# Assignment1

刘兆琦　学号 2201110045

2022 年 9 月 17 日

## 一、Monte Carlo integration

### 1.1 问题描述

数值计算积分

$$I(f)=\int_0^1 \sin(x)dx=\mathbb{E}(\sin X) \tag{1}$$

并分析 Monte Carlo 半阶收敛性质。

### 1.2 数值计算

在 [0,1] 中均匀取 $N$ 个点, 计算 $\mathbb{E}(\sin X)=\frac{1}{N}\sum_{i=1}^{N}\sin X_i$, 重复 $M$ 次。计算平均误差

$$\mathbb{E}(|e_n|)=\frac{1}{M}\sum_{j=1}^{M}|\bar{X}_j-(1-\cos(1))|$$

取 $N=\{10,100,1000,10000,100000\}$, 计算收敛阶

$$o_k=\frac{\log(e_{n_{k+1}}/e_{n_k})}{\log(n_{k+1}/n_k)}$$

结果如下表。

表 1: Monte Carlo 积分误差即收敛阶

| N | error | order |
|---|---|---|
| 10 | 0.65872412 | NaN |
| 100 | 0.20495824 | 0.50703819 |
| 1000 | 0.06347936 | 0.50903286 |
| 10000 | 0.01962411 | 0.50984243 |
| 100000 | 0.00630419 | 0.49316046 |

## 二、limit process

### 2.3 问题描述

数值分析 Binomial,Possion,Normal distribution 的极限性质。

## 2.4 数值计算

数值计算 Possion 分布 $P(\lambda)$ 的 pmf, 舍弃掉密度小于 $1e-6$ 的点。计算 Binomial 分布 $B(N,\lambda/N)$ 的 pmf, 并计算二者的偏差

$$loss_{\lambda,N} = \sum_{i=0}^{\infty} |\mathbb{P}(P(\lambda)=i) - \mathbb{P}(B(N,\lambda/N)=i)| \tag{2}$$

通过增大 $N$ 可以找到使得 $loss_{\lambda,N} < \epsilon$ 的最小的 $N$。取 $\epsilon = 0.01$, 结果如下。

表 2: Binomial 分布逼近 Possion 分布要求的 N

| $\lambda$ | $N/\lambda$ | $\lambda$ | $N/\lambda$ |
|---|---|---|---|
| 1 | 56 | 6 | 48.2 |
| 2 | 46 | 7 | 49.1 |
| 3 | 50.6 | 8 | 49.6 |
| 4 | 50.7 | 9 | 49.6 |
| 5 | 49.6 | 10 | 49.5 |

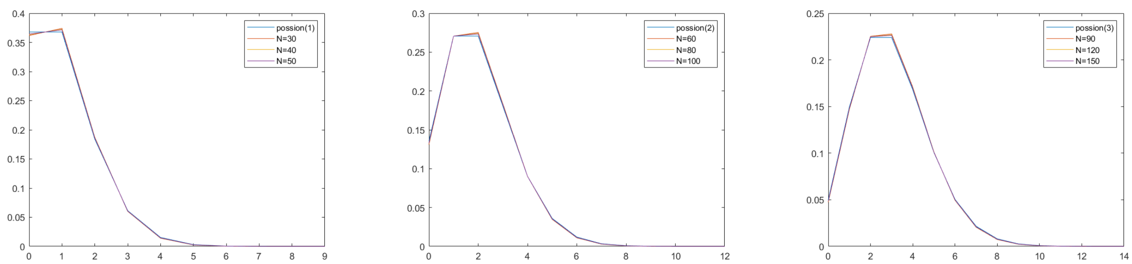
可以看出当 $p \simeq 0.02$ 时，大致有 $loss < 0.01$。

对正态分布 $N(0,1)$ 计算

$$p_{\lambda,i} = \int_{x_i-1/(2\sqrt{\lambda})}^{x_i+1/(2\sqrt{\lambda})} \frac{1}{\sqrt{2\pi}} e^{-x^2/2} dx \tag{3}$$

