生成函数方法

1 基本策略

定义：我们称

$$G(z) = g_0 + g_1 z + g_2 z^2 + \cdots = \sum_{n\geq 0} g_n z^n$$

是数列$\langle g_0, g_1, g_2, \cdots \rangle$的生成函数。$G(z)$中$z^n$的系数记为$[z^n]G(z)$。

设$F(z)$,$G(z)$分别是$\langle f_n \rangle$,$\langle g_n \rangle$的生成函数，我们可以利用生成函数的加减乘除、求导、积分来得到新的数列的生成函数。

- $\alpha F(z) + \beta G(z) = \sum_{n=0}^{\infty} (\alpha f_n + \beta g_n) z^n$
- $z^m G(z) = \sum_{n=0}^{\infty} g_n z^{n+m} = \sum_{n=m}^{\infty} g_{n-m} z^n$ ， $m \geq 0$
- $\frac{1}{z^m}\left(G(z) - \sum_{k=0}^{m-1} g_k z^k\right) = \sum_{n=m}^{\infty} g_n z^{n-m} = \sum_{n=0}^{\infty} g_{n+m} z^n$
- $G(cz) = \sum_{n=0}^{\infty} c^n g_n z^n$

- $G'(z) = \sum_{n=1}^{\infty} ng_n z^{n-1} = \sum_{n=0}^{\infty} (n+1) g_{n+1} z^n$
- $zG'(z) = \sum_{n=0}^{\infty} (n+1) g_{n+1} z^{n+1} = \sum_{n=1}^{\infty} ng_n z^n = \sum_{n=0}^{\infty} ng_n z^n$
- $\int_0^z G(t)dt = \sum_{n=0}^{\infty} \frac{g_n}{n+1} z^{n+1} = \sum_{n=1}^{\infty} \frac{g_{n-1}}{n} z^n$
- $F(z)G(z) = \sum_{n=0}^{\infty} \left(\sum_{k=0}^{n} f_k g_{n-k} \right) z^n = \sum_{n=0}^{\infty} (f * g)_n z^n$
- $\frac{1}{1-z} G(z) = \sum_{n=0}^{\infty} \left(\sum_{k=0}^{n} g_n \right) z^n$

另外，还有一些特殊的数列的生成函数可以按照如下规则给出：

- $\langle 1,0,0,0,\cdots \rangle : \sum_{n=0}^{\infty} [n = 0] z^n = 1$
- $\langle 0,0,\cdots,0,1,0,0,\cdots \rangle : \sum_{n=0}^{\infty} [n = m] z^n = z^m$

- $\left\langle 1,1,1,1,\cdots \right\rangle : \sum_{n=0}^{\infty} z^n = \frac{1}{1-z}$
- $\left\langle 1,-1,1,-1,\cdots \right\rangle : \sum_{n=0}^{\infty} \left(-1\right)^n z^n = \frac{1}{1+z}$
- $\left\langle 1,0,1,0,\cdots \right\rangle : \frac{1}{2}\left(\sum_{n=0}^{\infty} z^n + \sum_{n=0}^{\infty} \left(-1\right)^n z^n \right) = \frac{1}{2}\left(\frac{1}{1-z} + \frac{1}{1+z} \right) = \frac{1}{1-z^2}$
- $\left\langle 1,2,3,... \right\rangle : \sum_{n=0}^{\infty} \left(n+1\right) z^n = \frac{\mathrm{d}}{\mathrm{d}\,z}\left(\frac{1}{1-z} \right) = \frac{1}{\left(1-z\right)^2}$
- $\left\langle 0,1,\frac{1}{2},\frac{1}{3},\cdots \right\rangle : \sum_{n=1}^{\infty} \frac{1}{n} z^n = \int_0^z \sum_{n=0}^{\infty} t^n dt = \ln \frac{1}{1-z}$
- $\left\langle 0,1,-\frac{1}{2},\frac{1}{3},-\frac{1}{4},\cdots \right\rangle : \sum_{n=1}^{\infty} \frac{\left(-1\right)^{n+1}}{n} z^n = \ln\left(1+z\right)$
- $\left\langle 1,1,\frac{1}{2},\frac{1}{6},\frac{1}{24},\cdots \right\rangle : \sum_{n=1}^{\infty} \frac{1}{n!} z^n = e^z$

