only ordinary generating functions and exponential generating functions but also generating functions of the general form $\sum_n a_n(x)\omega(y,n)$ can now be computed automatically. Moreover, we generalize this approach and present explicit algorithms to compute 2-variable ordinary power series generating functions and mixed-type generating functions. As applications, various examples are given in the paper.

Keywords: Computer proofs; Generating functions; Holonomic functions; Zeilberger's holonomic algorithm

1. Introduction

Generating function (GF) is a very important tool for the study of sequences [18], and a lot of works have been devoted to this subject with the help of computers. For instance, Bergeron and Plouffe [3] outlined an approach to compute the generating function of a series given its first few terms. Salvy and Zimmermann [14] also studied the same problem and developed a package *GFun* to guess the generating functions.

*Corresponding author.
E-mail addresses: wpingwang@yahoo.com (Weiping Wang), wangtm@dlut.edu.cn (Tianming Wang).

In [20], Zeilberger studied Weyl algebra and showed that a large set of combinatorial identities can be proved by using properties of the class of holonomic functions. Furthermore, an algorithmic treatment of these identities was initiated by him, relying on a non-commutative version of Sylvester's dialytic elimination method. Zeilberger and his colleagues also gave a few examples concerning generating functions [2, 19], based on the holonomic theory. To deal with these examples, they made use of Cauchy's coefficient formula and the properties satisfied by the sequences or functions, rather than an elimination way. Some other works based on holonomy can be found in [14], where Salvy and Zimmermann used the *GFun* package to manipulate the generating functions of holonomic sequences. Chyzak and Salvy introduced in [5,6] a more general framework for operators, i.e., Ore algebra, in order to deal with holonomic systems in general, and gave a modified Zeilberger's holonomic algorithm to compute the generating functions of some special sequences (holonomic functions), based on a generalization of the theory of Gröbner basis. Particularly, Chyzak developed the *MgFun* project, which is a collection of Maple packages intended for calculations with multivariate generating functions. The reader is referred to [8] for some further details on this subject.

We have also studied in [9, 12, 16] the eliminations in non-commutative operator algebras and modified Zeilberger's holonomic algorithm, so that terminating hypergeometric identities, q-proper-hypergeometric identities as well as identities with the integral sign can be proved automatically. Furthermore, we find that using similar methods developed in [9,12,16], we can deal with differential operators and shift operators at the same time, and perform eliminations among them. In this way, we establish a unified automatic approach to the computation of generating functions.

In the sequel, we will give the explicit algorithms to compute generating functions of some special sequences, with a Maple implementation *gfsolver* to perform this computation. Readers who need the program may correspond with us via electronic mail.

The article is organized as follows. Some definitions are introduced below. In Section 2, we study formal ordinary power series generating functions (OPSGF). In Section 3, we give the algorithm to compute generating functions of the general form:

$$\sum_n a_n(x)\omega(y,n)\,,$$

where $(\omega(y,n))_{n\in\mathbb{N}}$ is a fixed given sequence. Thus, we can deal with ordinary generating functions, exponential generating functions (EGF) and some unusual generating functions at the same time. As applications, we show some examples in Section 4, and finally, in Sections 5, we discuss the

algorithms for 2-variable ordinary power series generating functions and mixed-type generating functions.

Now, we briefly introduce some definitions.

Let $\mathbb{K}$ be a field of characteristic zero. This field $\mathbb{K}$ will usually be $\mathbb{Q}, \mathbb{R}, \mathbb{C}$ in practice, or be a finitely generated extension of $\mathbb{Q}$.

Definition 1.1 ([11, Definition 2.1 and Theorem 3.7]). Let

$$f(x_1, \cdots, x_n) = \sum a(i_1, \cdots, i_n) x_1^{i_1} \cdots x_n^{i_n} \in \mathbb{K}[[x_1, \cdots, x_n]]$$

be a formal power series in the indeterminates $\boldsymbol{x} = (x_1, \cdots, x_n)$ over $\mathbb{K}$. f is called D-finite if all the derivatives $(\partial/\partial x_1)^{\alpha_1} \cdots (\partial/\partial x_n)^{\alpha_n} f$, for $\alpha_1, \cdots, \alpha_n \geq 0$, lie in a finite dimensional vector space over $\mathbb{K}(x_1, \cdots, x_n)$, the field of rational functions in $x_1, \cdots, x_n$. The sequence $a(i_1, \cdots, i_n)$ is P-recursive if and only if the formal power series $f(x_1, \cdots, x_n)$ is D-finite.

Especially, we have the next definition.

Definition 1.2 ([5, Proposition 1.3 and Definition 1.8]). A function $f(x)$ is called D-finite if and only if it satisfies a linear differential equation with polynomial coefficients. A sequence $a(n)$ is called P-recursive if and only if it satisfies a linear recurrence equation with polynomial coefficients.

D-finite functions and P-recursive sequences have many closure properties, and these two concepts are closely related by the equivalence between P-recursiveness of a sequence and D-finiteness of the associated power series. Moreover, D-finiteness and P-recursiveness can also coexist on the same system. For example, Chebyshev Polynomials of the second kind, which will be discussed in the sequel, as well as many other orthogonal polynomials are in the case. Further details concerning these two concepts can be found in [5, 11, 15].

