Mathematics for Structure Functions

- Introduction.
- Harmonic Sums.
- Harmonic Polylogarithms.
- Mellin Transforms.
- Difference equations.
- Status.

Introduction

DIS is a good testing ground for QCD.
HERA has measured the structure functions very accurately. It is going to be even more accurate.
LHC will need very accurate PDF's if we want to measure the properties of the Higgs (assuming they will observe it).
Short (incomplete) historical overview concerning structure functions.

- The one loop corrections to anomalous dimensions (1972).
- The one loop corrections coefficient functions (1974).
- The two loop anomalous dimensions (1979).
- σ_L/σ_T in two loops (1988).
- All two loop coefficient functions (1991).
- A number of moments of three loop anomalous moments and coefficient functions (1996).

Methods used are the 'standard way' of integration with Feynman parameters, AND a method that computes all Mellin moments and then reconstructs the structure functions from them.
When either all even or all odd Mellin moments are known one can reconstruct the original function of $x = Q{\cdot}Q/2P{\cdot}Q$.
We are trying to compute the three loop anomalous dimensions and the three loop coefficient functions with the technique of the Mellin moments.
This is a relatively poorly developed technique and much mathematics needed to be explored before the actual calculation could be started.

Harmonic sums

The functions in which the moments can be expressed are called Harmonic sums. They are defined by

$$
\begin{aligned}
S_m(n) &= \sum_{i=1}^{n} \frac{1}{i^m} && m > 0 \\
&= \sum_{i=1}^{n} \frac{(-1)^i}{i^m} && m < 0 \\
S_{m_1,\cdots,m_k}(n) &= \sum_{i=1}^{n} \frac{1}{i^{m_1}} S_{m_2,\cdots,m_k}(i) && m_1 > 0 \\
&= \sum_{i=1}^{n} \frac{(-1)^i}{i^{m_1}} S_{m_2,\cdots,m_k}(i) && m_1 < 0
\end{aligned}
$$

The parameters m_i are called the indices and n is the argument. The weight of a harmonic sum is the sum of the absolute values of the indices. Example of a weight 9 sum:

$$S_{1,-5,3}(n) = \sum_{i=1}^{n} \frac{1}{i} \sum_{j=1}^{i} \frac{(-1)^j}{j^5} \sum_{k=1}^{j} \frac{1}{k^3}$$

Alternative notation: An index that is zero which is followed by an index that is nonzero indicates that one should be added to the absolute value of the nonzero index. Hence:

$$S_{1,0,0,0,0,-1,0,0,1}(n) = S_{1,-5,3}(n)$$

In this notation it is easy to see how many sums there are for a given weight: Each index can be $\pm 1, 0$ but the last index can be only ± 1. Hence for weight w there are $2\ 3^{w-1}$ harmonic sums.

These sums form an algebra:

$$\begin{aligned} S_{\vec{m}}(n)S_{\vec{p}}(n) &= \sum_{\vec{q}} c^{\vec{q}}_{\vec{m}\vec{p}} S_{\vec{q}}(n) \\ w_q &= w_m + w_p \end{aligned}$$

It allows writing expressions in a unique way, and also it allows doing more complicated sums immediately. Simple example:

$$\begin{aligned} \sum_{i=1}^{n} \frac{S_1(i)S_{-2}(i)}{i^2} &= \sum_{i=1}^{n} \frac{1}{i^2}(S_{1,-2}(i) + S_{-2,1}(i) - S_{-3}(i)) \\ &= S_{2,1,-2}(n) + S_{2,-2,1}(n) - S_{2,-3}(n) \end{aligned}$$

There are whole classes of sums that can be done to any level of complexity:

$$\Sigma_{i=1}^{n} \frac{S_{\vec{p}}(i)S_{\vec{q}}(n-i)}{i^m}$$

$$\Sigma_{i=1}^{n} (-1)^i \frac{S_{\vec{p}}(i)S_{\vec{q}}(n-i)}{i^m}$$

$$\Sigma_{i=1}^{n} (-1)^i \binom{n}{i} \frac{S_{\vec{p}}(i)}{i^m}$$

$$\Sigma_{i=1}^{n} (-1)^i \binom{n}{i} \frac{S_{\vec{p}}(i)S_{\vec{q}}(n-i)}{i^m}.$$

In general multiparameter sums are potentially too difficult, but if we can write nested sums as a nesting of single parameter sums we can usually do the sums. The above sums are all single parameter sums. This means that the only parameters are the summation parameter and the upper bound.