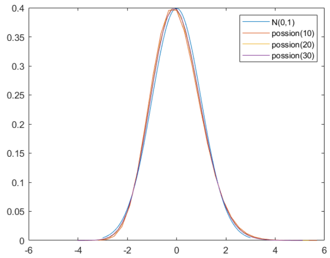
其中 $x_i = \frac{i-\lambda}{\sqrt{\lambda}}$. 计算 $loss_\lambda = \sum_{i=0}^{\infty} |\mathbb{P}(P(\lambda)=i) - p_{\lambda,i}|$。通过增大 $\lambda$ 来减少 $loss_\lambda$。取 $\epsilon = 0.05$, 在 $\lambda = 26$ 时有 $loss_\lambda < \epsilon$。

## 2.5 画图




(a) λ=1 (b) λ=2 (c) λ=3

Binomial 逼近 Possion




Possion 逼近正态分布

# 三、代码

## 3.6 数值积分

```
import numpy as np

N=[10,100,1000,10000,100000]
nlist=np.array(N)
dlist=np.array([])
olist=np.array([])

for n in nlist:
    d=0
    for k in range(1000):
        x=np.random.rand(n)
        y=np.sin(x)
        i=sum(y)/n
        d+=abs(i-(1-np.cos(1)))
    dlist=np.append(dlist,d/100)

olist=np.log(dlist[0:-1]/dlist[1:])/np.log(nlist[1:]/nlist[0:-1])
print('order:',olist)
print('error',dlist)
```

## 3.7 分布的极限

```
    global epsilon;
    epsilon = 1e-6;
    ratiob=[];

    %% binomial逼近possion
```

```
for lambda=1:10
    t=possion(lambda);
    nmin=t(1,1);
    nmax=t(end,1);
    loss=1;
    N=lambda+1;

    while(loss > 0.01)
        N=N+1;
        p=lambda/N;
        b=binomial(N,p,[nmin,nmax]);
        while(b(end,1)<nmax)
            b(end+1,:)=[b(end,1)+1;0];
        end
        loss=sum(abs(t(:,2)-b(:,2)));
    end
    ratiob(end+1)=N/lambda;
end
ratiob

%% possion 逼近Guassian
lambda = 10;
gaussianpdf=@(mu,sigma,x)1/sqrt(2*pi*sigma)*exp(-(x-mu).^2/2/sigma^2);

loss=1;
while(loss > 0.05)
    lambda=lambda+1;
    t=possion(lambda);
    s=sqrt(lambda);
    x=(t(:,1)-lambda)/s;
    probg=zeros(size(x));
    parfor i=1:numel(x)
        probg(i)=integral(@(xx)gaussianpdf(0,1,xx),x(i)-1/s/2,x(i)+1/s/2);
    end
    loss=sum(abs(t(:,2)-probg));
end

lambda

%% 画图
figure(10);
```

```
x=-3:0.01:3;
y=gaussianpdf(0,1,x);
plot(x,y);
for lambda = 10:10:30
    t=possion(lambda);
    s=sqrt(lambda);
    x=(t(:,1)-lambda)/s;
    y=t(:,2)*s;
    hold on;
    plot(x,y);
end
legend('N(0,1)','possion(10)','possion(20)','possion(30)');

for lambda=1:3
    figure(lambda);
    t=possion(lambda);
    nmin=t(1,1);
    nmax=t(end,1);
    plot(t(:,1),t(:,2));
    for N= (30:10:50) * lambda
        hold on;
        b=binomial(N,lambda/N,[nmin,nmax]);
        plot(b(:,1),b(:,2));
    end
    legend(['possion(',num2str(lambda),')'],['N=',num2str(30*lambda)],...
        ['N=',num2str(40*lambda)],['N=',num2str(50*lambda)])
end

function mf=possion(lambda)
    global epsilon
    i=0;
    mf=[];
    p=exp(-lambda);
    while(p<epsilon)
        i=i+1;
        p=p*lambda/i;
    end

    while(p>=epsilon)
        mf(end+1,:)=[i;p];
        i=i+1;
```

```
            p=p*lambda/i;
        end
    end

    function mf=binomial(n,p,range)
        q=1-p;
        nmin=max(0,range(1));
        nmax=min(n,range(2));
        mf=zeros(nmax-nmin+1,2);
        prob=nchoosek(n,nmin)*p^nmin*q^(n-nmin);
        for i = 1:nmax-nmin+1
            mf(i,:)=[i-1+nmin;prob];
            prob=prob*p/q/(i+nmin)*(n-i+1-nmin);
        end
    end
```