For convenience, we call *holonomic functions* the functions, sequences and formal power series whenever we want to denote any of these. Particularly, we call holonomic functions those special sequences which will be studied in this paper.

Next, let us introduce the definition of Ore algebra (cf., [5, 6]).

Definition 1.3 ([5, Definition 2.5]). Given two d-tuples of indeterminates $\boldsymbol{x} = (x_1, \cdots, x_d)$ and $\boldsymbol{\partial} = (\partial_1, \cdots, \partial_d)$ along with a field $\mathbb{K}$, we define the associated Ore algebra as the non-commutative ring of polynomials $\mathbb{K}\langle \boldsymbol{x}, \boldsymbol{\partial} \rangle = \mathbb{K}\langle x_1, \cdots, x_d, \partial_1, \cdots, \partial_d \rangle$, with the commutation rules

$$\partial_i x_j = \sigma_i(x_j)\partial_i + \delta_i(x_j), \quad \partial_i \partial_j = \partial_j \partial_i, \quad x_i x_j = x_j x_i,$$

where the σ_i's are endomorphisms of $\mathbb{K}[x_i]$ (as an algebra) extended to $\mathbb{K}[\boldsymbol{x}]$ by the identity, and the δ_i's are endomorphisms of $\mathbb{K}[x_i]$ (as a $\mathbb{K}$-vector space) extended to $\mathbb{K}[\boldsymbol{x}]$ by the identity, with the σ_i's and the δ_i's commuting two by two.

Definition 1.4 ([5, Definition 3.6]). An Ore algebra $\mathbb{K}\langle \boldsymbol{x}, \boldsymbol{\partial} \rangle$ is admissible when it satisfies

$$\sigma_i(x_i) = p_i x_i + q_i\,, \quad \delta_i(x_i) = r_i x_i + s_i\,,$$

where all coefficients are in $\mathbb{K}$ and no p_i is zero.

For instance, take $\sigma(x) = x$ and $\delta(x) = 1$, then $\partial x = x\partial + 1$ and the Ore algebra is the Weyl algebra in a single variable [4]. Now, ∂ can be viewed as the differential operator D_x and $\mathbb{K}\langle x, \partial \rangle = \mathbb{K}\langle x, D_x \rangle$. Similarly, take $\sigma(x) = x+1$ and $\delta(x) = 0$, then $\partial x = (x+1)\partial$, which implies that ∂ is the shift operator and $\mathbb{K}\langle x, \partial \rangle = \mathbb{K}\langle n, S_n \rangle$. We can also find that $\mathbb{K}\langle x, D_x \rangle$ and $\mathbb{K}\langle n, S_n \rangle$ are admissible.

In the sequel, S_k, S_m, S_n denote the shift operators with respect to k, m, n, respectively, and D_x, D_y, D_z denote the differential operators with respect to x, y, z, respectively.

Moreover, the set of operators of the Ore algebra that vanish on a given holonomic function has a prominent role in this article, and for any given holonomic function f, we use $\mathrm{Ann} f$ to denote it, i.e., $\mathrm{Ann} f := \{w \in \mathbb{K}\langle x_1, \cdots, x_d, \partial_1, \cdots, \partial_d \rangle,\ w \cdot f = 0\}$.

2. Algorithm for OPSGF

The algorithms below follow the way suggested by Chyzak in [5]. But instead of Gröbner basis method, we choose a process of elimination similar to that of [16], which is the essential difference from Chyzak's method.

Let $(a_n)_{n\in\mathbb{N}}$ be a P-recursive sequence. As mentioned above, $(a_n)_{n\in\mathbb{N}}$ is a holonomic function. The generating function of $(a_n)_{n\in\mathbb{N}}$ is

$$F(x) = \sum_{n\in\mathbb{N}} a_n x^n\,,$$

which is the sum of the sequence of functions $f_n(x) = a_n x^n$. Then, we can handle the generating function as a discrete-continuous identity.

Let $P(n, S_n)$ be a generic operator in $\mathbb{K}\langle n, S_n \rangle$ that vanishes on a_n, i.e., $P(n, S_n)a_n = 0$. Rewrite P as a polynomial in S_n:

$$P = b_k {S_n}^k + b_{k-1} {S_n}^{k-1} + \cdots + b_0\,,$$

where b_i can be considered as a polynomial in n for $i = 0, \cdots, k$. Since

$$(x^{-1}S_n)(a_n x^n) = x^{-1}(a_{n+1}x^{n+1}) = (S_n a_n)x^n\,,$$

then

$$\begin{aligned} b_k(x^{-1}S_n)^k(a_n x^n) &= b_k a_{n+k}x^n = ((b_k {S_n}^k)a_n)\cdot x^n\,,\\ P(n, x^{-1}S_n)f_n(x) &= (b_k(x^{-1}S_n)^k + b_{k-1}(x^{-1}S_n)^{k-1} + \cdots + b_0)(a_n x^n)\\ &= (P(n, S_n)a_n)\cdot x^n = 0\,. \end{aligned}$$

We can see that $P(n, x^{-1}S_n) \in \mathbb{K}(x)\langle n, S_n\rangle$, so multiplying $P(n, x^{-1}S_n)$ by an adequate power of x yields a polynomial $P' \in \mathbb{K}\langle x, n, D_x, S_n\rangle$ vanishing on $f_n(x)$. These steps are summarized into the following theorem.

