1.三种方法计算e^{-x}

直接计算

这种方法的稳定性最差。可以预料，当x, n不断增大时，x^n与$n!$会溢出

递归法

计算方式为 $s_n = s_{n-1} \times (-1)x/n, \quad s_0 = 1$ ，于是

$$T = e^{-x} = \sum_{n=0}^{\infty}(-1)^n \frac{x^n}{n!} = \sum_{n=0}^{\infty} s_n$$

中间项s_n的变化规律是：s_n的符号不断震荡，其绝对值在$n < x$时递增，$n > x$时递减并趋向0.
设我们取前n+1项$T_n = \sum_{k=0}^{n} s_k$作为e^{-x}的一个估计，则根据taylor展开的余项公式，有

$$T = T_n + f^{(n+1)}(\xi)\frac{(-1)^{n+1}x^{n+1}}{(n+1)!} \quad (\xi \in (0, x))$$

可以看到，T_n中每一项都是正负交错的，这导致级数的收敛很慢。
考察余项：

$$|e^{-x} - T_n| = \frac{e^{-\xi}x^{n+1}}{(n+1)!} < \frac{x^{n+1}}{(n+1)!} \overset{n>>1}{\sim} (\frac{ex}{n})^n$$

其中用到了Stirling公式：$n! \sim n^n e^{-n}\sqrt{2\pi n}$。
当$x < n/e$时$ex/n < 1$，此时估计值更接近。
从实际计算结果中来看，$maxn = 100$,$x < 100/e = 36.78$ 但x从大约17开始就与真值之间有很大偏差了。这是因为n小时，各个s_n都比较大，它们相减造成有效数字位数降低。

取逆法

先按递归法计算e^x，再计算其倒数。
由于不存在大数相减和溢出的情况，这样计算得到的值精度最高。

2.矩阵的模和条件数

2.1 A的行列式与奇异性

$$\det(A) = 1 \neq 0$$

故A不是奇异矩阵。

2.2 A的逆矩阵形式

设

$$J = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}$$
$$J^n = 0$$

A可以写作：

$$A = I - J - \cdots - J^{n-1}$$

设$A^{-1} = I + \sum_{k=1}^{n-1} c_k J^k$

$$A^{-1}A = (I + \sum_{k=1}^{n-1} c_k J^k)(I - \sum_{k=1}^{n-1} J^k) \tag{1}$$
$$= I + \sum_{l=1}^{n-1} (c_l - 1 - \sum_{k'=1}^{l-1} c_{l-k'}) J^l \tag{2}$$

选取各个c_k使得J^l系数为0：

$$c_l = 1 + \sum_{k=1}^{l-1} c_k$$

递推得到$c_l = 2^{l-1}$
从而

$$A^{-1} = I + \sum_{k=1}^{n-1} 2^{k-1} J^k$$

2.3 A的∞-模

$$
\begin{aligned}
||A||_p &= \sup_{||\vec{x}||_p=1} \left|\left| \begin{pmatrix} \vec{a}_1 & \vec{a}_2 & \cdots \vec{a}_n \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \right|\right|_p \\
&= \sup_{||\vec{x}||_p=1} || \sum_j \vec{a}_j x_j ||_p \\
&\leq \sup_{||\vec{x}||_p=1} \sum_j ||\vec{a}_j||_p x_j \\
&\overset{p\to\infty}{=} \sup_{\max_j\{x_j\}=1} \sum_j x_j \sup_i |a_{ij}| \\
&\leq \sum_j \sup_i |a_{ij}| \\
&= \sup_i \sum_j |a_{ij}|
\end{aligned}
$$

取各个$x_j = \pm 1 \quad j = 1, 2, \cdots$, 则无穷模可以取得其最大值:

$$
||A\vec{x}||_\infty = \max_i \left| \sum_j a_{ij} x_j \right| = \max_i \sum_{j=1}^n |a_{ij}|
$$