Some of the ways we get harmonic sums into our formulas:

$$\begin{aligned}\Gamma(-n+\epsilon) &= \frac{(-1)^n}{\epsilon n!}\Gamma(1+\epsilon)(1+S_1(n)\epsilon \\ &\quad +S_{1,1}(n)\epsilon^2+S_{1,1,1}(n)\epsilon^3 \\ &\quad +S_{1,1,1,1}(n)\epsilon^4+\cdots)\end{aligned}$$

and slightly messier on the positive side.

$$\sum_{i=1}^{n}(-1)^i\binom{n}{i}\frac{1}{n^3} = -S_{1,1,1}(n)$$

Harmonic Polylogarithms

Already at the two loop calculations (Zijlstra and van Neerven) it is clear that the polylogarithms will not be sufficient when continuing to higher orders. Also the Nielsen generalized polylogarithms are not sufficient. Hence we define

$$
\begin{aligned}
H(0;x) &= \ln x \ , \\
H(1;x) &= \int_0^x \frac{dx'}{1-x'} = -\ln(1-x) \ , \\
H(-1;x) &= \int_0^x \frac{dx'}{1+x'} = \ln(1+x) \ . \\
H(a,m_1,\cdots,m_w;x) &= \int_0^x dx' \ f(a;x') \ H(\vec{m}_w;x') \ , \\
f(0;x) &= \frac{1}{x} \ , \\
f(1;x) &= \frac{1}{1-x} \ , \\
f(-1;x) &= \frac{1}{1+x} \ . \\
H(\vec{0}_w;x) &= \frac{1}{w!}\ln^w x \ ,
\end{aligned}
$$

We can define weights and a more compressed notation in the same way as for the harmonic sums.

Why are these functions so special?

- They are a generalization of Nielsen's polylogarithms. They include a number of integrals that cannot be expressed in terms of NGP's and that are most likely needed for three loop results.
- The H-functions with identical arguments form an algebra. The algebraic relationships can be readily obtained. Hence it is easy to bring an expression into a unique form.
- The Taylor expansion of the H-functions have coefficients that are combinations of harmonic sums. The functions can be expanded in nearly the whole complex plane. Only the areas around $x = \pm i$ are a bit problematic in terms of convergence.

Mellin Transforms

The Mellin transform is defined by

$$M(f(x)) = \int_0^1 dx\ x^{N-1} f(x)$$

For the harmonic polylogarithms the Mellin transforms can be expressed in terms of harmonic sums in N and harmonic sums in infinity.

Recent work allows expressing these sums in infinity into a minimal set of fundamental constants and products of these. For weight 6 there are 8 such constants.

It becomes even better when we look at functions of the types

$$M\left(\frac{H(x)}{(1-x)}\right)$$
$$M\left(\frac{H(x)}{(1+x)}\right)$$

There are $2\ 3^w$ of such functions for each weight w. This is equal to the number of harmonic sums of weight $w+1$. And they do give harmonic sums with weights up to $w+1$. And for each such function there is only one such sum with the highest weight! There are additional lower weight sums.

There is a one to one correspondence between the harmonic polylogarithm (plus its additional denominator $1 \pm x$) and the highest weight harmonic sum.
In principle the answer may also contain terms in $(-1)^N$, and hence one would expect that there cannot be a one to one correspondence. Instead of using 1 and $(-1)^N$ we use $(1 \pm (-1)^N)/2$ and then it is easy to see what happens. On the 'sum'-side we have to specify whether the function is for even values of N or for odd values of N. This takes away the ambiguity.
This gives immediately a simple procedure to do the inverse Mellin transform.