Theorem 2.1. *Let $P(n, S_n)$ be a generic operator in $\mathbb{K}\langle n, S_n\rangle$ that vanishes on a_n, where $a := (a_n)_{n\in\mathbb{N}}$ is a P-recursive sequence, then there is some operator $P' \in \mathbb{K}\langle x, n, D_x, S_n\rangle$ vanishing on $f_n(x) := a_n x^n$, where P' can be obtained by multiplying $P(n, x^{-1}S_n)$ a power of x.*

Besides P', we have $(xD_x - n)f_n(x) = (xD_x - n)a_n x^n = xna_n x^{n-1} - na_n x^n = 0$. In other words, $xD_x - n$ also vanishes on $f_n(x)$.

Now, let G be a subset of Ann a in $\mathbb{K}\langle n, S_n\rangle$. Put $G' = \{g'\}_{g\in G} \cup \{xD_x - n\}$, where g' is obtained from g by Theorem 2.1. Then G' is a subset of $\mathbb{K}\langle x, n, D_x, S_n\rangle$ with each element of G' annihilating $f_n(x)$.

For each element $g \in G'$, rewrite it as a polynomial in n:

$$g = b_k n^k + b_{k-1}n^{k-1} + \cdots + b_0\,,$$

where $b_i \in \mathbb{K}\langle x, D_x, S_n\rangle$, $i = 0, \cdots, k$. Since S_n commutes with x and D_x, we can regard S_n as a constant so that b_i can be viewed as elements of $\mathbb{K}\langle x, D_x\rangle$. Recall that for any given polynomials $p, q \in \mathbb{K}\langle x, D_x\rangle$, two polynomials $u, v \in \mathbb{K}\langle x, D_x\rangle$ can be obtained such that $up = vq$ (see [12, 16] for details). Thus, following a similar way suggested in [16], we can eliminate n between every two elements of G', which eventually leads us to a non-empty set $G'' \subset \mathbb{K}\langle x, D_x, S_n\rangle$ such that $gf_n(x) = 0$ for all g in G''.

Next, we proceed as in the algorithm for definite sums by creative telescoping (cf., [5, Section 5.2.2], [17, Theorem 3.2] and [20, Theorem 5.1]).

Each element $g(x, D_x, S_n)$ of G'' can be considered as a polynomial in S_n. Write

$$g(x, D_x, S_n) = (S_n - 1)^i \tilde{g}(x, D_x, S_n)\,,$$

where i is maximal, that is, $\tilde{g}(x, D_x, S_n)$ is not divisible by $S_n - 1$ and $\tilde{g}(x, D_x, 1) \neq 0$. If $i > 0$, then we multiply g with $(n)_i$ from the left, where $(n)_i := n(n-1)\cdots(n-i+1)$ is the i-th falling factorial of n (see [7, p.

6]). According to the rules of non-commutative multiplication and the abbreviation $\bar{g}$ for $(S_n-1)^{i-1}\tilde{g}$, we have

$$\begin{aligned}(n)_i g &= (n)_i(S_n-1)^i\tilde{g} = (n)_i S_n\bar{g} - (n)_i\bar{g}\\ &= S_n(n-1)_i\bar{g} - (n-1)_i\bar{g} + (n-1)_i\bar{g} - (n)_i\bar{g}\\ &= (S_n-1)(n-1)_i\bar{g} - i(n-1)_{i-1}\bar{g}\,.\end{aligned}$$

The last term can be reduced by the same scheme, and finally we obtain

$$(n)_i g = (S_n-1)\sum_{m=1}^{i}(-1)^{m-1}(i)_{m-1}(n-m)_{i-(m-1)}(S_n-1)^{i-m}\tilde{g} + (-1)^i i!\tilde{g}\,,$$

which includes the case $i=0$. Since $\tilde{g}$ is not divisible by S_n-1, we can write

$$(-1)^i i!\tilde{g}(x,D_x,S_n) = (S_n-1)\tilde{g}'(x,D_x,S_n) + S(x,D_x)\,,$$

where $S(x,D_x) = (-1)^i i!\tilde{g}(x,D_x,1) \neq 0$. Thus, we have the recurrence

$$\begin{aligned}&(n)_i g(x,D_x,S_n)\\ &\quad= (S_n-1)\left(\tilde{g}' + \sum_{m=1}^{i}(-1)^{m-1}(i)_{m-1}(n-m)_{i-(m-1)}(S_n-1)^{i-m}\tilde{g}\right) + S\\ &\quad:= (S_n-1)\hat{g}(x,n,D_x,S_n) + S(x,D_x)\end{aligned}$$

which annihilates $f_n(x)$, and then

$$\begin{aligned}0 &= \sum_n (n)_i g(x,D_x,S_n) f_n(x)\\ &= \sum_n (S_n-1)\hat{g}(x,n,D_x,S_n) f_n(x) + \sum_n S(x,D_x) f_n(x)\,.\end{aligned}$$

Since the first term vanishes by telescoping, we have

$$S(x,D_x)\sum_n f_n(x) = \sum_n S(x,D_x) f_n(x) = 0\,.$$

Hence, for given P-recursive sequence $(a_n)_{n\in\mathbb{N}}$, we can compute its formal ordinary power series generating function as follows.