2.4

幺正矩阵是保持态矢量的内积不变的矩阵:

$$
\langle\psi|\psi\rangle = \langle\psi| U^\dagger U |\psi\rangle
$$

$$
\begin{aligned}
||U||_2 &= \sup_{|\psi\rangle\neq 0} \frac{||U|\psi\rangle||_2}{|| |\psi\rangle ||_2} \\
&= \sup_{|\psi\rangle=1} \langle\psi| U^\dagger U |\psi\rangle \\
&= 1
\end{aligned}
$$

矩阵的2-模服从乘法，故有

$$
||UA||_2 = ||U||_2 ||A||_2 = ||A||_2
$$

2.5

由上面计算结果知,

$$||A||_\infty = n, ||A^{-1}||_\infty = 2^{n-1}$$
$$K_\infty(A) = n \times 2^{n-1}$$

3.Hilbert矩阵

3.1

$$D(c_1, c_2, \cdots, c_n) = \int_0^1 dx \left(\sum_{i=1}^n c_i x^{i-1} - f(x)\right)^2$$

对各个参量c_i求偏导数，得到方程：

$$\frac{\partial D}{\partial c_i} = \int_0^1 dx \cdot 2\left(\sum_{i=1}^n c_i x^{i-1} - f(x)\right) x^{i-1} = 0$$
$$\int_0^1 x^{i-1} f(x) dx = \sum_{j=1}^n c_j \frac{1}{i+j-1}$$

得到各个系数：

$$\begin{cases} (H_n)_{ij} = \dfrac{1}{i+j-1} \\ b_i = \displaystyle\int_0^1 x^{i-1} f(x) dx \end{cases}$$

3.2

令$c_i = x^{i-1}$,代入：

$$g(x) = \equiv (1 \quad x \quad \cdots \quad x^{n-1}) H_n \begin{pmatrix} 1 \\ x \\ \vdots \\ x^{n-1} \end{pmatrix} = \sum_{k=2}^{2n} c_k \frac{x^{k-2}}{k-1}$$

其中

$$c_k = \begin{cases} k-1, & k \le n+1 \\ 2n-k+1, & k \ge n+1 \end{cases}$$

$$f(x) = xg(x)$$
$$f'(x) = \frac{(x^n-1)^2}{(x-1)^2} > 0$$

当$x > 0, f(x) > f(0) = 0 \Rightarrow f(x) \geq 0$
所以

3.3

$$\begin{aligned}\ln\det(H_n) &= 4\ln c_n - \ln c_{2n} \\ &= 4\sum_{i=1}^{n-1}\ln i! - \sum_{i=1}^{2n}\ln i!\end{aligned}$$

对c_n在$n >> 1$情形下取近似：

$$\ln c_n = \sum_{i=1}^{n-1} i! = \sum_{i=1}^{n-1}(i\ln i - i) \approx \int_1^n x\ln x dx - n(n-1)/2$$