2 求解方法

Step 1. 利用数列中的其它元素写出 g_n 的单个方程，使得方程对所有整数 n 都成立。

假设 $g_{-1} = g_{-2} = \cdots = 0$ 。

Step 2. 用 z^n 乘以方程两边，并对所有 n 求和，提取和式中的 $\sum_n g_n z^n$ ，令为 $G(z)$

从而得到关于 $G(z)$ 的方程。

Step 3. 求解上述方程得到 $G(z)$ 的闭形式解。

Step 4. 将 $G(z)$ 展开成幂级数形式，可以得到 $\langle g_n \rangle$ 的闭形式解。

2.1 常系数线性差分方程

例：菲波那切数列

$$\begin{cases} g_0 = 0, g_1 = 1 \\ g_n = g_{n-1} + g_{n-2} \end{cases}$$

列出单个方程，而不是上述分支方程。注意到对于 $n \leq 0$ ， $g_n = g_{n-1} + g_{n-2}$ 依然成立，唯一的转折点是 $n = 1$ 时的情况，于是，我们可以有

$$g_n = g_{n-1} + g_{n-2} + [n = 1]$$

然后乘上 z^n 并求和

$$\sum_n g_n z^n = \sum_n g_{n-1} z^n + \sum_n g_{n-2} z^n + \sum_n [n=1] z^n$$

$$G(z) = \sum_n g_n z^{n+1} + + \sum_n g_n z^{n+2} + z$$

$$= zG(z) + z^2 G(z) + z$$

$$G(z) = \frac{z}{1-z-z^2}$$

最后，将上述生成函数展开为级数形式。大部分情况下，这一步可以通过 Maple 集成的 FPS(Formal Power Series)函数包，利用`convert/FormalPowerSeries`来完成

```
convert(z/(1-z-z^2),FormalPowerSeries,z)
```

$$\sum_{k=0}^{\infty} \left(-\frac{1}{5}\sqrt{5}\left(\frac{1}{2}-\frac{1}{2}\sqrt{5}\right)^k + \frac{1}{5}\sqrt{5}\left(\frac{1}{2}\sqrt{5}+\frac{1}{2}\right)^k \right) z^k$$

2.2 简单常系数线性差分方程组

对于问题

$$U_0 = 1, U_1 = 0, V_0 = 0, V_1 = 1$$
$$U_n = 2V_{n-1} + U_{n-2}$$
$$V_n = U_{n-1} + V_{n-2}$$

可以建立方程组

$$U_n = 2V_{n-1} + U_{n-2} + [n = 0]$$
$$V_n = U_{n-1} + V_{n-2}$$

从而得到

$$U(z) = 2zV(z) + z^2U(z) + 1$$
$$V(z) = zU(z) + z^2V(z)$$

求解该方程组得到

$$U(z) = \frac{1 - z^2}{1 - 4z^2 + z^4}, V(z) = \frac{z}{1 - 4z^2 + z^4}$$

展开即可得到原方程组的闭形式解

```
U(z)=convert((1-z^2)/(1-4*z^2+z^4),FormalPowerSeries,z)
```

$$U(z)=\sum_{k=0}^{\infty}\sum_{\_\alpha 1=RootOf(\_Z^4-4\_Z^2+1)}\left(-\frac{1}{12}\frac{z^k\,\_\alpha 1\,(\_\alpha 1^2-5)}{\_\alpha 1^{k+1}}\right)$$

```
V(z)=convert(z/(1-4*z^2+z^4),FormalPowerSeries,z)
```

$$V(z)=\sum_{k=0}^{\infty}\sum_{\_\alpha 1=RootOf(\_Z^4-4\_Z^2+1)}\left(-\frac{1}{12}\frac{z^k\,(\_\alpha 1^2-2)}{\_\alpha 1^{k+1}}\right)$$

