GO FOR IT Q2

s336651 大垣慶介

2012 年 2 月 6 日

q2_gamma.py のドキュメント

```
============================================================
 q2_gamma.py
------------------------------------------------------------
 arguments
    |- 1.target : YYYY/MM/DD
    |- (2.alpha)
        |- default : 1000000
 output
    |- answer= $(target!)
    |- time= $(hour):$(minute):$(second)
 example
    |- $ q2_gamma.py 2.5 1000000
        |- output : "answer= 3.3233506936 \n time= 0:00:01.366805"
------------------------------------------------------------
 for Python 2.7
 K.Ogaki(ogaki@iis.u-tokyo.ac.jp)
 2012/02/06
------------------------------------------------------------
 This program is released under GPL.
 http://www.opensource.jp/gpl/gpl.ja.html.euc-jp
============================================================
```

仕様

上記のドキュメントのように、
$q2_gamma.py (target)

表 1: 環境

言語	python2.7
動作確認	Linux(Ubuntu10.10)

と入力することで (target)!を計算する。第二引数はオプションで、後述する alpha パラメータである。alpha を大きくすると精度が良くなる代わりに計算速度が遅くなる。引数が 0 の場合ドキュメントを出力する。実行環境は表のとおりである。

アルゴリズム

gamma 関数の計算にはスターリングの近似を用いた。以下が python でのコードである。

```
def stirling_approximation(z):
    return math.sqrt(2*math.pi*z)*( (z/math.exp(1))**z )
```

ただし、スターリングの近似は z が大きくなるほど近似精度がよくなるため、今回のような小さな z についてはあまり精度のよい近似ができない。そこで gamma 関数の基本的な性質 $gamma(z) = gamma(z+1)/(z+1)$ を用いて、大きな $z + alpha$ の $stirling\_approximation(z + alpha)$ から $stirling\_approximation\_plus(z)$ を求めることにした。これが先にのべた alpha パラメータである。alpha による計算量増加は O(alpha) のオーダー。また、実装上、大きな $gamma(z)$ は大きすぎてオーバーフローを起こすので、$stirling\_approximation(z + alpha)$ と $\prod_{z+1}^{z+alpha}$ による割り算を分解して逐次的に行うことでオーバーフローを防いでいる。以下にコードからの抜粋を示す。

```
    nowdivisor = z+alpha
    nowdivisor_int = math.floor(z+alpha)
    nowdivisor_rest = z-math.floor(z)
    powcount = 0

    now = math.sqrt(2*math.pi*(z+alpha) )

    while( powcount<(z+alpha) and (nowdivisor_int+nowdivisor_rest)>=(z+1) ):
        if(now>10 or now<-10):
            now /= (nowdivisor_int+nowdivisor_rest)
            nowdivisor_int-=1
        else:
            now *= ( (z+alpha)/math.exp(1) )
            powcount += 1
```

nowdivisor = nowdivisor_int + nowdivisor_rest に分解しているのは、alpha を大きく取ったときに nowdivisor が非常に大きな浮動小数点数になり、小数点以下の精度が悪くなるためである。特に z=-1.9 などを与えたときには nowdivisor=0.1 での割り算が発生するが、注意しないとここで符号の反転が起こる可能性がある。

出力

```
$ ./q2_gamma.py 5.0 1000
answer= 119.990050164
time= 0:00:00.001901
$ ./q2_gamma.py 5.0 100000
answer= 119.999900006
time= 0:00:00.224297
$ ./q2_gamma.py 5.0
```

```
answer= 119.999990003
time= 0:00:02.041776
$ ./q2_gamma.py 5.0 10000000
answer= 119.999999113
time= 0:00:23.263810
$ ./q2_gamma.py 2.5
answer= 3.3233506936
time= 0:00:02.396380
$ ./q2_gamma.py -1.9
answer= -10.5705632264
time= 0:00:02.311163
```