Counting tensor structures

Giel van Bergen

Bachelor Thesis

June 21, 2013

FNWI
Radboud Universiteit Nijmegen

Contents

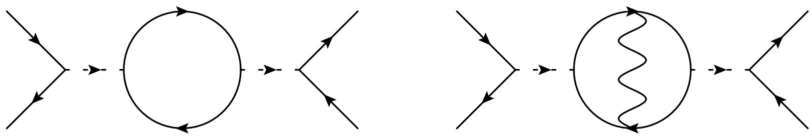
1 Introduction

Particle accelerators become more powerful and accurate, and to test the outcomes from the experiments to the predictions of theories, higher order loop corrections have to be calculated. The bottleneck there is the calculation of loop integrals. A method is described to simplify these integrals and new results are derived.

About notation, there is a reference in the appendix for the notation used. The appendix also contains a mathematical derivation that would be too much of a digression from the flow of the text.

2 Feynman Diagrams

Feynman diagrams are the prime method for calculating all kinds of things in high energy physics. They are diagrams consisting of lines (propagators) and parts where two lines join or branch (vertices). Two examples are shown in Figure 1. Every possible diagram describes a process that can occur between



(a) A single loop (b) A double loop

Figure 1: Two Feynman diagrams, a particle and an anti-particle scatter off each other, with two different processes in between.

particles. They work like a perturbation series and the larger[1] the diagram, the smaller it's contribution. All vertices and propagators contribute to the diagram, but I will be concerned with loops like the ones in the diagrams. Diagram (a) has one loop and diagram (b) has two loops. Of course one can imagine more loops.

To get more accurate values, higher order terms have to be considered. I will discuss arbitrary numbers of loops. One loop represents an integral of the form

$$\int \frac{dq}{D(q+p_1)D(q+p_2)\cdots D(q+p_n)} \tag{1}$$

with the denominators – also called the inverse propagators – $D(q+p_j) = (q+p_j)^2 - m_j^2 = q^2 + 2p_j \cdot q + \sigma_j$. The integrand has one D_j for each particle in the loop, with p_j

The more propagators, the harder it is to evaluate the integral, so methods have been devised to reduce the hard integrals to easier ones. One way this can be done is similar to partial fractions decomposition [1]. One writes the 1

[1]Larger in the sense of more propagators and vertices, not actual extent of the diagram.

in the numerator as $1 = \sum_j F_j D(q+p_j)\,\forall q$ (the $1=1$ problem), then one can write the integral as sum of integrals with less denominators

$$\begin{aligned}\frac{1}{D_1\cdots D_n} &= \frac{\sum_{j=1}^{n} F_j D_j}{D_1\ldots D_n} \\ &= \frac{F_1}{D_2\ldots D_n} + \frac{F_2}{D_1 D_3\cdots D_n} + \frac{F_n}{D_1\cdots D_{n-1}}. \end{aligned} \tag{2}$$

The major issue at this moment is determining when the resulting equations have a solution. The F_j can be more than just scalars; also combinations of powers of q, that is

$$F_j = a_j + \sum_{k=1}^{4} t_{j,k}\cdot q + \sum_{k,l=1}^{4} (s_{jk}\cdot q)(s'_{jl}\cdot q) + \cdots$$

Of course, the sum is finite. Higher powers of q give a bigger chance of obtaining enough equations solve the $1 = 1$ problem. The $1 = 1$ problem then finally looks like

$$1 = A + B_\mu q^\mu + C_{\mu\nu} q^\nu q^\mu + D q^2 + E_\mu q^2 q^\mu + \cdots \tag{3}$$

The reduction technique discussed here is already solved for the one loop case [1], this discussion will be for arbitrary number of loops, with emphasis on two loops.