2.3 发散级数

对于方程

$$\begin{aligned} g_0 &= 1 \\ g_n &= n g_{n-1} \end{aligned}$$

显然有 $g_n = n!$，现在来看如何用生成函数方法来求解这种情况。显然，满足条件的方程为

$$g_n = n g_{n-1} + \left[n = 0 \right]$$

求和有

$$\begin{aligned} G\left(z \right) &= 1 + \sum_n n g_{n-1} z^n \\ &= 1 + z^2 G'\left(z \right) + z G\left(z \right) \end{aligned}$$

可以得到

$$G\left(z \right) = F\left(1,1;;z \right) = \sum_{n \geq 0} \frac{1^{\overline{n}} 1^{\overline{n}}}{n!} z^n = \sum_{n \geq 0} n! z^n$$

2.4 完全返回递归式

对于递归方程

$$\begin{aligned} f_n &= 2f_{n-1} + f_{n-2} + \cdots + f_1 + 1 \\ f_1 &= 1 \end{aligned}$$

尽管我们可以通过令 $S_n = \sum_{k=1}^{n} f_n$ 得到更简单的递归方程

$$S_n - S_{n-1} = S_{n-1} - S_{n-2} + S_{n-1} + 1$$

但是，我们仍旧想要直接采用生成函数方法来求解，以至于将其拓展至更复杂的和式。首先，我们可以得到方程

$$f_n = f_{n-1} + \sum_{k<n} f_k + [n > 0]$$

两边求和

$$\begin{aligned} F(z) = \sum_n f_n z^n &= \sum_n f_{n-1} z^n + \sum_{k,n} f_k z^n [k < n] + \sum_n [n > 0] z^n \\ &= zF(z) + \sum_k f_k z^k \sum_n [n > k] z^{n-k} + \frac{z}{1-z} \\ &= zF(z) + F(z)\frac{z}{1-z} + \frac{z}{1-z} \end{aligned}$$

从而可以继续求解。

3 卷积

我们有

$$F(z)G(z)=\sum_{n=0}^{\infty}\left(\sum_{k=0}^{n}f_k g_{n-k}\right)z^n=\sum_{n=0}^{\infty}(f*g)_n z^n$$

即数列卷积的生成函数是对应数列生成函数的乘积。该结论可以推广至多个序列

$$\prod_{k=0}^{m}F_k(z)=\sum_{n=0}^{\infty}\left(\sum_{k_1+k_2+\cdots+k_m=n}f_{k_1}^{(1)}f_{k_2}^{(2)}\cdots f_{k_m}^{(m)}\right)z^n$$

例

$$C_n=C_0C_{n-1}+C_1C_{n-2}+\cdots+C_{n-1}C_0,C_0=1$$

满足条件的方程为

$$C_n=\sum_{k=0}^{n-1}C_kC_{n-1-k}+[n=0]$$

求和有

$$
\begin{aligned}
C(z) &= \sum_n C_n z^n \\
&= z\sum_n \sum_k C_k C_{n-1-k} z^{n-1} + 1 \\
&= zC^2(z) + 1 \\
C(z) &= \frac{1 \pm \sqrt{1-4z}}{2z}
\end{aligned}
$$

因为$C(0) = C_0 = 1$，故取$C(z) = \dfrac{1-\sqrt{1-4z}}{2z}$，从而有

$$
C_n = \binom{2n}{n}\frac{1}{n+1}
$$

4 指数生成函数

有时候$\langle g_n \rangle$会有一个相当复杂的生成函数，但是$\langle g_n / n! \rangle$可能具有一个简单的生成函数，最后将级数展开的结果乘上$n!$即可。我们称

$$G(z) = \sum_{n=0}^{\infty} \frac{g_n}{n!} z^n$$

为指数生成函数(exponential generation function,简称 erf)。

例如，e^z是$\langle 1,1,1,\cdots \rangle$的指数生成函数。Erf 也有着它的基本运算规则。

$$zG(z) = \sum_{n\ge0} g_n \frac{z^{n+1}}{n!} = \sum_{n\ge1} g_{n-1} \frac{z^n}{(n-1)!} = \sum_{n\ge0} n g_{n-1} \frac{z^n}{n!}$$

$$G'(z) = \sum_{n\ge1} g_n \frac{z^{n-1}}{(n-1)!} = \sum_{n\ge0} g_{n+1} \frac{z^n}{n!}$$

$$\int_0^z G(t)\,dt = \sum_{n\ge0} g_n \frac{z^{n+1}}{(n+1)!} = \sum_{n\ge0} g_{n-1} \frac{z^n}{n!}$$

