求和的算法

2018-07-16 23:29:25

一.动机

由于之前工作需要（使用Mellin-Barnes表象计算一些积分），遇到了求和的问题。专门去看了一本A=B的书。

传送门在此：

https://www.math.upenn.edu/~wilf/AeqB.html​

这其中介绍的一些计算求和的方法，我觉得比较有意思，有可能也会有人需要用到，所以在博客里分享一下。同时也记录一下以备说不定自己以后还会遇到。

这本书里的一些算法，我用Mathematica 11.1实现了一些，做了些尝试。在此博客的最后我会分享到Github中。

二.自动化的主要思路

由于我并不是这个专业的，所以可能遇到的问题并不多。从我遇到的问题来看，解决思路主要有两个。

  (1）化为方程组。（最好是，化为线性方程组。）

（2）正则化。

 可以用一个最简单的例子来介绍这两者。比方说，如果我们要求证明

[http://s9.sinaimg.cn/mw690/001mqGQAzy7m5nKbCbe38](http://album.sina.com.cn/pic/001mqGQAzy7m5nKbCbe38)

第一种方法是这么运作的：

1.当x=0时，左边=-1，右边=-1

2.当x=1时，左边=0，右边=0

3.当x=2时，左边=3，右边=3

所以等式左右相等。

这种题目估计是初中的题目，我不记得了。但是如果一个初中生在他的试卷上这么回答，恐怕是要打个大叉叉，然而为何可以这么去证明呢？因为等式左边减去右边一定是一个ax^2+bx+c。​

我们需要三个点来确定抛物线。带入三个x的值，左右两边相等，就可以知道左右两边确实相等了。

第二种方法可以这么看：

等式两边都是多项式，而多项式有一种规则的写法，比如说，升幂。写成升幂左右两边都成为了1+x^2

对于多项式，这种写法是唯一的，写出来后就可以比较是否相等了。

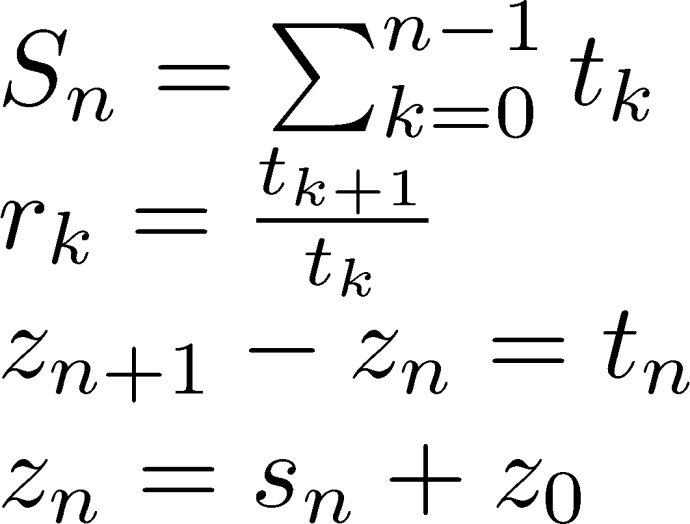
这回介绍的书中的几个求和算法，主要还是前一类型的方法。

 Gosper算法

Gosper算法主要是用来计算definitesums。也就是，有上下限的求和

[](http://album.sina.com.cn/pic/001mqGQAzy7m5o6cuKY18)*﻿不能改图片大小的吗？？？？*

为了方便，先约定

[](http://album.sina.com.cn/pic/001mqGQAzy7m5o6JB1N3b)

​

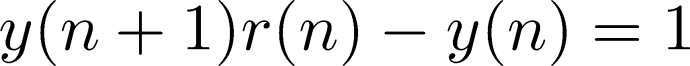
如果算出来z(n)，就可以知道S(n)了。而z(n)是满足一个递归关系式的，也就是上面的

z(n+1)-z(n)=t(n)

 假定，z(n)可以写为

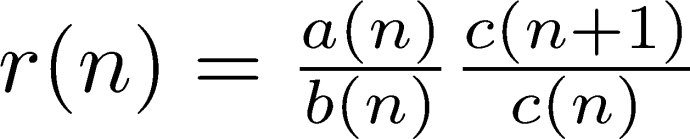
z(n)=y(n)t(n)​

这样的话，递归关系式成为了

[](http://album.sina.com.cn/pic/001mqGQAzy7m5o9s4u94d)