Algorithm 2.2. (Algorithm for computing OPSGF)

INPUT: a set G of operators vanishing on a_n.

OUTPUT: a set of operators vanishing on $\sum_{n\in\mathbb{N}} a_n x^n$.

1. Substitute $x^{-1}S_n$ for S_n in each element of G; multiply each by an adequate power of x to make them all polynomials.

2. Add $xD_x - n$ to the set.

3. Eliminate n by using the modified Euclidean algorithm in the non-commutative context.

4. For each operator $g(x, D_x, S_n)$ obtained, find $\tilde{g}(x, D_x, S_n)$ such that $g(x, D_x, S_n) = (S_n - 1)^i \tilde{g}(x, D_x, S_n)$, where i is as big as possible.

5. Substitute $S_n = 1$ in $\tilde{g}(x, D_x, S_n)$ and obtain the resulting operator $S(x, D_x)$. Then $S(x, D_x)$ vanishes on $\sum_{n\in\mathbb{N}} a_n x^n$. Solve the corresponding differential equation of $S(x, D_x)$ with the initial conditions.

3. Algorithm for general GF

For the sequence $(a_n(x))_{n\in\mathbb{N}}$, let us consider the generating function of the general form:

$$\sum_n a_n(x)\omega(y, n)\,. \tag{3.1}$$

In this way we treat at the same time ordinary GF ($\Leftrightarrow \omega(y, n) = y^n$), exponential GF ($\Leftrightarrow \omega(y, n) = y^n/n!$), and GF according to Ω_n ($\Leftrightarrow \omega(y, n) = \Omega_n y^n$), where $(\Omega_n)_{n\in\mathbb{N}}$ is a fixed given sequence (see [7, Section 1.13]).

Suppose $a_n(x)$ is annihilated by $P(x, n, \theta, S_n) \in \mathbb{K}\langle x, n, \theta, S_n\rangle$. Similarly to Theorem 2.1, we find that

$$P' = P\left(x, n, \theta, \left(\frac{\omega(y, n+1)}{\omega(y, n)}\right)^{-1} S_n\right)$$

vanishes on $a_n(x)\omega(y, n)$. It also holds that $Q' = -D_y(\omega(y, n))/\omega(y, n) + D_y$ annihilates $a_n(x)\omega(y, n)$. Thus, multiplying P', Q' by some adequate factors yields two polynomials

$$P'', Q'' \in \mathbb{K}\langle x, y, n, \theta, D_y, S_n\rangle$$

that vanish on $a_n(x)\omega(y, n)$.

Therefore, we have the following theorem.

Theorem 3.1. *Let $P(x, n, \theta, S_n)$ be a generic operator in $\mathbb{K}\langle x, n, \theta, S_n\rangle$ that vanishes on $a_n(x)$, where $a(x) := (a_n(x))_{n\in\mathbb{N}}$ is a holonomic function, then there is some*

$$P'' \in \mathbb{K}\langle x, y, n, \theta, D_y, S_n\rangle$$

vanishing on $f_n(x) := a_n(x)\omega(y, n)$. Moreover, we can always find another operator

$$Q'' \in \mathbb{K}\langle x, y, n, \theta, D_y, S_n\rangle$$

vanishing on $f_n(x)$.

Now, we can modify Algorithm 2.2 so that it can be used to compute general generating function (3.1) as well. In fact, nothing is changed in Steps 3 to 5. What we should do is determining in Step 1 the sequence $(\omega(y,n))_{n\in\mathbf{N}}$ and substituting $\left(\frac{\omega(y,n+1)}{\omega(y,n)}\right)^{-1} S_n$ for S_n in each element of G and adding Q'' in Step 2. As an example, for exponential GF, we should replace S_n by $y^{-1}(n+1)S_n$ in Step 1 and add $yD_y - n$ in Step 2. It should be noticed that

$$((n+1)S_n)^k = \underbrace{(n+1)S_n(n+1)S_n\cdots(n+1)S_n}_{k} = (n+k)\cdots(n+1){S_n}^k .$$

4. Applications

With the algorithm and the corresponding Maple program, we can compute the generating functions of some special combinatorial sequences.

Example 4.1. Let N be a finite set and $|N| = n$. Let $d(n)$ be the number of derangements of N. Then $d(n)$ satisfies the following recurrence relation (see [7, Section 4.2]):

$$d(n+1) = nd(n) + nd(n-1) .$$

We now use *gfsolver* to find the differential equation satisfied by the EGF of $d(n)$:

```
> GL:=[Sn^2-(n+1)*Sn-(n+1)]:
  EQ:=gfgeteq(GL,x,Dx,expogf);
```

$$EQ := [(1-x)\,Dx - x]$$

Then, by appealing to the ODE solver provided by Maple, we have

```
> dsolve({(1-x)*diff(f(x),x)-x*f(x),f(0)=1});
```

$$f(x) = -\frac{e^{-x}}{-1+x}$$

which suggests that

$$D(x) = \sum_{n=0}^{\infty} d(n)\frac{x^n}{n!} = e^{-x}(1-x)^{-1} .$$

In fact, following the algorithm, we can obtain the result even by hand. The readers may have a try.