积分可以算出：

$$\int x\ln x dx = \frac{1}{4}x^2(2\ln x - 1)$$

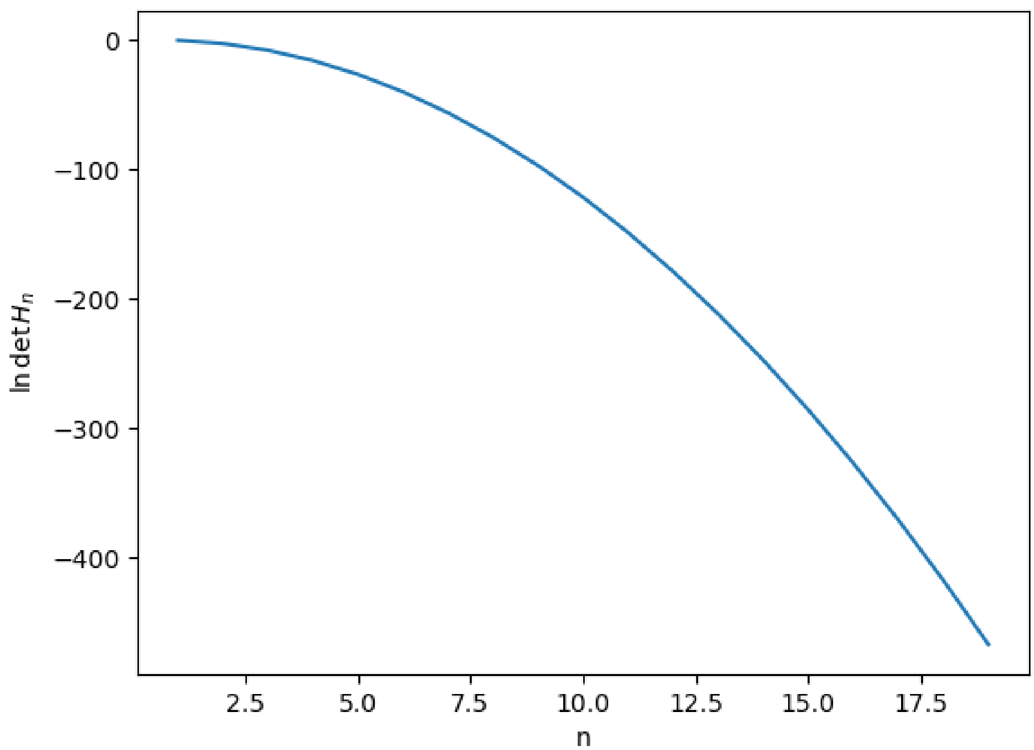
所以有：

$$\begin{gathered}4\ln c_n - \ln c_{2n} \approx -2n^2\ln 2 + C \\ \ln\det H_n \sim -n^2\ln 4 + C\end{gathered}$$

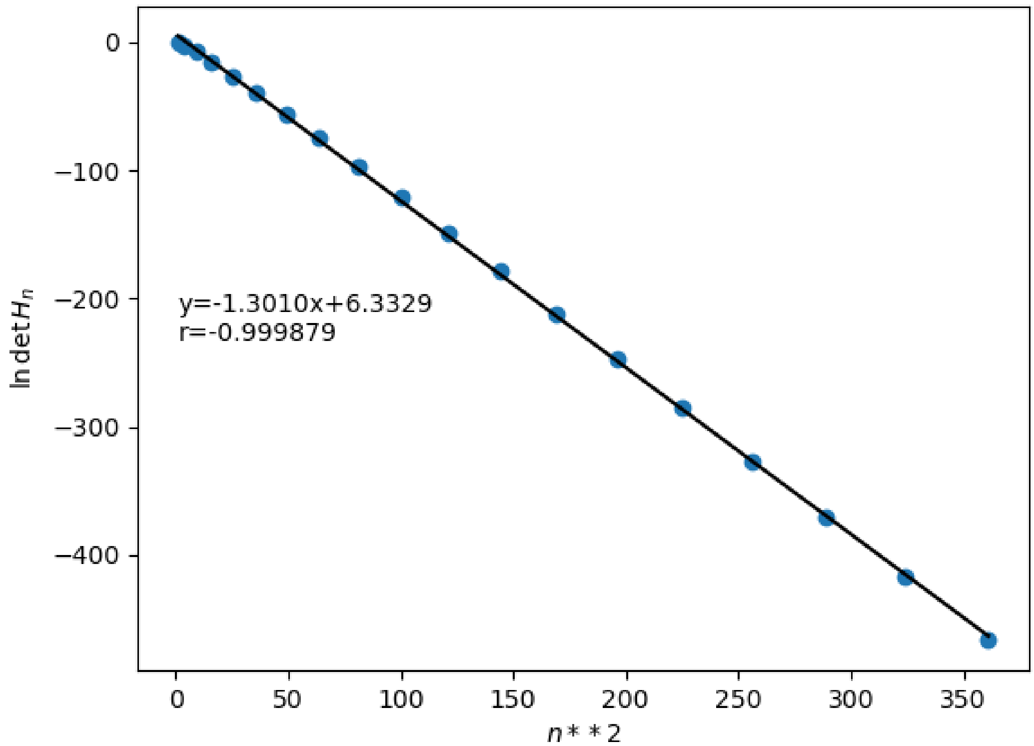
实际计算的结果如下：

n	$\ln\det(H_n)$
1	0.0
2	-2.4849066497880004
3	-7.67786350067821
4	-15.615238196841508
5	-26.309453258276445
6	-39.76620670609766
7	-55.988580206842556
8	-74.97842732916047