In the case of two loops, the Feynman integral becomes

$$\int \frac{dl_1\, dl_2}{D_1\cdots D_n},$$

where the D_j are of the form $D(l_1+p_j), D(l_2+p_j)$ or $D(l_1+l_2+p_j)$. Similarly with L loops, the integral has the form

$$\int \frac{dl_1\cdots dl_L}{D_1\cdots D_n}$$

with D_j of the form $D((l_1) + (l_2) + \cdots (l_L) + p_j)$, where the loop momenta between parentheses can be either present or not. Some integrals are trivially decomposable, for example

$$\int \frac{dl_1 dl_2}{D(l_1+p_1)D(l_2+p_2)} = \int \frac{dl_1}{D(l_1+p_1)} \int \frac{dl_2}{D(l_2+p_2)}.$$

This happens when the diagram consists of two concatenated single loops rather than a real double loop. I will therefore only discuss true loops. This means that every possible inner product appears in some D_j. On tensor notation: The subject of this discussion will be tensors like

$$l_1^{2a}(l_1\cdot l_2)^b l_2^{2b} l_1^{\mu_1}\cdots l_1^{\mu_\alpha} l_2^{\nu_1}\cdots l_2^{\nu_\beta}.$$

Because this is a very cluttered and therefore non-transparent notation so, if it is advantageous, I will write such tensors in shorthand as

$$l_1^{2a}(l_1\cdot l_2)^b l_2^{2b} l_1^{\mu_1}\cdots l_1^{\mu_\alpha} l_2^{\nu_1}\cdots l_2^{\nu_\beta} \leftrightarrow (a,b,c)(\alpha,\beta)$$

which contains the same information, but is much tidier. I will write $\mathbf{a} = (a, b, c)$ and $\boldsymbol{\alpha} = (\alpha, \beta)$, such that $\mathbf{l} = \mathbf{a}\boldsymbol{\alpha} = (a, b, c)(\alpha, \beta)$. It turns out to be useful to introduce two norms on the tuples: the L^1-norm[2]: $|\mathbf{a}| = \sum_j a_j$ and the Hamming length: $H(\mathbf{a}) = \#\{k \in a | k \neq 0\}$, which counts the number of non-zero elements of $\mathbf{a}$.

For describing a general tensor created from 3 or more tensors the notation and norms are easily generalised.

3 Counting Tensor Structures

3.1 A Two Loop Example

What makes the problem of finding the number of tensor structures interesting and non-trivial? An elaborate example (in two loops) to illustrate the ideas: say the degrees of the F_j's are $\varphi = 2$, so they are a linear combination of

$$1, l_1^\mu, l_2^\mu, l_1^\mu l_1^\nu, l_1^\mu l_2^\nu, l_2^\mu l_2^\nu.$$

The D_j's contain terms proportional to[3]

$$1, l_1^\mu, l_2^\mu, l_1^2, l_2^2, l_1 \cdot l_2$$

then $\sum_j D_j F_j$ is a linear combination of

$$1, l_i^\mu, l_i^\mu l_j^\mu, l_i \cdot l_j, l_i^\mu l_j^\nu l_k^\rho, l_i \cdot l_j l_k^\mu \text{ and } l_i \cdot l_j l_k^\mu l_m^\nu \tag{4}$$

with i, j, k, m ranging over $1, 2$. Basic linear algebra says that the coefficients of the non-constant terms have to be zero, giving a linear system of equations relating the coefficients F_j with the coefficients of the tensor structures. However not all the coefficients have to be zero. Take for example the subproblem $A_{\mu\nu} l_1^\mu l_1^\nu + B l_1 \cdot l_1 = 0 \ \forall l_1$. This does not imply that $A_{\mu\nu} = B = 0$. In fact we can forget about $B = 0$ by recognising that

$$A_{\mu\nu} l_1^\mu l_1^\nu + B l_1 \cdot l_1 = A_{\mu\nu} l_1^\mu l_1^\nu + B g_{\mu\nu} l_1^\mu l_1^\nu = (A_{\mu\nu} + B g_{\mu\nu})\, l_1^\mu l_1^\nu = A'_{\mu\nu} l_1^\mu l_1^\nu.$$

This freedom shows that not every tensor appearing contributes to total number of tensor structures. Only the number of *independent* tensor structures matters. Still $A_{\mu\nu} l^\mu l^\nu = 0$ does not imply that $A_{\mu\nu} = 0$. For example in Euclidean space and $d = 2$, if we take

$$A^\mu{}_\nu = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

then

$$A^\mu{}_\nu x_\mu x^\nu = \begin{pmatrix} x & y \end{pmatrix} \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x & y \end{pmatrix} \begin{pmatrix} -y \\ x \end{pmatrix} = 0.$$

However, there is a one-to-one correspondence between $d \times d$ *symmetric* matrices and quadratic forms, so we can add the requirement of symmetry

[2] Usually in the L^1-norm there is an absolute value sign, but the entries of the vectors here are always non-negative (and integers).