Remark. In the case of computing OPSGF with *a single recurrence*, our algorithm is essentially the same as the *rectodiffeq* procedure in *GFun* package [14], which makes the conversion between recurrence and differential equation by translating each $n^k u_{n+i}$ into

$$\left(z\frac{\mathrm{d}}{\mathrm{d}z}\right)^k [y(z)z^{-i}].$$

While in the case of computing EGF, our algorithm is different from the usual one. For instance, if we want to compute the EGF of $d(n)$ with the *GFun* package, we should first turn the recurrence of $d(n)$ to that of $d(n)/n!$, and then translate the recurrence to differential equation satisfied by the generating function, as follows:

```
> RE:=d(n+1)-n*d(n)-n*d(n-1):
> RE:={subs(n=n+1,RE),d(0)=1};
```

$$RE := \{d(n+2) - (n+1)\,d(n+1) - (n+1)\,d(n), d(0) = 1\}$$

```
> RE2:='rec*rec'(RE,{(n+1)*d(n+1)-d(n),d(0)=1},d(n));
```

$$RE2 := \{-d(n) + (-n-1)\,d(n+1) + (n+2)\,d(n+2), d(0) = 1\}$$

```
> DE:=rectodiffeq(RE2,d(n),F(x));
```

$$DE := \left\{-F(x)\,x + (1-x)\left(\frac{d}{dx}F(x)\right) - \_C0, F(0) = 1\right\}$$

```
> dsolve(subs(_C[0]=0,DE),F(x));
```

$$F(x) = -\frac{e^{-x}}{x-1}$$

Comparing with Example 4.1, it can be seen that our algorithm is more convenient. In fact, it provides a unified approach to OPSGF and EGF.

Example 4.2. Compute the EGF of the number g_n of clouds with n points (see [7, Section 7.3]). $g(n)$ is also named as the number of undirected 2-regular labeled graphs and satisfies the following recurrence relation:

$$g_n = (n-1)g_{n-1} + \binom{n-1}{2} g_{n-3},\ n \geq 3;\ g_0 = 1,\ g_1 = g_2 = 0.$$

```
> GL:=[2*Sn^3-2*(n+2)*Sn^2-(n+2)*(n+1)]:
  EQ:=gfgeteq(GL,x,Dx,expogf);
```

$$EQ := [(-2\,x+2)\,Dx - x^2]$$

```
> simplify(dsolve({(-2*x+2)*diff(g(x),x)-x^2*g(x),g(0)=1}));
```

$$g(x) = \frac{ie^{-\frac{1}{4}x(x+2)}}{\sqrt{x-1}}$$

Then the generating function is

$$g(x) = \sum_{n=0}^{\infty} g_n \frac{x^n}{n!} = \frac{1}{\sqrt{1-x}} \exp\left(-\frac{x^2+2x}{4}\right).$$

The algorithm also holds for some generic holonomic functions, including some orthogonal polynomials. In the following example, let us compute the OPSGF of the orthogonal Chebyshev Polynomials of the second kind.

Example 4.3. We recall the definition of these polynomials as well as some equations that they satisfy (see [1, formulae(22.3.7, 22.6.12, 22.8.4)]):

$$U_n(x) = \sum_{m=0}^{[\frac{n}{2}]} (-1)^m \frac{(n-m)!}{m!(n-2m)!}(2x)^{n-2m},$$
$$(1-x^2)U_n''(x) - 3xU_n'(x) + n(n+2)U_n(x) = 0,$$
$$(1-x^2)U_{n+1}'(x) + (n+1)xU_{n+1}(x) - (n+2)U_n(x) = 0.$$

Input these equations and call *gfgeteq* function to find the relations satisfied by the derivatives of the generating function:

```
> GL:=[(1-x^2)*Dx^2-3*x*Dx+n*(n+2),
       (1-x^2)*Dx*Sn+(n+1)*x*Sn-(n+2)]:
  EQ:=gfgeteq(GL,y,Dy);
```

$$EQ := [Dx^2 - Dx^2x^2 - 3\,xDx + 3\,yDy + y^2Dy^2,$$
$$(-y^2 + yx)\,Dy - 2\,y + Dx - Dx\,x^2]$$

It should be noticed that the output will be given in the standard form in which the indeterminates are in front of the differential operators. If it is not, it is still meant that way. Now, eliminate D_x from the result:

```
> LEQdy:=gfeliminate(EQ[1],EQ[2],Dx);
```

$$LEQdy := y\left(-2\,yx + y^2 + 1\right)Dy^2 + \left(-8\,yx + 6\,y^2 + 2\right)Dy + 6\,y - 4\,x$$

Thus, using ODE solver again gives us the desired generating function:

```
> dsolve({y*(-2*y*x+y^2+1)*diff(f(y),y,y)+(-8*y*x+6*y^2+2)
*diff(f(y),y)+(6*y-4*x)*f(y),f(0)=1});
```

$$f(y) = \frac{1}{-2\,yx + y^2 + 1}$$

Besides Example 4.3, we have verified the generating functions of many other orthogonal polynomials, such as Hermite polynomials, Ultraspherical polynomials, Legendre polynomials as well as generalized Laguerre polynomials.