- Given a HS, construct the single term with a HP with the highest weight and the denominator $1/(1 \pm x)$. Subtract and add.
- Do a Mellin transform on the subtracted term. This will create a term that cancels the original HS and gives also terms with lower weight HS's.
- Keep repeating this procedure until there are no more terms with HS's.
- The remaining constant terms will get a factor $\delta(1 - x)$.

```
#include nndecl.h
#call htables(6)
S y1,...,y8;
CF ln;
.global
Format 50;
L F = Mel*H(R(1,-2,1,0),x)/[1-x];
#call hmellin(H,x,S,N)
  ..... output processing
B S,sign_;
Print +f;
.end
```

```
   F =
       + S(R(1,-2),N) * (  - 1/2*z2*ln2 + z3 )

       + S(R(1,-2,-1,2),N) * ( 1 )

       + S(R(1,1),N) * (  - z2*ln2^2 - 13/40*z2^2
          + 1/6*ln2^4 + 4*li4half )

       + S(R(1,2),N) * ( 1/2*z2*ln2 - z3 )

       + 7/16*z2*z3*ln2 + 1/8*z2*ln2^4 + 3*z2*
         li4half - 3/4*z2^2*ln2^2 - 31/28*z2^3 -
         15/16*z3^2 + 7/2*s6;
```

```
#include nndecl.h
#call htables(6)
S y1,...,y8;
CF ln;
.global
Format 50;
L F = Mel*H(R(1,-2,1,0),x)/[1+x];
#call hmellin(H,x,S,N)
  ..... output processing
B S,sign_;
Print +f;
.end
```

```
   F =
       + sign_(N)*S(R(-1,-2),N) * ( 1/2*z2*ln2 -
         z3 )

       + sign_(N)*S(R(-1,-2,-1,2),N) * (  - 1 )

       + sign_(N)*S(R(-1,1),N) * ( z2*ln2^2 + 13/
         40*z2^2 - 1/6*ln2^4 - 4*li4half )

       + sign_(N)*S(R(-1,2),N) * (  - 1/2*z2*ln2
          + z3 )

       + sign_(N) * (  - 1/2*z2*z3*ln2 - 1/6*z2*
         ln2^4 - 45/16*z2^2*ln2^2 - 3277/560*z2^3
          - 37/4*z3^2 - 547/32*z5*ln2 + 16*ln2*
         li5half + 2*ln2^2*li4half + 1/60*ln2^6
          + 48*li6half + 26*s6 );
```

```
#include nndecl.h
#call htables(6)
#call tables(6)
S tsign;
CF Sacc,Sfuntemp;
.global
L F = S(R(-1,1,-2,2),N);
#call invmel(S,N,Ha,x)
id Ha(R,x) = 1;
.sort:Preliminary;
   ....
*   Some code for normalizing the output
   ....
B H,[1-x],[1+x],sign_,Even,Odd,delta_;
Print +f;
.end
```

```
F =
    + delta_(1 - x)*Even(N) * ( 19/8*z2*z3*ln2
       + 1/3*z2*ln2^4 + 4*z2*li4half - 13/20*
      z2^2*ln2^2 - 3571/1680*z2^3 - 7/12*z3*
      ln2^3 - 217/128*z3^2 - 103/32*z5*ln2 + 4
      *ln2*li5half - 1/45*ln2^6 + 8*li6half +
      15/4*s6 )

    + Even(N)*[1-x]^-1 * ( 5/16*z2*z3 + 2/3*z2
      *ln2^3 - 7/4*z3*ln2^2 + 103/32*z5 - 4*
      ln2*li4half - 2/15*ln2^5 - 4*li5half )

    + Even(N)*[1+x]^-1 * ( 5/16*z2*z3 + 2/3*z2
      *ln2^3 - 7/4*z3*ln2^2 + 103/32*z5 - 4*
      ln2*li4half - 2/15*ln2^5 - 4*li5half )

    + Even(N)*H(R(-1),x)*[1-x]^-1 * (  - z2*
      ln2^2 - 51/40*z2^2 + 7/2*z3*ln2 + 1/6*
      ln2^4 + 4*li4half )

    + Even(N)*H(R(-1,-1),x)*[1-x]^-1 * (  - 2*
      z3 )

    + Even(N)*H(R(-1,-1,2,0),x)*[1-x]^-1 * (
       - 1 )

    + Even(N)*H(R(1),x)*[1+x]^-1 * ( z2*ln2^2
       + 51/40*z2^2 - 7/2*z3*ln2 - 1/6*ln2^4
       - 4*li4half )
```

```
+ delta_(1 - x)*Odd(N) * ( 19/8*z2*z3*ln2
   + 1/3*z2*ln2^4 + 4*z2*li4half - 13/20*
  z2^2*ln2^2 - 3571/1680*z2^3 - 7/12*z3*
  ln2^3 - 217/128*z3^2 - 103/32*z5*ln2 + 4
  *ln2*li5half - 1/45*ln2^6 + 8*li6half +
  15/4*s6 )

+ Odd(N)*[1-x]^-1 * ( 5/16*z2*z3 + 2/3*z2*
  ln2^3 - 7/4*z3*ln2^2 + 103/32*z5 - 4*ln2
  *li4half - 2/15*ln2^5 - 4*li5half )

+ Odd(N)*[1+x]^-1 * (  - 5/16*z2*z3 - 2/3*
  z2*ln2^3 + 7/4*z3*ln2^2 - 103/32*z5 + 4*
  ln2*li4half + 2/15*ln2^5 + 4*li5half )

+ Odd(N)*H(R(-1),x)*[1-x]^-1 * (  - z2*
  ln2^2 - 51/40*z2^2 + 7/2*z3*ln2 + 1/6*
  ln2^4 + 4*li4half )

+ Odd(N)*H(R(-1,-1),x)*[1-x]^-1 * (  - 2*
  z3 )

+ Odd(N)*H(R(-1,-1,2,0),x)*[1-x]^-1 * (
   - 1 )

+ Odd(N)*H(R(1),x)*[1+x]^-1 * (  - z2*
  ln2^2 - 51/40*z2^2 + 7/2*z3*ln2 + 1/6*
  ln2^4 + 4*li4half );
```