​

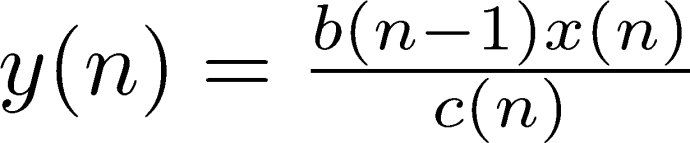
然后假定r(n)可以写为

[](http://album.sina.com.cn/pic/001mqGQAzy7m5o9ZtfI2a)

​

其中，a,b,c都是多项式，且当h为任何非负的整数时，GCD(a(n),b(n+h))=1。

我们后面可以看到，对于r(n)是一个所谓的”rational function”的时候，确实是可以写为这样的形式的。目前就认为它是可以这么写的。于是带入当上面的递归后，可以发现，带回到原方程

[](http://album.sina.com.cn/pic/001mqGQAzy7m5ocwrwL5b)

​

是一个解。其中x(n)是一个多项式，满足

[http://s15.sinaimg.cn/mw690/001mqGQAzy7m5od33oGfe](http://album.sina.com.cn/pic/001mqGQAzy7m5od33oGfe)

​

这里，a(n),b(n),c(n)都是已知的了，x(n)又是一个多项式，那么我们就可以假定

x(n)=c0+c1 n+ c2 n^2+...​

带回到原方程，解出来系数c0,c1,c2..

这里简单介绍一下degree和leading coefficient。Degree就是n的最大的次方的次方数（x(n)=0这种特殊情况除外，此时degree为负无穷。），leading coefficient 就是最大次方的系数。如果我们知道了x最大可能的degree，我们就可以方便的带回到方程里去计算系数了。

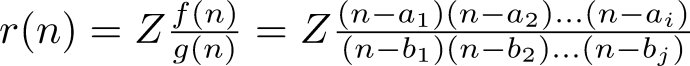
在得到x(n)后，我们就得到了y(n)。而t(n)又是已知的，这样的话，就得到了z(n)。从z(n)到s(n)我们只需要比较一下z(0)和s(0)即可。

**瞧！~**

**我们的求和问题，变成了一个求解方程组的问题！**

梳理下来可以发现，真正最关键的两步，一步就是如何把r(n)写成我们想要的形式。另一步就是知道x最大到n的多少次方。

为此，我先解释下，什么是rational function。所谓rational function，就是分式，分子分母都是多项式。比方说，我们的r(n)是一个rational function，那么它就可以写为

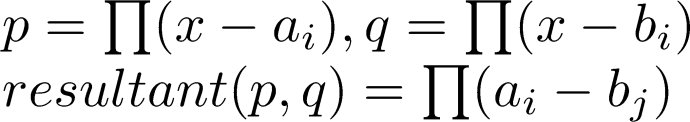
[](http://album.sina.com.cn/pic/001mqGQAzy7m5opQrr52b)

​

还记得我们要求a(n)和b(n+h)最大公因数是1。所以我们不能直接另a(n)=Zf(n)，b(n)=g(n),c(n)=1。我们要把它们的公因数找出来并“除掉”。

假定a1<a2<…<ai，b1<b2<…<bj。

我们先看，b1到bj中，是否包含任何a1到ai，接下来看b1+1,b2+1,…,bj+1中是否包含任何a1到ai，一直到b1+h>ai为止。我们需要找到所有这样的组合。有一个简单的方式来完成这件事情。需要用到一个叫做resultant的概念

[](http://album.sina.com.cn/pic/001mqGQAzy7m5oqBqcB2d)

​

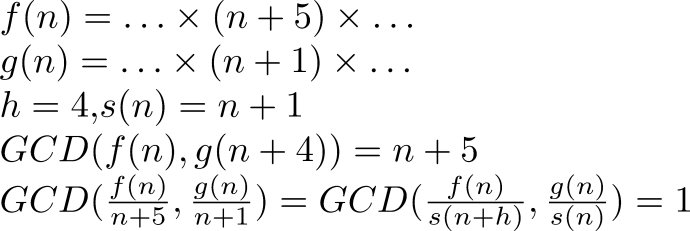
然后只需要以h为未知数，解方程

[http://s7.sinaimg.cn/mw690/001mqGQAzy7m5or99lQ16](http://album.sina.com.cn/pic/001mqGQAzy7m5or99lQ16)