Finally, let us consider a special type of generating function. It can be found that the sequence $(\omega(y,n))_{n\in\mathbf{N}}$ is now defined by $(y^n/(n!)^2)_{n\in\mathbf{N}}$.

Example 4.4. Let P_n be the number of bipermutations, then P_n satisfies the following recurrence relation (see [7, Section 6.3]):

$$P_n = \binom{n}{2}(2P_{n-1} + (n-1)P_{n-2}).$$

By means of the algorithm, we can compute the GF of P_n, as mentioned above, which has the form $f(x) = \sum_{n=0}^{\infty} P_n x^n/(n!)^2$. The following are the input and output in Maple.

```
> GL:=[2*Sn^2-2*(n+2)*(n+1)*Sn-(n+2)*(n+1)^2]:
  EQ:=gfgeteq(GL,x,Dx,x^n/(n!*n!),other);
```

$$EQ = [(-2x+2)\,Dx - x]$$

```
> dsolve({(2-2*x)*diff(f(x),x)-x*f(x),f(0)=1});
```

$$f(x) = \frac{ie^{-\frac{1}{2}x}}{\sqrt{-1+x}}$$

Thus, we obtain the final result:

$$f(x) = \sum_{n=0}^{\infty} P_n \frac{x^n}{n!^2} = \frac{e^{-x/2}}{\sqrt{1-x}}.$$

Analogously, let $a_n := 4^{-n}\sum_i (2n-2i)!i!\binom{n}{i}^2 2^i$, then a_n satisfies the recurrence $a_n = n^2 a_{n-1} - (n-1)\binom{n}{2}a_{n-2}$ (see [7, Exercise 25, p. 125]). According to the algorithm, we can obtain the following generating function:

$$\sum_{n=0}^{\infty} a_n \frac{x^n}{n!^2} = e^{x/2}(1-x)^{-1/2}.$$

The readers may verify it themselves.

5. Algorithms in 2-variable case

By far, what we have discussed are all in single-variable case. In fact, we can extend those studied above to 2-variable case and give algorithms to compute 2-variable ordinary power series generating functions and mixed type generating functions.

Theorem 5.1. *Let*

$$P(\boldsymbol{x}, n, m, \boldsymbol{\partial}, S_n, S_m)$$

be a generic operator in $\mathbb{K}\langle \boldsymbol{x}, n, m, \boldsymbol{\partial}, S_n, S_m\rangle$ *that vanishes on* $a_{n,m}(\boldsymbol{x})$, *where* $a(\boldsymbol{x}) := (a_{n,m}(\boldsymbol{x}))_{n,m\in\mathbb{N}}$ *is a holonomic function, then*

$$P_1(\boldsymbol{x}, n, m, \boldsymbol{\partial}, y^{-1}S_n, z^{-1}S_m) \text{ and } P_2(\boldsymbol{x}, n, m, \boldsymbol{\partial}, y^{-1}(n+1)S_n, z^{-1}S_m)$$

vanish on $a_{n,m}(\boldsymbol{x})y^n z^m$ *and* $a_{n,m}(\boldsymbol{x})y^n z^m/n!$, *respectively. Multiplying* P_1 *and* P_2 *by some* $y^p z^q$, *where* p *and* q *are some adequate non-negative integers, we can obtain two polynomials*

$$P_1', P_2' \in \mathbb{K}\langle \boldsymbol{x}, y, z, n, m, \boldsymbol{\partial}, D_y, D_z, S_n, S_m\rangle$$

that vanish on $a_{n,m}(\boldsymbol{x})y^n z^m$ *and* $a_{n,m}(\boldsymbol{x})y^n z^m/n!$.

We omit the proof here, because it is similar to that of Theorem 2.1. Once we have found the operators that annihilate the summand of the generating function, what we should do next is eliminating n and m among them. To do this, we may follow the way suggested by [16, Algorithm 4]. Let us recall the algorithm and modify it to our context at the same time.

Algorithm 5.2. (Algorithm for eliminating two variables)

INPUT: a set of operators AS in $\mathbb{K}\langle \boldsymbol{x}, y, z, n, m, \boldsymbol{\partial}, D_y, D_z, S_n, S_m\rangle$.

OUTPUT: a set of operators which are free of n and m.

1. Set $AS := \{A_1, A_2, \cdots, A_l\}$. Divide AS into two subsets B and C so that each element b_i of B satisfies $\deg_n b_i = 0$ and each element c_i of C satisfies $\deg_n c_i > 0$.

2. If $|C| = 1$, set $AS := B$; else, let $|C| = k$, choose c_j such that $\deg_n c_j = \min\{\deg_n c_i,\ c_i \in C\}$, construct $k-1$ pairs $\{c_i, c_j\}$, $c_i \in C$, $i \neq j$, and obtain $k-1$ polynomials r_i by eliminating n respectively. Append r_i into B and set $AS := B$.

3. Now, each element a_i of AS satisfies $\deg_n a_i = 0$. Similarly, choose a_j which has the smallest non-zero degree on m, and construct pairs to eliminate m. Thus, a set of operators free of n and m can be obtained.

We now propose the algorithm to compute the 2-variable generating functions.