可以看出，和普通生成函数不同，$\langle g_n \rangle$引入关于n的多项式的生成函数由复杂的导数操作，变成了简单的倍数操作。与之对应的，$\langle g_n \rangle$的移位操作由简单的倍数操作变成了复杂的积分和微分操作。但是，$\langle g_n \rangle$乘上关于n的多项式，依然需要

借助积分和微分操作。

另外，erf 的乘法，对应的是数列的二项式卷积

$$\begin{aligned} F(z)G(z) &= \left(\sum_n f_n \frac{z^n}{n!}\right)\left(\sum_n g_n \frac{z^n}{n!}\right) \\ &= \sum_n \left(\sum_{k=0}^{n} \frac{f_k}{k!} \frac{g_{n-k}}{(n-k)!}\right) z^n \\ &= \sum_n \left(\sum_{k=0}^{n} \frac{n!}{k!(n-k)!} \frac{f_k g_{n-k}}{n!}\right) z^n \\ &= \sum_n \left(\sum_{k=0}^{n} \binom{n}{k} f_k g_{n-k}\right) \frac{z^n}{n!} \end{aligned}$$

5 狄利克雷生成函数

有许多方法可以从一个级数生成一个数列，从原则上来说，任何满足

$$G\left(z\right)=\sum_{n}g_nK_n\left(z\right)$$

形式的函数都能作为生成函数。其中 $K_n\left(z\right)$ 称为生成函数的核。

通常的生成函数取 $K_n\left(z\right)=z^n$，指数生成函数取 $K_n\left(z\right)=z^n/n!$，也可使用 $z^{\underline{n}}$，或者使用二项式系数 $z^{\underline{n}}/n!$。另一种重要的核是 $1/n^z$。我们称

$$G\left(z\right)=\sum_{n=1}^{\infty}\frac{g_n}{n^z}$$

为狄利克雷生成函数。

例如，$\left\langle 1,1,1,\cdots\right\rangle$ 的狄利克雷生成函数为 $\sum_{n\geq 1}\frac{1}{n^z}=\zeta\left(z\right)$，是黎曼 ζ 函数。

狄利克雷生成函数的乘积和一类特殊的卷积对应一类特殊的卷积

$$F\left(z\right)G\left(z\right)=\sum_{n\geq 1}\frac{1}{n^z}\sum_{lm=n}f_lg_m$$

6 变系数线性差分方程

现在，我们考虑变系数线性差分方程

$$g_n = f_1(n)g_{n-1} + f_2(n)g_{n-2} + \cdots + f_d(n)g_{n-d} + h(n), \quad n \geq d$$

而 $g_0, g_1, \cdots, g_{d-1}$ 的值已知。此时我们可以写出一般的方程

$$g_n = \left(f_1(n)g_{n-1} + f_2(n)g_{n-2} + \cdots + f_d(n)g_{n-d} + h(n)\right)[n \geq d] + \sum_{k=0}^{d-1} g_k [n = k]$$

两边乘上 z^n 求和有

$$\sum_n g_n z^n = \sum_{n \geq d}\left(f_1(n)g_{n-1} + f_2(n)g_{n-2} + \cdots + f_d(n)g_{n-d} + h(n)\right)z^n + \sum_{n=0}^{d-1} g_n z^n$$

该方程可以化为关于 $G(z)$ 的方程的关键在于

$$\sum_{n \geq d} f_1(n)g_{n-1}z^n, \sum_{n \geq d} f_2(n)g_{n-2}z^n, \cdots, \sum_{n \geq d} f_d(n)g_{n-d}z^n, \sum_{n \geq d} h(n)z^n$$