Difference equations

The formula $F(N)$ for a given diagram can often not be determined directly, but we can find an equation of the type

$$\begin{aligned} A_0(N)F(N)+A_1(N)F(N-1)+ & \\ \cdots+A_n(N)F(N-n)+G(N) &= 0 \end{aligned} \tag{1}$$

We call this an n-th order difference equation. A first order difference equation is also called a recursion. Those can be solved easily:

$$\begin{aligned} F(N) &= (-1)^N \frac{\Pi_{j=1}^N A_1(j)}{\Pi_{j=1}^N A_0(j)} F(0) \\ &+ \sum_{i=1}^N (-1)^{N-i} \frac{\Pi_{j=i+1}^N A_1(j)}{\Pi_{j=i}^N A_0(j)} G(i)\,. \end{aligned} \tag{2}$$

When the A_i fulfill some simple conditions the products can be written as Gamma functions. These can be expanded in epsilon, giving harmonic sums. The function G usually also contains harmonic sums. The whole can then be summed and gives new harmonic sums. We define Basic Building Blocks (BBB) as diagrams that hve a P flow through only a single internal line. The idea is to express all diagrams into such BBB's. At the two loop level there are two such BBB's and each can be written as a first order difference equation. At the three loop level we have 17 BBB's of which 7 are two loop graphs with a one loop insertion.

- Four give a first order equation.
- Twelve give a second order equation.
- One gives a third order equation (thus far).