Algorithm 5.3. (Algorithm for computing 2-variable GF)

INPUT: a set G of operators vanishing on $a_{n,m}(\boldsymbol{x})$.

OUTPUT: a set of operators vanishing on $\sum_{n,m\in\mathbb{N}} a_{n,m}(\boldsymbol{x})y^n z^m$.

1. Substitute $y^{-1}S_n$, $z^{-1}S_m$ for S_n, S_m in each element of G; multiply each by an adequate power to make them all polynomials.

2. Add $yD_y - n$ and $zD_z - m$ to the set.
3. Eliminate n and m by Algorithm 5.2.
4. For each $g(\boldsymbol{x}, y, z, \theta, D_y, D_z, S_n, S_m)$ obtained, find

$$\tilde{g}(\boldsymbol{x}, y, z, \theta, D_y, D_z, S_n, S_m)$$

such that $\tilde{g}(\boldsymbol{x}, y, z, \theta, D_y, D_z, 1, 1) \neq 0$. Then we can see that the operator $\tilde{g}(\boldsymbol{x}, y, z, \theta, D_y, D_z, 1, 1)$ vanishes on $\sum_{n,m\in\mathbb{N}} a_{n,m}(\boldsymbol{x}) y^n z^m$.

For generating functions of mixed type, i.e., it is an OPSGF with respect to z and an EGF with respect to y, we can modify Step 1 of Algorithm 5.3 by substituting $y^{-1}(n+1)S_n, z^{-1}S_m$ for S_n, S_m, respectively.

Example 5.4. Verify

$$G(x,y) = \sum_{n,k\geq 0} f(n,k)\frac{x^n}{n!}y^k = \sum_{n,k\geq 0}\binom{n}{k}\frac{x^n}{n!}y^k = e^{x(1+y)}.$$

Since $f(n,k)$ is annihilated by $P(n,k,S_n,S_k) = (n+1-k)S_n - (n+1)$ and $Q(n,k,S_n,S_k) = (k+1)S_k - (n-k)$, we can obtain the following operators vanishing on $f(n,k)x^n y^k/n!$:

$$(n+1-k)(n+1)S_n - x(n+1)\,,\ (k+1)S_k - y(n-k)\,,\ xD_x - n\,,\ yD_y - k\,.$$

By eliminating n, we have

$$-S_n kxD_x - x^2 D_x - x + S_n x^2 {D_x}^2 + S_n xD_x\,,\ (S_k+y)k - yxD_x\,,\ -k + yD_y\,,$$

and by eliminating k, we have

$$-x^2 D_x - x + S_n x^2 {D_x}^2 + S_n xD_x - S_n xyD_xD_y\,,\ -yxD_x + (S_k+y)yD_y\,.$$

Then setting $S_n = S_k = 1$ yields the operators annihilating $f(n,k)x^n y^k/n!$:

$$x^2D_x^2 - x^2D_x + xD_x - xyD_xD_y - x\,,\ -xyD_x + (1+y)yD_y\,.$$

Eliminating D_x, we have

$$y^2(1+y)D_y^2 - y^2(x+xy-1)D_y - xy^2\,.$$

Based on $G(x,0) = e^x$ and $G(x,1) = e^{2x}$, Maple gives the result:

```
> simplify(dsolve({y^2*(1+y)*diff(f(y),y,y)-y^2*(x+x*y-1)
*diff(f(y),y)-x*y^2*f(y),f(0)=exp(x),f(1)=exp(2*x)}));
```

$$f(y) = e^{x(1+y)}$$

Remark. This example can also be done by using Koepf's algorithm given in [10] twice. In this way the generating function is used to find the double sum representing it.

```
> fps:=FPS(exp(x*(1+y)),x,n);
```

$$fps := \sum_{n=0}^{\infty} \frac{(1+y)^n x^n}{n!}$$

```
> standardsum(FPS(fps,y,k));
```

$$\sum_{n=0}^{\infty} \left(\sum_{k=0}^{\infty} \frac{x^n (-1)^k \,\mathrm{pochhammer}\,(-n,k)\, y^k}{n!\, k!} \right)$$

Or one can compute the generating function iteratively by first summing with respect to k and afterwards with respect to n by using the *GFun* package.

Example 5.5. Let $s(n,k)$ and $S(n,k)$ be the Stirling numbers of the first and second kinds (e.g., see [7, Chapter 5]). The numbers $s(n,k)$ satisfy the recurrence $s(n,k) = s(n-1,k-1)-(n-1)s(n-1,k)$, from which we can find that the operator $g = S_n S_k + nS_k - 1$ annihilates $s(n,k)$. Then the operators $P_1 = (S_k S_n + xS_k)n - xy$, $P_2 = -n + xD_x$ and $P_3 = -k + yD_y$ vanish on $s(n,k)x^n y^k/n!$. Because $P_1 + (S_k S_n + xS_k)P_2 = -xy + (S_k S_n + xS_k)xD_x$ is free of n and k, then the operator $(1+x)D_x - y$ vanishes on the generating function $\psi(x,y) = \sum_{n,k\geq 0} s(n,k)x^n y^k/n!$. Let

$$f(x) := \psi(x,y) = \sum_{n=0}^{\infty} \left(\sum_{k=0}^{n} s(n,k) y^k \right) \frac{x^n}{n!},$$