能够用 $G(z)$ 表示。

- 当 $h(n) = 0$，该问题是对应一个齐次线性差分方程
  - 当 $f_1(n), \cdots, f_d(n)$ 是常数时，可以预见，能够解得 $G(z)$ 是一个有理函数。这和“常系数齐次线性差分方程的解的生成函数都是有理函数想对应”。

- 当$f_1(n),\cdots,f_d(n)$是关于n的多项式时，能借助$G(z)$的微分运算来得到$\sum_{n\ge d} f_1(n)g_{n-1}z^n,\sum_{n\ge d} f_2(n)g_{n-2}z^n,\cdots,\sum_{n\ge d} f_d(n)g_{n-d}z^n$，从而可以对应“多项式系数的齐次线性递归方程的解的生成函数是多项式系数的线性微分方程的解”。
- 当$f_1(n),\cdots,f_d(n)$不是关于n的多项式时，就无能为力了。

- 当$h(n)$非零时，对应一个非齐次线性差分方程，在能够转化其它部分的前提下，关键是解决$\sum_{n\ge d} h(n)z^n$这个求和问题。

6.1 多项式系数的线性差分方程

注意到

$$
\begin{aligned}
z^m G^{(m)}(z) &= z^m \sum_{n \ge m} n(n-1)\cdots(n-m+1) g_n z^{n-m} \\
&= \sum_{n \ge m} n^{\underline{m}} g_n z^n = \sum_{n \ge 0} n^{\underline{m}} g_n z^n
\end{aligned}
$$

而任意 m 阶多多项式 $p(n) = p_0 + p_1 n + \cdots + p_m n^m$ 又能被 $n^{\underline{0}}, n^{\underline{1}}, \cdots n^{\underline{m}}$ 线性表出，从而 $p(n) g_n$ 的生成函数也能被 $G(z), zG'(z), z^2 G^{(2)}(z), \cdots, z^m G^{(m)}(z)$ 线性表出。

从而任意多项式系数的线性差分方程

$$
g_n = p_1(n) g_{n-1} + p_2(n) g_{n-2} + \cdots + p_d(n) g_{n-d} + p(n), \quad n \ge d
$$

都能得到其生成函数满足系数为多项式的微分方程。

事实上 Maple 的 LRETools[REtoDE]函数实现了这个功能

```
> with(LRETools):
> REtoDE(REcreate(a(n+2)-n^2*a(n+1)+(n-17)*a(n) = 0, a(n), {a(0)=0}),f(z))
```

$$
DESol\left(\left\{\left(17\ z^2+1\right)\ f+\left(15\ z^3-z\right)\ \mathrm{D}(f)+\left(-z^4+z^3\right)\ \mathrm{D}^{(2)}(f)+z^4\ \mathrm{D}^{(3)}(f)\right\},\ \{f\},\ \left\{f(0)=0,\ \mathrm{D}^{(2)}(f)(0)=0\right\}\right)
$$

6.2 求和问题

将生成函数方法推广到任意系数的线性差分方程的关键是计算

$$\sum_{n=0}^{\infty} f(n)\, g_n z^n$$

对于任意的 $f(n)$，可以展开为

$$f(n)=\sum_{k=0}^{\infty}\frac{\Delta^k f(0)}{k!}n^{\underline{k}}=\sum_{k=0}^{\infty}f_k n^{\underline{k}}$$

从而

$$\begin{aligned}\sum_{n=0}^{\infty} f(n)\, g_n z^n &= \sum_{n=0}^{\infty}\left(\sum_{k=0}^{\infty} f_k n^{\underline{k}}\right)g_n z^n = \sum_{k=0}^{\infty} f_k\left(\sum_{n=0}^{\infty} n^{\underline{k}} g_n z^n\right)\\ &= \sum_{k=0}^{\infty} f_k z^k G^{(k)}(z) = \sum_{k=0}^{\infty}\Delta^k f(0)\left(\frac{G^{(k)}(z)}{k!}z^k\right)\end{aligned}$$

虽然看起来很神奇，但是并没有什么卵用。