[3] As previously indicated, I will not discuss the case when there are no denominators with overlap (no $l_1 \cdot l_2$), since in that case the integral already factors.

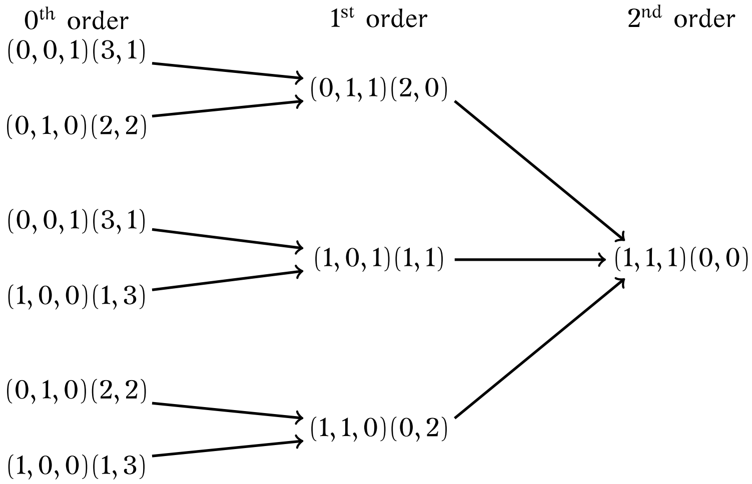


Figure 2: The tensor $(1,1,1)(0,0)$ can be obtained after two contractions from three different tensors.

to $A_{\mu\nu}$. Then the number of independent equations is not d^2 but $d(d+1)/2$, which is the number of different elements in a 2-rank symmetric tensor as proven in the appendix. This is the same as the number independent tensor structures in $l_1^\mu l_1^\nu$. That makes counting the tensor structures a valuable activity.

Note that that the reason that the terms with $l_i \cdot l_j$ and $l_i \cdot l_j l_k^\mu$ don't give independent equations is that they are contractions of $l_i^\mu l_j^\nu$ and $l_i^\mu l_j^\nu l_k^\rho$, which appear also by themselves. Also notice that $l_i \cdot l_j l_k^\mu l_m^\nu$ cannot be forgotten in favor of $l_i^\mu l_j^\nu l_k^\rho l_m^\sigma$, since the only quadratic terms in D_j are already contracted, so $l_i^\mu l_j^\nu l_k^\rho l_m^\sigma$ does not appear in the $1 = 1$ problem.

As proven in the appendix, if we write $[\xi]^j = \xi(\xi+1)\ldots(\xi+j-1) = \Gamma(\xi+j)/\Gamma(\xi)$, the total number of tensor structures of rank j is $[2d]^j/j!$ so $\sum_{j=0}^{3}[2d]^j/j! = [2d+1]^3/3!$ is the number of tensor structures combined. We have to add to that number the tensor structures coming from tensors of the form $l_i \cdot l_j l_k^\mu l_m^\nu$, which is $3 \cdot [2d]^2/2!$ where the three comes from the three different inner products that can be in front of the $l_k^\mu l_m^\nu$.

By the same reasoning that says that the tensor structures from tensors like l_1^2 should not be counted, it also follows that the equality $g_{\mu\nu} l_1^2 l_2^\mu l_2^\nu = g_{\mu\nu} l_2^2 l_1^\mu l_1^\nu$ shows that I have counted 1 tensor structure too many. Therefore for each $l_i \cdot l_j l_k \cdot l_m$ with $i, j \neq k, m$ one has to subtract 1 from the total, giving a total of $[2d+1]^3/3! + 3[2d]^2/2! - 3$ tensor structures.

Although this example only has $\varphi = 2$, if $\varphi \geq 4$ there are tensors with more than this sort of dependencies. For example in $\varphi = 4$, we have the scheme of Figure 2. There are dependencies between the dependencies, lowering the actual number of dependencies, and therefore increasing the number of independent tensor structures. This dependency would count as two since there are two equalities: one between $(0,1,1)(2,0)$ and $(1,0,1)(1,1)$ and the other one between $(1,0,1)(1,1)$ and $(1,1,0)(0,2)$. However since for each first order tensor one dependency has been thrown away, only one extra dependency remains that has to be added to the total number of tensor structures.