n	$\ln\det(H_n)$
9	-96.736949274397
10	-121.26496874930368



拟合得到：



可见近似效果比较理想。

3.4

为了实现计算，我们编写程序matrixCalc.py，实现矩阵类与基本的乘法、转置、上三角矩阵、下三角矩阵的反代，高斯消元、Cholesky分解等操作。可以运行T3.py来用Cholesky分解与GEM两种方法求解前10阶Hilbert矩阵在特定情形下的解。

```
问题    输出    终端    JUPYTER    注释    调试控制台

GEM sol 9:218789.27994628725
n=10
Cholesky sol 1:-9.997668234803768
Cholesky sol 2:989.7983719614995
Cholesky sol 3:-23755.70340866239
Cholesky sol 4:240200.9288274541
Cholesky sol 5:-1261073.6092823984
Cholesky sol 6:3783267.5764416014
Cholesky sol 7:-6725879.256791485
Cholesky sol 8:7000467.5376583515
Cholesky sol 9:-3937793.4846944376
Cholesky sol 10:923686.2089171143
GEM sol 1:-9.99726592260788
GEM sol 2:989.7630367370259
GEM sol 3:-23754.9411215222
GEM sol 4:240193.92766718712
GEM sol 5:-1261039.9343178642
GEM sol 6:3783174.355768353
GEM sol 7:-6725725.404387591
GEM sol 8:7000318.103178086
GEM sol 9:-3937714.6884527057
GEM sol 10:923668.8139455386

(base) d:\PKU\notes\Computational Physics\homework-1>
```

可以看到，由于Hilbert矩阵的条件数非常大，得到的解稳定性很差。但GEM与Cholesky得到的结果相差不大。

4.级数求和与截断误差

a.题目分析

$$f(q^2) = \left(\sum_{\vec{n}\in\mathbb{Z}^3} - \int d^3\vec{n}\right)\frac{1}{|\vec{n}|^2 - q^2}$$

所求的$f(q^2)$是一个无穷求和减去无穷积分的形式，其中对离散格点的求和是对积分的一个近似。二者的渐进行为在最高阶是相同的，很容易估计：

$$\begin{aligned}\int d^3\vec{n}\frac{1}{|\vec{n}|^2 - q^2} &= \lim_{\Lambda\to\infty}\int^{\Lambda}\frac{4\pi r^2}{r^2 - q^2}dr \\ &\approx \lim_{\Lambda\to\infty}\int^{\Lambda}4\pi dr \sim 4\pi\Lambda\end{aligned}$$

可以看到，求和或积分都是发散的，二者相减才能出现有限值。因此，为了计算求和，我们需要估计关于Λ^{-1}的零

阶和更高阶的项的系数。此外，关于$r=q$处的奇点，我们只要取积分的主值即可收敛，因此无需担心。
定义

$$
\begin{aligned}
g(\vec{r}_0,\Delta) &\equiv \int_{x_0-\frac{\Delta}{2}}^{x_0+\frac{\Delta}{2}}\int_{y_0-\frac{\Delta}{2}}^{y_0+\frac{\Delta}{2}}\int_{z_0-\frac{\Delta}{2}}^{z_0+\frac{\Delta}{2}} \frac{1}{x^2+y^2+z^2-q^2}dxdydz \quad (3)\\
&= \iiint f(\vec{r})d^3\vec{r} \quad (4)\\
f(\vec{r}) &\equiv f(\vec{r}_0)+(r_i-r_{0i})\frac{\partial f}{\partial r_i}(\vec{r}_0)+\frac{1}{2!}(r_i-r_{0i})(r_j-r_{0j})\partial_i\partial_j f(\vec{r}_0)+\cdots \quad (5)
\end{aligned}
$$