Some of the second order equations can be solved by 'constructive' methods (writing the solution as nested sums in some type of expansion in epsilon). But we could not do all! HOWEVER
There is a powerful method that solves all equations, provided a solution exists inside the space of functions we consider. This space is finite, although it may contain a few thousand elements.
The coefficients A_i are polynomials in N and ϵ, and G is given by simpler diagrams. Hence G is given in terms of harmonic sums. The method is now:

- Divide the equation by N^q in which q is the highest power of N occurring in the coefficients A_i.
- Synchronize the arguments of the sums in G and the denominators.
- Write $S_{\vec{m}}(N)/N^p = S_{p,\vec{m}}(N) - S_{p,\vec{m}}(N-1)$ and similar forms when powers of -1 are involved.
- Assume that $F(N) = \sum c_{i,n,\vec{m}} \epsilon^i S_{\vec{m}}(N-n)$. We have to include powers of ζ_3, ζ_4 and ζ_5 as well, but let us ignore that here. We have to guess a bit what range of values we need for i and n.
- We also bring the $A_j(N)F(N-j)$ terms into the standard form with just a single S-sum without denominators.
- This defines a large number of equations in the $c_{i,n,\vec{m}}$ which we solve. If the range in i and n was large enough we should find a solution. The maximum weight we use defines (amoung others) to what power in ϵ our answer can be correct.
- There will still be a few free parameters left, as we have not fixed the boundary conditions yet. For an n-th order equation

we will need n values. These can be computed with the Mincer program, as they concern fixed values of N.

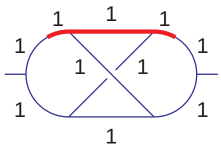
- The boundary conditions give a few more equations, and these fix all remaining constants. This then determines the function F.

This method is actually faster than the constructive method (when we could compare). For weight 6 there are 486 sums for each value of n. Choosing several values of n this leads quickly to a few thousand parameters to be determined. Fortunately FORM could handle this rather quickly.

Status

We have derived all the equations for the Basic Building Blocks. The seven BBB's for two loop diagrams with a one loop insertion have been solved. At the moment we are setting up the programs for the three loop BBB's. Because the function G contains simpler diagrams one has to do this in a well defined order. This is much work.
The most complicated diagrams are NO_{22} which gives a third order equation, and BE_{88} which gives an second order equation with about 200000 terms (we may be able to simplify this as we have found some new equations since then).
The reduction for composite diagrams into basic building blocks follows a similar scheme. Take for instance the diagram

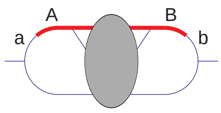
$$NO_{13} \; =$$



$$= \int dp_1^D dp_2^D dp_3^D \frac{1}{p_1^2 (P+p_1)^2 (P+p_2)^2} \times \frac{1}{(P+p_3)^2 p_3^2 p_4^2 p_5^2 p_6^2 p_7^2 p_8^2} \tag{3}$$

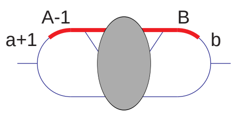
We derived a general formula for such diagrams that is relatively simple:

$$(N+a+b)$$

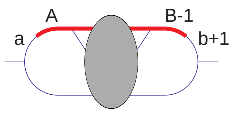


$$=$$

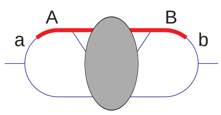
$$a$$



$$+b$$



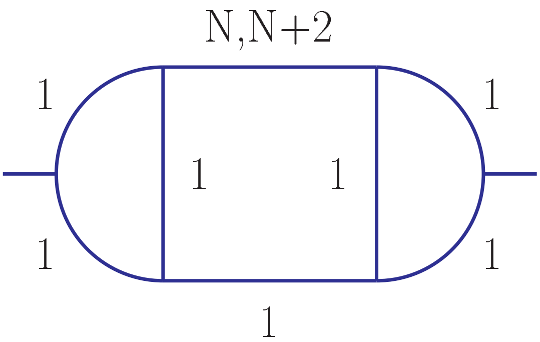
$$-(N-1+n)\frac{2P{\cdot}Q}{Q{\cdot}Q} \tag{4}$$



This allows us to express the NO_{13} diagram into a simpler diagram by means of a single recursion. Of course one can only solve this recursion once the simpler diagrams have been evaluated.
The main problem is now to break down all the composite diagrams into simpler diagrams and eventually into BBB's. This is just an enormous amount of work. After that it has to be programmed and tested which is also an enormous amount of work.