then $f(0) = s(0,0) = 1$, and we have

```
> dsolve({(1+x)*diff(f(x),x)-y*f(x),f(0)=1});
```

$$f(x) = (1+x)^y$$

In the above, we obtain the ordinary differential equation of the associated mixed-type GF from just one recurrence relation. In fact, it can not be expected that we are so lucky each time. For example, when we compute the generating function $\phi(x,y) = \sum_{n,k\geq 0} S(n,k)x^n y^k/n!$ of Stirling numbers of the second kind, based on the recurrence relation $S(n,k) = S(n-1,k-1)+kS(n-1,k)$, we only obtain the following partial differential equation:

$$(D_x - yD_y - y)\phi(x,y) = 0\,.$$

```
> simplify(pdsolve(diff(f(x,y),x)
                   -y*diff(f(x,y),y)-y*f(x,y)));
```

$$f(x,y) = \_F1\left(ye^{x}\right)e^{-y}$$

Let $g(x) := \phi(x,y)$, then $g(0) = S(0,0) = 1$, which means $\_F1\left(ye^{0}\right)e^{-y} = 1$. Then $\_F1(y) = e^{y}$, from which we obtain

$$\phi(x,y) = e^{y(e^{x}-1)}\,.$$

Theorem 3.1 can be adapted to the 2-variable case easily. For instance, let us compute the generating function of Lah numbers (see [7, Exercise 2, p. 156]).

Example 5.6. The generating function of Lah numbers $L_{n,k}$ is

$$f(x,y) = 1 + \sum_{k=1}^{n} L_{n,k}\frac{(-x)^n}{n!}y^k\,,$$

where $L_{n,k} = (-1)^n\binom{n-1}{k-1}n!/k!$. According to the algorithm, we can finally find the operators which annihilate $f(x,y)$ by eliminating n and k:

$$(1-x)xD_x - yD_y\,,\ -xD_x + yD_y + yD_y{}^2\,,\ (1-x)D_x - yD_y - y\,.$$

Thus, from the first and the third operators, we obtain $(1-x)^2D_x - y$. Using *dsolve*, we have

```
> simplify(dsolve({(1-x)^2*diff(f(x),x)-y*f(x),f(0)=1}));
```

$$f(x) = e^{-\frac{yx}{-1+x}}$$

which means

$$f(x,y) = 1 + \sum_{k=1}^{n} L_{n,k}\frac{(-x)^n}{n!}y^k = e^{xy(1-x)^{-1}}\,.$$

Acknowledgments

The first author is supported by the National Natural Science Foundation of China under Grant 11001243 and the Science Foundation of Zhejiang Sci-Tech University (ZSTU) under Grant 1013817-Y.

References

[1] M. Abramowitz, I. A. Stegun, Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, Reprint of the 1972 edition, Dover Publications, Inc., New York, 1992.

[2] G. Almkvist, D. Zeilberger, The method of differentiating under the integral sign, J. Symbolic Comput. 10 (6) (1990) 571–591.

[3] F. Bergeron, S. Plouffe, Computing the generating function of a series given its first few terms, Experiment. Math. 1 (4) (1992) 307–312.

[4] J.-E. Björk, Rings of Differential Operators, North-Holland Publishing Co., Amsterdam-New York, 1979.

[5] F. Chyzak, Holonomic systems and automatic proofs of identities, Research Report 2371, Institut National de Recherche en Informatique et en Automatique, October 1994.

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

[7] L. Comtet, Advanced Combinatorics, D. Reidel Publishing Co., Dordrecht, 1974.

[8] P. Flajolet, B. Salvy, Computer algebra libraries for combinatorial structures, J. Symbolic Comput. 20 (5-6) (1995) 653–671.

[9] Y. Huang, T. Wang, Elimination in Weyl algebra and q-identities, J. Syst. Sci. Complex. 20 (4) (2007) 601–609.

[10] W. Koepf, Power series in computer algebra, J. Symbolic Comput. 13 (6) (1992) 581–603.

[11] L. Lipshitz, D-finite power series, J. Algebra 122 (2) (1989) 353–373.

[12] H. Liu, T. Wang, Elimination and identities with the integral sign, J. Syst. Sci. Complex. 19 (4) (2006) 470–477.

[13] M. Petkovšek, H. S. Wilf, D. Zeilberger, $A = B$, A K Peters, Ltd., Wellesley, MA, 1996.

[14] 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.

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

[16] T. Wang, W. Wang, Y. Xu, Eliminations in Weyl algebras and identities, Adv. in Appl. Math. 35 (3) (2005) 254–270.

[17] K. Wegschaider, Computer Generated Proofs of Binomial Multi-Sum Identities, Diploma Thesis, RISC, Johannes Kepler University, Linz, 1997.

[18] H. S. Wilf, Generatingfunctionology, Second edition, Academic Press, Inc., Boston, MA, 1994.

[19] H. S. Wilf, D. Zeilberger, An algorithmic proof theory for hypergeometric (ordinary and "q") multisum/integral identities, Invent. Math. 108 (3) (1992) 575–633.

[20] D. Zeilberger, A holonomic systems approach to special functions identities, J. Comput. Appl. Math. 32 (3) (1990) 321–368.