其中第一项为常量，在体积为1的空间积分得到本身；第二项关于中心点为奇宇称，积分必为0。
考虑第三项：

$$
\begin{aligned}
&\iiint_{(\vec{r}_0,\Delta)}(r_i-r_{0i})(r_j-r_{0j})\partial_i\partial_j f|_{\vec{r}_0}dV\\
=&\iiint_{(0,\Delta)}\sum_{i=1}^{3}\delta_i^2\partial_i\partial_i f|_{\vec{r}_0}dV\\
=&\Delta\Delta\cdot\frac{1}{3}\frac{\Delta^3}{4}\nabla^2 f(\vec{r}_0)\\
=&\frac{\Delta^5}{12}\nabla^2 f(\vec{r}_0)
\end{aligned}
$$

求$f(\vec{r}_0)$在格点处的Laplace：

$$
\begin{aligned}
\nabla^2 f &= \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}f\right)\\
&= \frac{2r^2+6q^2}{(r^2-q^2)^3}
\end{aligned}
$$

因此可以得出对原式的一个估计

$$
\begin{aligned}
f(q^2) &= \left(\sum_{\vec{n}\in\mathbb{Z}^3}-\int d^3\vec{n}\right)\frac{1}{|\vec{n}|^2-q^2}\\
&= \frac{-1}{12}\sum_{\vec{n}}\frac{2n^2+6q^2}{(n^2-q^2)^3}+\sum_{\vec{n}}O(|\nabla^3 f(\vec{n})|)+\cdots
\end{aligned}
$$

由于积分对奇数次幂项一定为零，因此我们可以进一步得到

$$
f(q^2) = \frac{-1}{12}\sum_{\vec{n}}\frac{2n^2+6q^2}{(n^2-q^2)^3}+\sum_{\vec{n}}O(|\nabla^4 f(\vec{n})|)+\cdots
$$

通过积分近似，可以得知这些项的收敛规律：$|\vec{n}| < \Lambda, \quad \Lambda >> 1$得到，上式中第一项$\sim \frac{1}{\Lambda}$,第二项$\sim \frac{1}{\Lambda^3}$。当我们计算$|\vec{n}| < \Lambda$处的$(\sum - \int d^3n)\frac{1}{n^2-q^2}$的值后，为了估算计算值与真实值之间的误差，我们用上式估计，有：

$$(\sum_{|\vec{n}|>\Lambda} - \int_{\Lambda}^{\infty} d^3n)\frac{1}{n^2-q^2} \approx -\frac{1}{12}\sum_{|\vec{n}|>\Lambda}\frac{2n^2+6q^2}{(n^2-q^2)^3} + O(\Lambda^{-3})$$

