|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| f(u(x))’ | f(u)’u(x)’ | f(u)’’ | 1f(u)’’’ | 1f(u)’’’’ | 1f(u)’’’’’ |
|  |  | f(u)’u(x)’’ | 3f(u)’’ | 6f(u)’’’ | 10f(u)’’’’ |
|  |  |  | 1f(u)’u(x)’’’ | 4f(u)’’u(x)’u(x)’’’  +3f(u)’’ | 10f(u)’’’  +15f(u)’’’ |
|  |  |  |  | 1f(u)’u(x)’’’’ | 5f(u)’’u(x)’u(x)’’’’  +10f(u)’’u(x)’’u(x)’’’ |
|  |  |  |  |  | 1f(u)’u(x)’’’’’ |

我们先

=

一方面，通过替换序数得到它的一阶导：

又由=，知=

另一方面，用求导法则对其进行一次求导：(这时我们便努力构造像一样i从1取至j+1的求和，用于之后将各式符号内的内容合并在同一个求和符号里。)

=+

=+

=+

=

通式必须首先对此种挑战兼容并包，那么有：

=，即有：

.根据以下我们便可以直接推知下一行的表达式具体模样：

=+=+++···+++

=+=+++···++

又根据=+我们有=+0

则有=或=

又因=·

则====+=+=+=+

【根据三次两种=以及一次[n=i+1，k=3]】

即有=+

.同理：

=+=+++···+++

=+=+++···++

又根据=+我们有=0+

则有=或=

又因=

则=

不过到此为止，这个一斜排一斜排地向↗右上方的内地进发的方法会因此被淘汰：它到最后会涉及到相当复杂的越来越多个函数的乘积的求导，这不是我们想看到的，这正是我们想通过转化成其他求法来避免的。

.现在我们一心一意地专注于、、这种一横排一横排地向↓下方的内地进发的方法，因为它有规律可循而且其规律及其求法相对于偶数序数的做法来说比较简单：

由于==

所以==

又因=+

则= 现将此方程分成两部分，一部分类上一步，一部分为新object：

Part1: 由于

==+ 所以仿照它的步骤，有：

part1==+

Part2:，将它再分为两部分，一部分为part2.1: 仿照之前的有：part2.1==+

另一部分为Part2.2： 仿照之前的有：part2.2==3[+]

part2.1、Part2.2合并：part2= [+]=+

Part1、Part2合并：=++

.其实如你所见，不论是以斜排为基准向↗前进，还是以横排为基准向↓前进，最终都会遇到对多个函数的积求导，只不过横排的前进是函数的因式一点一点地变多，且每次求导只求一次，而斜排是一来就求n阶导，然后因式也一点一点变多、求导运算次数一点点地变少。

也就是说，其实由于各横排的从左往右第m个的组合，本身就构成了一斜排，所以当横排向下走的时候，最终将会有斜排的复杂程度。所以我们的横排也就写来到此为止。在这里还有个规律性的东西需要介绍一下：每个横排中非系数部分且除了之后求导求出来的数字之外的，纯组合数的衍生规律：例如：

=++

它的组合数们在下一次的操作=中，会如此地扮演如下角色：多项式1.→→→and→→→→→→ 多项式2.→→→and→→→→→→→ 多项式3.→→→and→→→→→→→

当这些多项式乘上各自之前的系数，再乘上各自式子中求导出来的系数，合并同类的导函数们后，将构成下一组带系数的组合数们。

This is the general rule ：对于单个目标，我们有：

Mode1:→→→ 即→

Mode2:→→→→→→ 即→