​

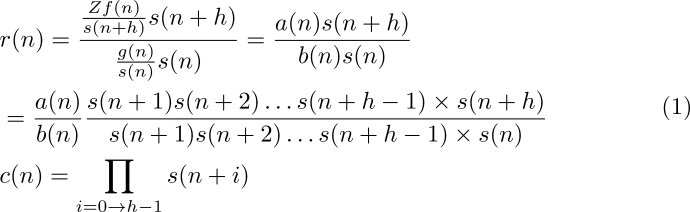
这个方程的所有h>=0的整数解，就是我们要找的，需要除掉的公因子了。

假定我们找到了这么一个公因子s(n)，我们要从分子上除掉s(n+h)，从分母上除掉s(n)。举个例子，假如说我们的f(n)和g(n+h)只有一个非1的公因子，所谓除掉，就是

[](http://album.sina.com.cn/pic/001mqGQAzy7m5ospXgr1f)

​

除掉后，就满足了我们要的条件，此时我们也同时知道c(n)是多少了：

[](http://album.sina.com.cn/pic/001mqGQAzy7m5otb27Uaa)

​

这样，我们就从r(n)算出来了a(n),b(n),c(n)。

第二个关键步骤就是解出x(n)的系数了。而这步的关键就是确定x(n)最大的度数。这就稍微有点繁琐，

这个方程

a(n)x(n+1)-b(n-1)x(n)​=c(n)

需要使每一个n^i的系数分别为0。

第一种情况：

如果a(n)和b(n-1)的degree不同或者leading coefficient不同。

那么x(n)的leadingcoefficient不会互相消掉。

deg(a(n))!=deg(b(n))或lc(a(n))!=lc(b(n))

因此，如果我们把x(n)的最大degree叫dmax，有

[http://s5.sinaimg.cn/mw690/001mqGQAzy7m5oO9H5a04](http://album.sina.com.cn/pic/001mqGQAzy7m5oO9H5a04)

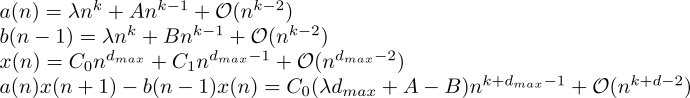
第二种情况:

最高次项消掉，第二高次项没消掉。那么等号左边的degree为dmax+deg(a(n))-1，右边为deg(c(n))因此

[http://s5.sinaimg.cn/mw690/001mqGQAzy7m5oOMWlC04](http://album.sina.com.cn/pic/001mqGQAzy7m5oOMWlC04)

第三种情况:

最高次，第二高次都消掉。可以显式写出来a,b和x，有

[](http://album.sina.com.cn/pic/001mqGQAzy7m5oQhdRi14)

因此

dmax=(B-A)/lambda

这样一来，我们就把所有的问题解决了。

总结一下Gosper算法：

我们需要计算s(n)=sum(t(k),{k,0,n})

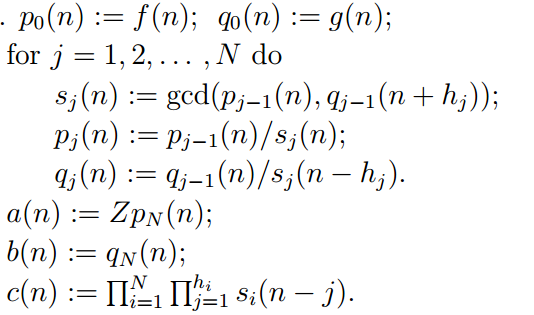
 （1）首先，我们计算r(n)=t(n+1)/t(n)

 （2） 我们由r(n)得到a(n),b(n),c(n):

取r(n)的分子为f(n)，分母为g(n)，另R(h)=Resultant (f(n),g(n+h))。

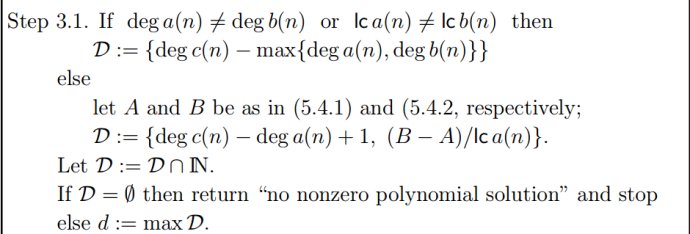
以h为未知数解方程R(h)=0。

对所有N个非负整数的解hj，做

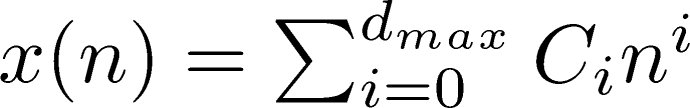
[](http://album.sina.com.cn/pic/001mqGQAzy7m5p2QN9Sc8)