3.2 Two Dimensions

In [2] a program is described that can calculate the number of tensor structures for φ up to 4 and arbitrary d. Briefly stated it generates many random values for l_1^μ and l_2^μ and substitutes them in $\sum_j F_j D_j = 1$. This gives a big linear system which it solves. The number of tensor structures then is the rank of that matrix. This gives a method for testing these results. Sadly the result from the previous section does not hold in $d = 2$. Where the formula predicts 62, numeric calculations give 60. The reason lies in an implicit assumption which is false. It is the assumption that contractions are the only dependencies between tensors. In 2 dimensions strange things happen, take the case $\varphi = 2$:[4]

If $\mathbf{l}_1 = (a, b)$ and $\mathbf{l}_2 = (c, d)$ and $A^{\mu\nu} = l_1^2 l_2^\mu l_2^\nu$, $B^{\mu\nu} = l_2^2 l_1^\mu l_1^\nu$ and $C^{\mu\nu} = l_1 \cdot l_2 l_1^\mu l_2^\nu$, then (in this case the superscripts are Lorentz indices)

$$C^{12} + C^{21} = (ac - bd)(bc + ad) = (a^2 - b^2)cd + (c^2 - d^2)ab = A^{12} + B^{12} \quad (5)$$

and

$$\begin{aligned} 2(C^{11} + C^{22}) &= 2\left[(ac - bd)(ac + bd)\right] \\ &= (a^2 - b^2)(c^2 + d^2) + (c^2 - d^2)(a^2 + b^2) = A^{11} + A^{22} + B^{11} + B^{22}. \end{aligned} \quad (6)$$

These two equalities reduce the number of independent tensorstructures by 2 giving the 60 that was determined. Because the interest of course goes out to the 4 dimensions, I have avoided the 2D case and restricted the analysis to $d \geq 3$.

3.3 In General

The construction in general goes along the same lines. Take the trivial tensors

$$\mathbf{l} = \mathbf{a}\boldsymbol{\alpha} = (a_1, \ldots, a_{\mathcal{L}})(\alpha_1, \ldots, \alpha_L)$$

with $\mathbf{a} = 0$ and $\mathcal{L} \equiv L(L + 1)/2$ the number of different inner products that can be formed with L tensors.[5] Those ive just a term $\sum_{j=0}^{\varphi+1}[Ld]^j/j! = [Ld + 1]^{\varphi+1}/(\varphi + 1)!$ since $[Ld]^j/j!$ is the number of tensorstructures in a tensor with $|\alpha| = j$. Without counting the overlap, one gets $\mathcal{L}[Ld]^\varphi/\varphi!$ from the $l_i \cdot l_j(\alpha, \beta)$. The k-th order overlaps Q_k come from tensors with $|\mathbf{a}| = k+1$ and $|\boldsymbol{\alpha}| = \varphi - 2k$. The number of distinct loop momenta that can contract to n is only dependent on $\mathbf{a}$, indeed it is $H(\mathbf{a})$, and the amount of overlap is only dependent on $\boldsymbol{\alpha}$, call it $P(\boldsymbol{\alpha})$. Such that

$$\begin{aligned} Q_k &= \sum_{\substack{|\mathbf{a}|=k+1 \\ |\boldsymbol{\alpha}|=\varphi-2k}} \max\left(H(\mathbf{a}) - k, 0\right) P(\boldsymbol{\alpha}) \\ &= \sum_{|\mathbf{a}|=k+1} \max\left(H(\mathbf{a}) - k, 0\right) \sum_{|\boldsymbol{\alpha}|=\varphi-2k} P(\boldsymbol{\alpha}) \end{aligned}$$

---

[4]The metric is $+-$.

[5]Every l_i has $L-1$ l_j's, $j \neq i$ to contract with. That gives $L(L-1)$ double counted pairs, so $L(L-1)/2$ pairs. The squares l_i^2 add L more inner products, giving a total of $L(L+1)/2$ pairs.

