今天与大家分享一个很高效的精确求解无穷个随机变量之和的CDF（[累积分布函数](https://en.wikipedia.org/wiki/Cumulative_distribution_function)）的方法。

题意：设，其中为互相独立的正随机变量。已知有上界。给定一个正数，希望求出。

尝试：由于题目是无穷个随机变量的和，无法一个个枚举所有变量的情况，而且由[大数定理](https://en.wikipedia.org/wiki/Law_of_large_numbers)，的分布类似于高斯分布，[重尾效应](https://en.wikipedia.org/wiki/Heavy-tailed_distribution)严重（即有限的部分和无法精确拟合真正的的分布），故直接暴力枚举（有限个）并不能精确求解概率。

分析：定义函数

可以发现。观察到定义域是，可将周期延拓至整个实数轴，并将展开成**傅里叶级数**，有傅里叶系数，，。于是可以得到

。

令，上式化简为。由欧拉公式，可以发现，。继续代入，得到。

代入一下，由于期望的线性累加性，可以发现我们需要求和的期望。

下面介绍一下概率论中的特征函数。对随机变量，它的特征函数定义为，其中指数上的是虚数单位。它与概率分布函数一样，也完全定义了随机变量的所有性质。可以看到，。代入一下，得到（公式1）。

好了，接下来我们利用的性质去求它的特征函数。（倒数第二步用到了之间互相独立的性质）。对于不同的，我们可以单独计算其特征函数，他们的乘积即为的特征函数。

（建议自己手推一下加深理解；如果觉得步骤太麻烦，记住最后的结论就可以用啦~）

上面写了那么多理论公式，我们来看看实际怎么应用吧。

例题：

令

设，其中互相独立。给定一个正数，希望求出。

解：

可以发现的上界是。的特征函数为。所以的特征函数为

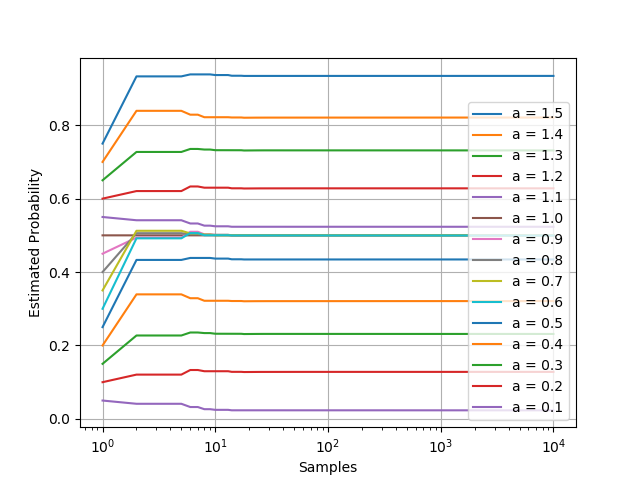
。代入上面公式1中，对每个固定的，我们估算（即取前一千项乘积作近似）；我们算前个值，即可达位收敛精度。

附上代码（这里我用了更宽松的上界）：

#include <cstdio>  
#include <cmath>  
const double a=0.5,U=2;  
int main()  
{  
 double ans=0;  
 for (int n=1;n<=10000;++n)  
 {  
 const double A=2\*M\_PI\*n/U;  
 double temp=1;  
 for (int i=1;i<=1000;++i)  
 temp\*=cos(A/2/i/i);  
 ans+=(sin(A\*a)\*(cos(A\*M\_PI\*M\_PI/12))+(1-cos(A\*a))\*(sin(A\*M\_PI\*M\_PI/12)))\*temp/A;  
 }  
 printf("%.12f\n",(ans\*2+a)/U);  
 return 0;  
}

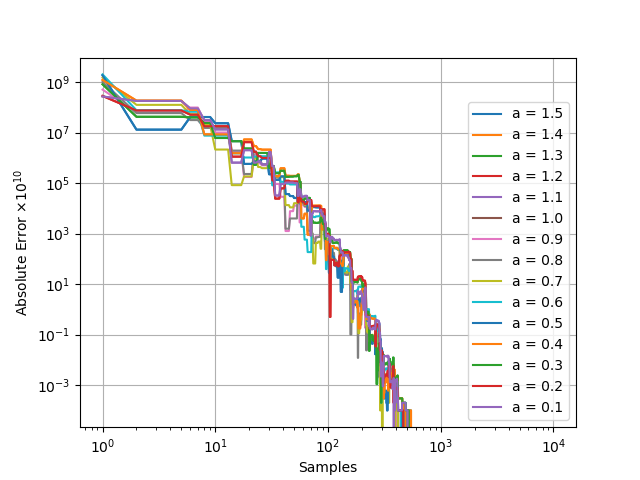
更进一步，不同的计算直接没有依赖，可进一步结合并行技巧进行加速。

收敛性分析：这里取了若干进行分析，横轴表示的样本数量，纵轴表示计算的概率值。（注意到在到之间的概率都一样，这是因为若不取第一项1，后面所有项的和是，所以无论怎么取都不会超过，即对。）



convergence

把绝对误差放大倍，再详细观察算法的收敛性：



convergence2

可以看到在个样本之后所有情况的收敛效果已经非常好了（误差均达到了）级别以下。

关于此方法的收敛性分析：傅里叶系数以速度收敛，特征函数最坏情况下以常数速度收敛，但一般常用函数情况下以速度指数级收敛，一般使用几万个样本即可收敛至位小数，非常高效。

关于上界的分析：不一定使用的上确界。实测下来发现即使比上确界大几十倍，收敛效果依然很好。此方法在难以估算上确界的情况下也可使用，扩展性高。

关于随机变量的正负：以上方法可扩展至任意有界的随机变量，过程从略，有兴趣的同学可以自行练手。

关于是否是无穷个随机变量的求和：以上方法可直接应用于有限个随机变量的求和，简单粗暴。

关于随机变量是连续变量or离散变量：以上方法也可直接应用，简单粗暴\*2。