​得到a(n)，b(n), c(n)。

（3）由a(n),b(n),c(n)计算dmax

[](http://album.sina.com.cn/pic/001mqGQAzy7m5p6KSBR13)

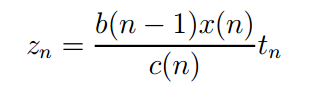
（4）然后将

[](http://album.sina.com.cn/pic/001mqGQAzy7m5p85tBmf8)

​

带入方程a(n)x(n+1)-b(n-1)x(n)​=c(n)，解出Ci，得到x(n)。

（5）然后由b,c,x得到z

[](http://album.sina.com.cn/pic/001mqGQAzy7m5peiDl224)

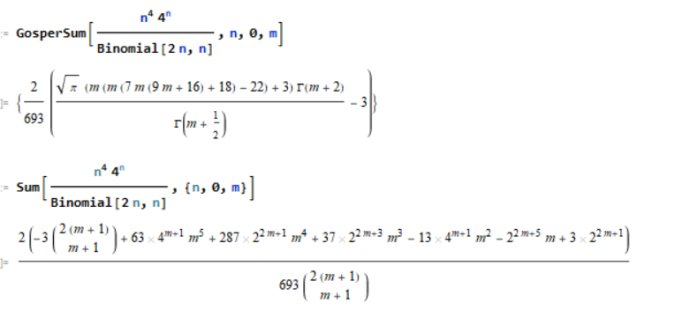
​

（6）最后，比较z(0)和s(0)，得到s(n)。

**结束​**

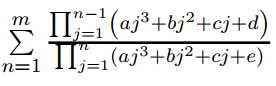
这个程序用mathematica实现了，并和mathematica本身的求和比较了一下：

（1）经常Gosper sum能比mathematica自带的求和，给出更简洁的形式，比如

[](http://album.sina.com.cn/pic/001mqGQAzy7m5pgi8pod6)

（2）一般来说mathematica更强大一些，不少不是Gosper summable的求和，是mathematica能计算的。

（3）但是也存在Gosper更强大一些的情况，比如

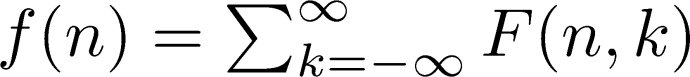
[](http://album.sina.com.cn/pic/001mqGQAzy7m5phGu3012)

​

虽然Gosper求和没法计算出最终结果，因为z(n)=t(n) b(n-1)x(n)/c(n)，这一步中的t(n)是算不出来的。但是Gosper确实能把求和计算掉（上式中的b,c,x都能计算出来）。这是mathematica做不到的。

Sister Mary Celine 算法

这个算法主要是用来计算indefinite sums，也就是

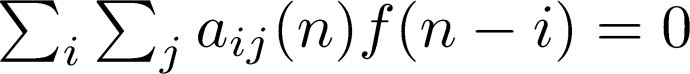
[](http://album.sina.com.cn/pic/001mqGQAzy7m5seeLU07c)

假定我们能找到递归关系

[http://s3.sinaimg.cn/mw690/001mqGQAzy7m5seUn5072](http://album.sina.com.cn/pic/001mqGQAzy7m5seUn5072)

​

那么上面这个求和问题就化为了一个递归问题

[](http://album.sina.com.cn/pic/001mqGQAzy7m5sfEzp10b)

而第一步往往比较简单。因为我们可以直接把aij(n)作为待定系数，并且带入到上面的方程求解待定系数。

虽然书里面给出了i,j求和的上限。不过更简单的实现方式是让用户手动输入一个上限，如果这个上限没找到递归关系式，那么提示“试试看更高阶”。

这个算法实现起来较为容易，就不累述了。

Creative Telescope算法

这个算法同样是计算indefinite sum。它同时借用了Gosper算法和SisterMary Celine算法。

为了方便描述我们定义算符

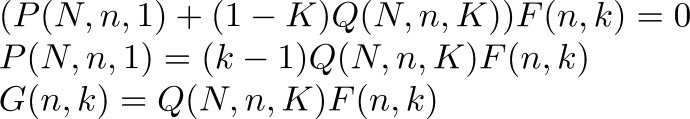
[http://s7.sinaimg.cn/mw690/001mqGQAzy7m5shCAnQf6](http://album.sina.com.cn/pic/001mqGQAzy7m5shCAnQf6)