b.题目解答

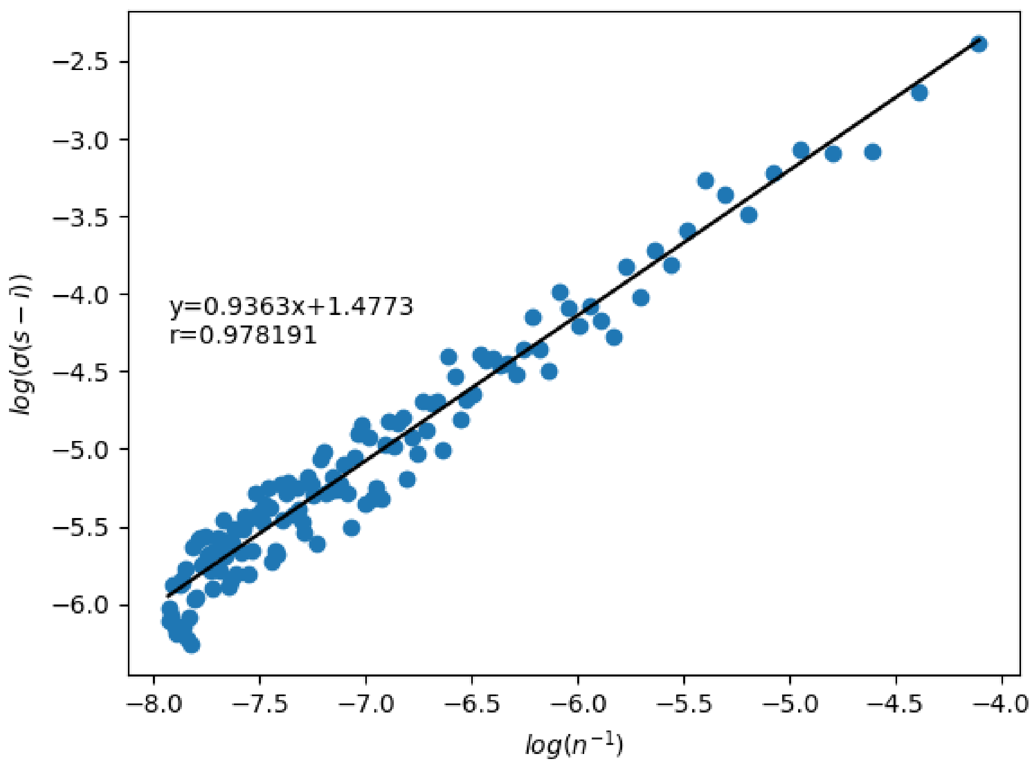
4.1

$f(0.5)$的近似值可以通过程序直接计算得到。详情请看T4.py。这个程序实现了通过累计球壳的和来计算$f(0.5)$的值，并把json数据存储在data中。提交作业时已经计算了$\Gamma \approx 3600$的结果。可直接运行来继续得到更高的结果，输入ctrl+C可自动终止并保存数据。

$$f(0.5) \approx 1.106$$

4.2

由于误差首项正比于Λ^{-1}，我们可以预测，当计算精度达到10^{-5}时大致需要$\Lambda \sim 10^5$。
为了得到更精确的数据，我们对计算数据进行拟合。取相邻20项计算标准差，与当前的半径值进行对数拟合，得到如下图所示结果：



可以看到，拟合曲线为：

$$\log\sigma = 0.93634\log n^{-1} + 1.47726 \quad R = 0.978$$

其中的比例系数$k=0.936$接近1，说明之前的近似是合理的。
代入$\sigma=10^{-5}$，得到$n\approx 1.06\times 10^6$

4.3

为了提高求和效率，我们接下来寻找该求和的其他表达形式，使得其收敛的速度更快。
定义$l(q,\vec{r})$如下:

$$l(q,\vec{r})=\left(\sum_{\vec{n}\in\mathbb{Z}^3}-\int d^3\vec{n}\right)\frac{e^{i\vec{n}\cdot\vec{r}}}{|\vec{n}|^2-q^2}$$

那么 $\lim_{|\vec{r}|\to 0} l(q,\vec{r})$ 就是我们要求的值
我们先考虑前面的求和项, 将其拆分为

$$\begin{aligned}\sum_{\vec{n}}\frac{e^{i\vec{n}\cdot\vec{r}}}{|\vec{n}|^2-q^2}&=\sum_{\vec{n}}\frac{e^{i\vec{n}\cdot\vec{r}}e^{-\left(|\vec{n}|^2-q^2\right)}}{|\vec{n}|^2-q^2}+\sum_{\vec{n}}\frac{e^{i\vec{n}\cdot\vec{r}}\left(1-e^{-\left(|\vec{n}|^2-q^2\right)}\right)}{|\vec{n}|^2-q^2}\\&=\sum_{\vec{n}}\frac{e^{i\vec{n}\cdot\vec{r}}e^{-\left(|\vec{n}|^2-q^2\right)}}{|\vec{n}|^2-q^2}+\sum_{\vec{n}}\int_0^1 dte^{i\vec{n}\cdot\vec{r}}e^{-t\left(|\vec{n}|^2-q^2\right)}\end{aligned}$$