An example

Here we have an example of a rather nontrivial integral that was solved using a second order difference equation. The integral is given by the diagram



This integral can be written as

$$
\begin{aligned}
LA_{22}(2,N) &= \int dp_1^D dp_2^D dp_3^D \frac{(2P\cdot p_2)^N}{p_1^2(p_2^2)^{N+2}p_3^2p_4^2p_5^2p_6^2p_7^2p_8^2} \\
&= \frac{2}{3\epsilon^3}(\frac{1}{N+1}(S_2(N+1)-S_1(N+1) \\
&\quad -S_{1,1}(N+1))+\frac{1}{N+2}S_1(N+2)-\frac{1}{(N+2)^2}) \\
&\quad +\frac{1}{\epsilon^2}(\frac{1}{N+1}(+\frac{16}{3}S_1(N+1)+\frac{2}{3}S_{1,1}(N+1) \\
&\quad -4S_{1,1,1}(N+1)+4S_{1,2}(N+1)-\frac{2}{3}S_2(N+1) \\
&\quad +\frac{14}{3}S_{2,1}(N+1)-\frac{14}{3}S_3(N+1)) \\
&\quad +\frac{1}{N+2}(-\frac{16}{3}S_1(N+2)+\frac{10}{3}S_{1,1}(N+2)) \\
&\quad -\frac{10}{3}S_2(N+2)+\frac{1}{(N+2)^2}(+\frac{16}{3} \\
&\quad -\frac{14}{3}S_1(N+2))+\frac{14}{3}\frac{1}{(N+2)^3})
\end{aligned}
$$

$$
\begin{aligned}
&+\frac{1}{\epsilon}(\frac{1}{N+1}(-\frac{56}{3}S_1(N+1)+\frac{56}{3}S_{1,1}(N+1)\\
&+\frac{34}{3}S_{1,1,1}(N+1)-\frac{50}{3}S_{1,1,1,1}(N+1)\\
&+16S_{1,1,2}(N+1)-12S_{1,2}(N+1)\\
&+18S_{1,2,1}(N+1)-16S_{1,3}(N+1)\\
&-\frac{56}{3}S_2(N+1)-\frac{44}{3}S_{2,1}(N+1)\\
&+\frac{58}{3}S_{2,1,1}(N+1)-\frac{56}{3}S_{2,2}(N+1)\\
&+\frac{50}{3}S_3(N+1)-\frac{68}{3}S_{3,1}(N+1)\\
&+\frac{62}{3}S_4(N+1))+\frac{1}{N+2}(\\
&+\frac{56}{3}S_1(N+2)-\frac{80}{3}S_{1,1}(N+2)\\
&+\frac{38}{3}S_{1,1,1}(N+2)-12S_{1,2}(N+2)\\
&+\frac{80}{3}S_2(N+2)-\frac{40}{3}S_{2,1}(N+2)\\
&+\frac{34}{3}S_3(N+2)))+\frac{1}{(N+2)^2}(\\
&-\frac{56}{3}+\frac{112}{3}S_1(N+2)\\
&-\frac{58}{3}S_{1,1}(N+2)+\frac{56}{3}S_2(N+2)\\
&)+\frac{1}{(N+2)^3}(-\frac{112}{3}\\
&+\frac{68}{3}S_1(N+2))-\frac{62}{3}\frac{1}{(N+2)^4})\\
&+20\zeta_5+\frac{1}{N+1}(+\frac{128}{3}S_1(N+1)
\end{aligned}
$$