如果Sister Mary Celine 算法的递归关系式存在，那么存在一个多项式P

[](http://album.sina.com.cn/pic/001mqGQAzy7m5sk0A7P75)

​

可以在K=1处展开

[](http://album.sina.com.cn/pic/001mqGQAzy7m5skVwRKbc)

上面的递归关系式化为了

[http://s8.sinaimg.cn/mw690/001mqGQAzy7m5sm18LZd7](http://album.sina.com.cn/pic/001mqGQAzy7m5sm18LZd7)

注意a(n)对于变量k来说，可以看做是一个常数。这时我们就可以使用待定系数法了。我们假定用户给出一个i的上限。等号左边是一个已知的t(k)，等号右边是z(k+1)-z(k)

**这个问题化为了Gosper问题！**

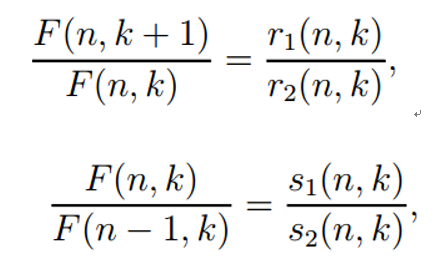
如果此时解出来ai，那么返回结果，如果没解出来，那么返回“试试看更高阶的”。这就是creative telescope算法。

总结一下：

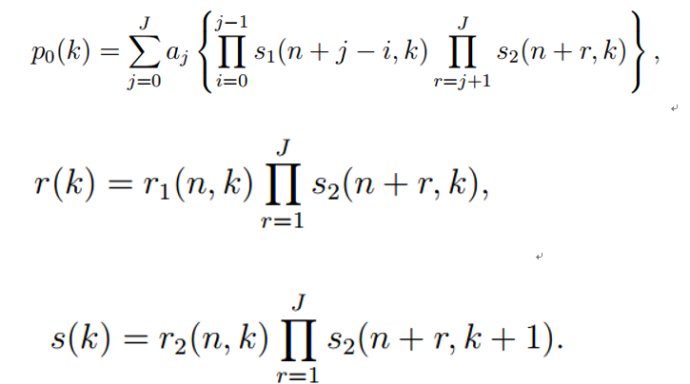
我们已知F(n,k)需要计算​f(n)=sum(F(n,k),{k,-infity,infty})满足的递归关系式。

（1）输入F(n,k)，以及我们需要尝试的递归的阶数J。

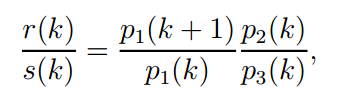
（2）由F(n,k)得到多项式s1(n,k),s2(n,k),r1(n,k),r2(n,k)

[](http://album.sina.com.cn/pic/001mqGQAzy7m5stbc6ce8)

（3）由s1,s2,r1,r2，以及J，以及待定系数aj计算p0,r,s

[](http://album.sina.com.cn/pic/001mqGQAzy7m5su36I54d)

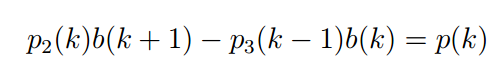
（3）由r(k),s(k)以及Gosper算法，由r(n)计算a(n),b(n),c(n)的算法，得到p1,p2,p3

[](http://album.sina.com.cn/pic/001mqGQAzy7m5swIRdJ87)

​

（4）再由p1,p0得到p(k)=p1(k)p0(k)。

（5）然后用Gosper算法解递归

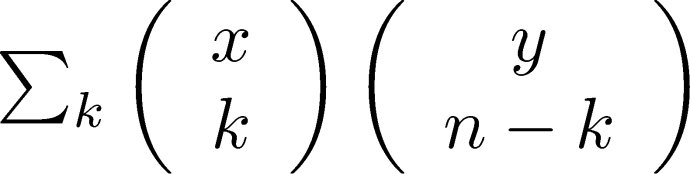
[](http://album.sina.com.cn/pic/001mqGQAzy7m5sxXkrzf5)