将后面一项的求和号与积分号交换得到

$$\int_0^1 dte^{tq^2}\sum_{\vec{n}}e^{i\vec{n}\cdot\vec{r}}e^{-t|\vec{n}|^2}$$

采用Possion求和公式

$$\sum_{\vec{n}\in\mathbb{Z}^3}f(\vec{n})=\sum_{\vec{k}\in\mathbb{Z}^3}\hat{f}(\vec{k})$$
$$\hat{f}(\vec{k})=\int \mathrm{d}^3\vec{n}f(\vec{n})e^{-i2\pi\vec{k}\cdot\vec{n}}$$

代入$f(\vec{n})$的表达式:

$$
\begin{aligned}
\hat{f}(\vec{k}) &= \int \mathrm{d}^3\vec{n} e^{i\vec{n}\cdot\vec{r}-t|\vec{n}|^2} e^{-i2\pi\vec{k}\cdot\vec{n}} \\
&= \int \mathrm{d}^3\vec{n} e^{i\vec{n}\cdot(\vec{r}-2\pi\vec{k})-t|\vec{n}|^2} \\
&= 2\pi\int_0^\infty \mathrm{d}n n^2 \int_0^\pi \mathrm{d}\theta \sin\theta e^{i|\vec{r}-2\pi\vec{k}|n\cos\theta-tn^2} \\
&= 2\pi\int_0^\infty \mathrm{d}n n^2 \frac{2\sin(|\vec{r}-2\pi\vec{k}|n)}{|\vec{r}-2\pi\vec{k}|n} e^{-tn^2} \\
&= \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{|\vec{r}-2\pi\vec{k}|^2}{4t}}
\end{aligned}
$$

观察可以发现，$f(\vec{n})$在t较大时收敛较快，$f(\vec{k})$在t较小时收敛较快
于是我们使用Poisson求和公式可以提高求和的收敛速度。
求和项写作:

$$
\sum_{\vec{n}\in\mathbb{Z}^3} \frac{e^{i\vec{n}\cdot\vec{r}}}{|\vec{n}|^2-q^2} = \sum_{\vec{n}\in\mathbb{Z}^3} \frac{e^{i\vec{n}\cdot\vec{r}} e^{-\left(|\vec{n}|^2-q^2\right)}}{|\vec{n}|^2-q^2} + \int_0^1 dt e^{tq^2} \sum_{\vec{k}\in\mathbb{Z}^3} \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{|\vec{r}-2\pi\vec{k}|^2}{4t}}
$$

但后面一项中$\vec{k}=\vec{0}$的项是发散的，这对应了求和中的发散项。
于是我们需要改写原公式中的积分形式:

$$
\begin{aligned}
&\mathrm{P.V.} \iiint \frac{e^{-i\vec{r}\cdot\vec{n}}}{|\vec{n}|^2-q^2} d^3\vec{n} \\
&= \ \mathrm{P.V.} \int_0^\infty dn \int_0^\pi d\theta e^{-irn\cos\theta} \frac{2\pi n^2\sin\theta}{n^2-q^2} \\
&= 2\pi\frac{2}{r}\ \mathrm{P.V.} \int_0^\infty dn \frac{n\sin rn}{n^2-q^2} \\
&= 2\pi\frac{1}{r}\,\mathrm{Im}\ \mathrm{P.V.} \left(\int_{-\infty}^\infty dn \frac{ne^{irn}}{n^2-q^2}\right) \\
&= 2\pi^2\frac{\cos rq}{r}
\end{aligned}
$$

最后一个等号可以通过留数定理得到:

$$
\mathrm{P.V.} \int_{-\infty}^\infty dn \frac{ne^{irn}}{n^2-q^2} = \pi i(\mathrm{resf}(-q)+\mathrm{res}\, f(q)) = \pi i\cos rq
$$