The $-k$ in $\max(H(\mathbf{a}) - k, 0)$ comes from the fact that if $k = 1$ if there are two tensors that contract to $\mathbf{a}\boldsymbol{\alpha}$ then there is only 1 extra dependency and if there is only one tensor that contracts to $\mathbf{a}\boldsymbol{\alpha}$ then there are no dependencies. For higher values of k one should count every joint dependency between T tensors with weight $\max(T - k, 0)$ because at every previous order one tensor was already eliminated. The max ensures that dependencies won't get counted negatively. The sum over $\boldsymbol{\alpha}$ is easy to calculate:

$$\sum_{|\boldsymbol{\alpha}|=T} P(\boldsymbol{\alpha}) = \sum_{\alpha_1+\cdots+\alpha_L=T} \frac{[d]}{\alpha_1!}^{\alpha_1} \cdots \frac{[d]}{\alpha_L!}^{\alpha_L} = \frac{[d]}{T!}^{T} * \cdots * \frac{[d]}{T!}^{T} = \frac{[Ld]}{T!}^{T} .$$

The end result for the number of tensor structures in d dimensions with L loops and the powers of the terms in the F_j up to φ is

$$\begin{aligned} \frac{[2d+1]}{(\varphi+1)!}^{\varphi+1} &+ \sum_{k=0}^{\infty} (-1)^k Q_k \\ &= \frac{[2d+1]}{(\varphi+1)!}^{\varphi+1} + \sum_{k=0}^{\infty} (-1)^k \frac{[Ld]^{\varphi-2k}}{(\varphi-2k)!} \sum_{|\mathbf{a}|=k+1} \max\left(H(\mathbf{a}) - k, 0\right). \end{aligned} \quad (7)$$

4 Results

In the range $L = 2$, $1 \leq \varphi \leq 4$, $3 \leq d \leq 6$ the formula agrees with numerical results. Since the computation time and memory requirements grow fast with φ, higher values have not been checked. On the available hardware solving for $\varphi = 4$, $d = 6$ took 200 minutes and had to be fine tuned such that it was accurate enough, but would not require more memory than available.

Higher values of φ or d were therefore impossible to check. For $d = 2$ none of the numerical values agree with the formula, which is to be expected since it is easy to imagine that more dependencies of the strange type will appear.

That the type of dependencies that appear in the case of $d = 2$ may appear for higher φ and/or d is very much possible, although the numerical evidence does not show it. In that case the given expression will give a too high number of tensor structures than in reality. Since the calculations take a long time it is improbable that this technique will be feasible in short term. On the long term when it will become possible to efficiently do these decompositions it will also be feasible to calculate the number of tensorstructures in more extreme cases.

Appendix

A Notation

There is no universal notation, therefore I place a reference for convenience.

notation/term	meaning
d	Number of physical dimensions considered.
"$1=1$"	$1=\sum_j F_j D_j$
q^μ, l_j^μ	loopmomenta; integration variables.
D_j	$(q+p_j)^2 - m_j^2$, where q could also be a sum of loop momenta.
φ	$\max_j(\partial F_j)$ = maximum degree of the l_i's in the F_j's.
L	Number of different loop momenta
$\mathcal{L}$	$L(L+1)/2$, the number of different inner products with L l_i
$(a,b,c)(\alpha,\beta)$	Shorthand for $l_1^{2a}(l_1 \cdot l_2)^b l_2^{2b} l_1^{\mu_1} \cdots l_1^{\mu_\alpha} l_2^{\nu_1} \cdots l_2^{\nu_\beta}$.
$\mathbf{a}$	The (a,b,c) above, but for general $L \geq 2$.
$\boldsymbol{\alpha}$	The (α,β) above, but for general $L \geq 2$
$H(\boldsymbol{\nu})$	Number of non-zero elements of $\boldsymbol{\nu}$
$\lvert\boldsymbol{\nu}\rvert$	$=\sum_j \nu_j$
$[d]^\varphi$	$= d(d+1)\cdots(d+\varphi-1) = \Gamma(d+\varphi)/\Gamma(d)$

B Number Of Elements In A Symmetric Tensor

Here I discuss finding the number of independent elements of a tensor of the form $l^{\mu_1} l^{\mu_2} \cdots l^{\mu_k}$, where l is a d-dimensional tensor. Recognise that this problem is the same as determining how many possibilities there are to put n indistinguishable dots in d containers: The containers correspond to the values the μ_j can have and the amount of dots in container k is how many μ_j have the value k. Because the tensor is symmetric the dots are indistinguishable, so only the *number* of μ's that are 1 or 2, etc. are important. So for example