​

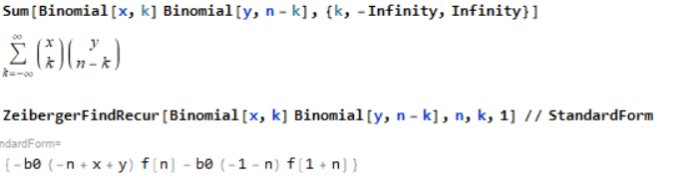
解这个递归，也就是首先确定b(k)的最大degree，再用待定系数法。此时，待定系数aj(n)，相对于自变量k来说，可以看做是常数。解的时候同时也解出来了aj(n)。此时就可以得到递归关系了。

**接下来可以实验这个算法的强大之处。**

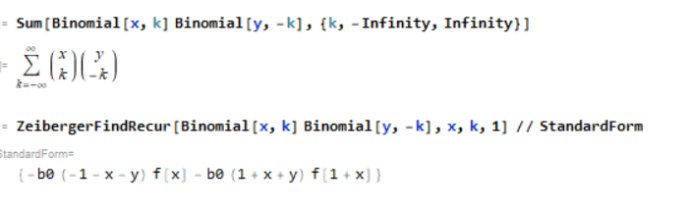
 比如，我们试图计算

[](http://album.sina.com.cn/pic/001mqGQAzy7m5sBgpXZ7b)

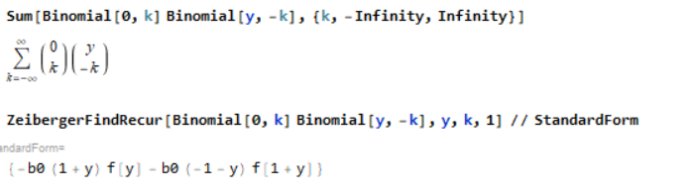
Mathematica算不出来，但是我们能用creative telescope得到递归：

[](http://album.sina.com.cn/pic/001mqGQAzy7m5sCklM77b)

 接下来我们需要n=0的值，这个mathematica依旧算不出来，但是我们可以得到一个关于x的递归

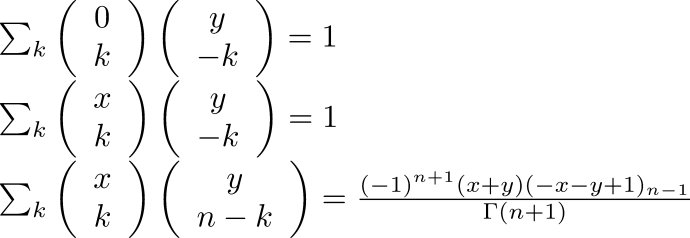
[](http://album.sina.com.cn/pic/001mqGQAzy7m5sE94rHbd)

 接下来，我们需要知道x=0时的值，这个mathematica依旧算不出来，但是我们可以得到一个关于y的递归

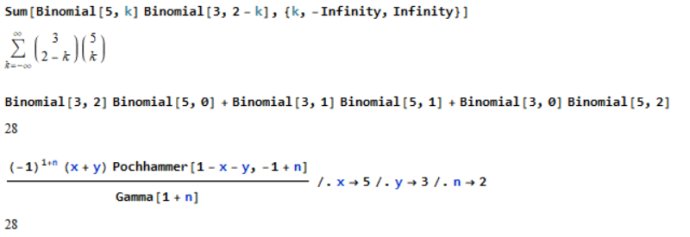
[](http://album.sina.com.cn/pic/001mqGQAzy7m5sESzLTb5)

​y=0的值，我们肉眼就能看出来，是1。

 由此一步一步的用递归回去，我们可以得到

[](http://album.sina.com.cn/pic/001mqGQAzy7m5sFOKYR35)

​可以实际带入x,y,n验证一下，比如

[](http://album.sina.com.cn/pic/001mqGQAzy7m5sH1P9S98)

 由这个例子可以看出来，对于一些特殊情况，creative telescope已经比Mathematica强大一些了。

 实际上，我测试的时候遇到不少能用creative telescope加解递归的方式计算的求和，是Mathematica算不出来的。

不过，也有不少是Mathematica能计算出来，但creative telescope的方法是算不出来的求和。

二者各有所长。​

Poly和Hyper算法

我们从creative telescope算法得到的递归，如果是1阶的，那么是很方便解决的。但如果不是一阶的该怎么办呢？

 A=B书中给出来了Poly和Hyper算法专门计算递归。

根据我的实验，这套算法，是不如Mathematica的自带的RSolve的。尽管如此我也写了一份。全部的代码我都放到了Github。如果有需要也可以参考下。