$$
\begin{aligned}
&+\frac{44}{3}S_1(N+1)\zeta_3-88S_{1,1}(N+1)\\
&+\frac{20}{3}S_{1,1}(N+1)\zeta_3+\frac{160}{3}S_{1,1,1}(N+1)\\
&+\frac{170}{3}S_{1,1,1,1}(N+1)-60S_{1,1,1,1,1}(N+1)\\
&+56S_{1,1,1,2}(N+1)-52S_{1,1,2}(N+1)\\
&+62S_{1,1,2,1}(N+1)-56S_{1,1,3}(N+1)\\
&-48S_{1,2}(N+1)-64S_{1,2,1}(N+1)\\
&+66S_{1,2,1,1}(N+1)-60S_{1,2,2}(N+1)\\
&+56S_{1,3}(N+1)-68S_{1,3,1}(N+1)\\
&+56S_{1,4}(N+1)+88S_2(N+1)\\
&-\frac{44}{3}S_2(N+1)\zeta_3-\frac{152}{3}S_{2,1}(N+1)\\
&-\frac{220}{3}S_{2,1,1}(N+1)+\frac{206}{3}S_{2,1,1,1}(N+1)\\
&-68S_{2,1,2}(N+1)+\frac{212}{3}S_{2,2}(N+1)\\
&-\frac{212}{3}S_{2,2,1}(N+1)+\frac{188}{3}S_{2,3}(N+1)\\
&+\frac{104}{3}S_3(N+1)+\frac{272}{3}S_{3,1}(N+1)\\
&-\frac{244}{3}S_{3,1,1}(N+1)+\frac{236}{3}S_{3,2}(N+1)\\
&-\frac{266}{3}S_4(N+1)+\frac{272}{3}S_{4,1}(N+1)\\
&-\frac{230}{3}S_5(N+1))-8S_2(N+1)\zeta_3\\
&-4S_{2,1,2}(N+1)+4S_{3,1,1}(N+1)\\
&+\frac{1}{N+2}(-\frac{128}{3}S_1(N+2)
\end{aligned}
$$

$$
\begin{aligned}
&-\frac{20}{3}S_1(N+2)\zeta_3+\frac{280}{3}S_{1,1}(N+2)\\
&-\frac{304}{3}S_{1,1,1}(N+2)+\frac{130}{3}S_{1,1,1,1}(N+2)\\
&-40S_{1,1,2}(N+2)+96S_{1,2}(N+2)\\
&-44S_{1,2,1}(N+2)+40S_{1,3}(N+2)\\
&-\frac{280}{3}S_2(N+2)+\frac{320}{3}S_{2,1}(N+2)\\
&-\frac{140}{3}S_{2,1,1}(N+2)+\frac{124}{3}S_{2,2}(N+2)\\
&-\frac{272}{3}S_3(N+2)+\frac{136}{3}S_{3,1}(N+2)\\
&-\frac{106}{3}S_4(N+2))+\frac{1}{(N+2)^2}(+\frac{128}{3}+\frac{20}{3}\zeta_3\\
&-\frac{392}{3}S_1(N+2)+\frac{464}{3}S_{1,1}(N+2)\\
&-\frac{206}{3}S_{1,1,1}(N+2)+64S_{1,2}(N+2)\\
&-\frac{448}{3}S_2(N+2)+\frac{212}{3}S_{2,1}(N+2)\\
&-\frac{188}{3}S_3(N+2))+\frac{1}{(N+2)^3}(\\
&+\frac{392}{3}-\frac{544}{3}S_1(N+2)+\frac{256}{3}S_{1,1}(N+2)\\
&-\frac{236}{3}S_2(N+2))+\frac{1}{(N+2)^4}(+\frac{496}{3}\\
&-\frac{272}{3}S_1(N+2))+\frac{230}{3}\frac{1}{(N+2)^5})
\end{aligned}
$$