1	2	3	$\cdots$	d
$\cdot$	$\therefore$		$\cdots$	$\cdot$

By only looking at the bottom row, we see that this problem is equivalent to taking $n+d-1$ objects (dots ánd vertical lines) and then choosing n of those elements to be vertical lines which can be done in $\binom{n+d-1}{n}$ ways. $\square$

Note that $\binom{n+d-1}{n} = d(d+1)\cdots(d+n-1)/n!$ so if I define $[d]^k \equiv \Gamma(d+k)/\Gamma(d) = d(d+1)\cdots(d+k-1)$ then this type of binomial coefficient can be written as $[d]^k/k!$, a much cleaner notation.

Given two tensors l_1^μ, l_2^μ, the number of independent tensor structures in the expression $l_1^{\mu_1} \cdots l_1^{\mu_m} l_2^{\nu_1} \cdots l_2^{\nu_n}$ in d dimensions is $([d]^m/m!)([d]^n/n!)$. Therefore the number of tensors of a given rank k is this product, summed over all integers $n, m \geq 0$ such that their sum is k

$$C_k^d \equiv \sum_{n+m=k} \frac{[d]^n}{n!}\frac{[d]^m}{m!} = \left(\frac{[d]^n}{n!}\right) * \left(\frac{[d]^n}{n!}\right)(k),$$

with $*$ denoting convolution. Taking the generating function[6] of the sequence C_k^d for fixed d gives

$$\sum_{k=0}^{\infty} C_k^d z^k = \left(\sum_{k=0}^{\infty} \frac{[d]^k}{k!} z^k\right)^2 = (f_d(z))^2. \tag{8}$$

The function $f_d(z)$ can be seen to equal $(1-z)^{-d}$, the easiest by using the identity

$$\frac{[d]^k}{k!} = \binom{d+k-1}{k} = (-1)^k \binom{-d}{k}$$

such that by the binomial theorem

$$\sum_{k=0}^{\infty} \frac{[d]^k}{k!} z^k = \sum_{k=0}^{\infty} \binom{-d}{k} (-z)^k = (1-z)^{-d}$$

$$\sum_{k=0}^{\infty} C_k^d z^k = (1-z)^{-2d} = f_{2d}(z) = \sum_{k=0}^{\infty} \frac{[2d]^k}{k!} z^k$$

$$\text{so } \frac{[d]^m}{m!} * \frac{[d]^m}{m!} = \frac{[2d]^m}{m!}. \tag{9}$$

Multiple convolutions give powers of f, therefore with L convolutions with itself

$$\underbrace{\frac{[d]^n}{n!} * \frac{[d]^n}{n!} * \cdots * \frac{[d]^n}{n!}}_{L} = \frac{[Ld]^n}{n!}.$$

There is also a closed form expression for the sum sequence $S_k^d = \sum_{j=0}^{k} C_j^d$, by again using generating functions. If $a_k = \sum_{j=0}^{k} b_j = b_k * 1$, where 1 denotes the (one-sided) sequence of ones then

$$\sum_{k=0}^{\infty} a_k z^k = \sum_{k=0}^{\infty} (b_k * 1) z^k = \left(\sum_{k=0}^{\infty} b_k z^k\right)\left(\sum_{k=0}^{\infty} z^k\right) = \frac{1}{1-z} \sum_{k=0}^{\infty} b_k z^k$$

So with $a_k = S_k^d$ and $b_k = C_k^d$, it follows that

$$\sum_{k=0}^{\infty} S_k^d z^k = (1-z)^{-1} f_d(z) = (1-z)^{-(d+1)} = \sum_{k=0}^{\infty} \frac{[d+1]^k}{k!} z^k.$$

References

[1] Ioannis Malamos: *Reduction of one and two loop Amplitudes at the Integrand level*

[2] Ronald H. P. Kleiss, Ioannis Malamos, Costas G. Papadopoulos, Rob Verheyen: *Counting to one: reducibility of one- and two-loop amplitudes at the integrand level*

[6]By the nature of generating functions one does not care about convergence, but all identities here hold for $|z| < 1$.