取r很小的展开:

$$
\begin{aligned}
&\text{P.V.} \iiint \frac{e^{-i\vec{r}\cdot\vec{n}}}{|\vec{n}|^2-q^2} d^3\vec{n} = 2\pi^2 \frac{\cos rq}{r} \\
&= \frac{2\pi^2}{r} + \mathcal{O}(r) \\
&= \int_0^\infty \mathrm{d}t \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{r^2}{4t}} + \mathcal{O}(r) \\
&= \int_0^\infty \mathrm{d}t \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{|\vec{r}-2\pi\vec{k}|^2}{4t}} \Big|_{\vec{k}=\vec{0}} + \mathcal{O}(r)
\end{aligned}
$$

我们将级数和积分合并起来

$$
\begin{aligned}
l(q,\vec{r}) &= \sum_{\vec{n}\in\mathbb{Z}^3} \frac{e^{i\vec{n}\cdot\vec{r}} e^{-\left(|\vec{n}|^2-q^2\right)}}{|\vec{n}|^2-q^2} + \int_0^1 dt e^{tq^2} \sum_{\vec{k}\in\mathbb{Z}^3} \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{|\vec{r}-2\pi\vec{k}|^2}{4t}} - \int_0^\infty \mathrm{d}t \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{|\vec{r}-2\pi\vec{k}|}{4t}} \Big|_{\vec{k}=\vec{0}} + \mathcal{O}(r) \\
&= \sum_{\vec{n}\in\mathbb{Z}^3} \frac{e^{i\vec{n}\cdot\vec{r}} e^{-\left(|\vec{n}|^2-q^2\right)}}{|\vec{n}|^2-q^2} + \int_0^1 dt e^{tq^2} \sum_{\vec{k}\in\mathbb{Z}^3,\vec{k}\neq\vec{0}} \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{|\vec{r}-2\pi\vec{k}|^2}{4t}} \\
&\quad - \int_1^\infty \mathrm{d}t \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{r^2}{4t}} + \int_0^1 \mathrm{d}t \left(e^{tq^2}-1\right) \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{r^2}{4t}} + \mathcal{O}(r)
\end{aligned}
$$

在这个式子中我们可以轻松取 $\vec{r}\to\vec{0}$ 的极限, 得到

$$
\begin{aligned}
l(q,\vec{0}) = &\sum_{\vec{n}\in\mathbb{Z}^3} \frac{e^{-\left(|\vec{n}|^2-q^2\right)}}{|\vec{n}|^2-q^2} + \int_0^1 dt e^{tq^2} \sum_{\vec{k}\in\mathbb{Z}^3,\vec{k}\neq\vec{0}} \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{\pi^2|\vec{k}|^2}{t}} \\
&- \int_1^\infty \mathrm{d}t \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{r^2}{4t}} + \int_0^1 \mathrm{d}t \left(e^{tq^2}-1\right) \left(\frac{\pi}{t}\right)^{3/2}
\end{aligned}
$$

对积分化简，可以得到

$$
\begin{aligned}
l(q,\vec{0}) = &\sum_{\vec{n}\in\mathbb{Z}^3} \frac{e^{-\left(|\vec{n}|^2-q^2\right)}}{|\vec{n}|^2-q^2} + \int_0^1 dt e^{tq^2} \sum_{\vec{k}\in\mathbb{Z}^3,\vec{k}\neq\vec{0}} \left(\frac{\pi}{t}\right)^{3/2} e^{-\frac{\pi^2|\vec{k}|^2}{t}} \\
&- 2\pi^{\frac{3}{2}} e^{q^2} + 2\pi^2 q \operatorname{erfi}(q)
\end{aligned}
$$

其中$\operatorname{erfi}(x)$定义为

$$
\operatorname{erfi}(q) = \frac{2}{\sqrt{\pi}} \int_0^x e^{t^2}\ \mathrm{d}t
$$

通过该种方式的计算程序为T4_Fourier.py可以看到，该求和收敛速度很快，达到n=7时精度已经有16位.

```
(base) D:\PKU\notes\Computational Physics\homework-1>python T4_Fourier.py
1 -8.350851754936643
2 0.9754399409192533
3 1.1051729849172247
4 1.1061622251048497
5 1.1061625668146355
6 1.1061625668715358
7 1.1061625668715358
8 1.1061625668715358
9 1.1061625668715358
```