统计计算 张楠 2019秋: Methods for Generating Random Variables

(返回 统计计算 张楠 2019秋)

统计计算中一个基础问题就是从特定的概率分布中产生随机变量(随机数). 在最简单的情形, 从一个有限总体中产生一个随机观测, 就需要一种从离散均匀 总体中产生随机观测的方法. 从而, 一个合适的均匀(伪)随机数产生器是从根本上所需要的, 从其他概率分布中产生随机数都依赖于均匀随机数产生器.

目录

Generating Uniform(0,1) random number

线性同余发生器: $X_{n+1} = aX_n + c \ (mod) \ m$ 这里:

(1) m: 模$= 2^{31} - 1$(32位计算机可以直接表示的最大素数).

(2) a: 乘数, 要小心选择

(3) c: 增量(可以)=0.

(4) X_0: 初始值(种子).

- $X_n \in \{0, 1, \ldots, m-1\}$, $U_n = X_n/m$.
- 使用奇数作为种子.
- 代数和群理论有助于a的选择.
- 希望生成器的周期很大.
- 不要产生多与$m/1000$个数.

在R中, 使用帮助主题来了解**.Random.seed**或者**RNGkind**关于缺省均匀随机数 产生器的详细信息. 各种不同类型的随机数产生器及其性质可以参考一些数值计算方面的资料.

Random Generators of Common Probability Distribution in R

在 R 中均匀伪随机数的产生器是**runif**. 产生伪均匀随机数方式为

```
runif(n)        #产生0到1上的长度为n的向量
runif(n,a,b)    #产生a到b上的长度为n的向量
matrix(runif(n*m),nrow=n,ncol=m)  #产生0到1上的nxm的矩阵
```

常用的一元概率分布的概率质量函数()或者概率密度函数(), 累积分布函数(), 分位数函数()以及随机数 产生器函数在R中已经集成, 比如对二项分布, 可以参看帮助文档

```
dbinom(x, size, prob, log = FALSE)
pbinom(q, size, prob, lower.tail = TRUE, log.p = FALSE)
qbinom(p, size, prob, lower.tail = TRUE, log.p = FALSE)
rbinom(n, size, prob)
```

R中常见的一元分布函数

分布	cdf	随机数产生器	参数
beta	pbeta	rbeta	shape1, shape2
二项分布	pbinom	rbinom	size, prob
χ^2分布	pchisq	rchisq	df
指数分布	pexp	rexp	rate
F分布	pf	rf	df1,df2
gamma	pgamma	rgamma	shape,rate or scale
几何分布	pgeom	rgeom	prob
对数正态分布	plnorm	rlnorm	meanlog,sdlog
负二项分布	pnbinom	rnbinom	size, prob
正态分布	pnorm	rnorm	mean,sd
Poisson分布	ppois	rpois	lambda
t分布	pt	rt	df
均匀分布	punif	runif	min,max

使用**sample**函数从一个有限离散总体中抽样：

sample可以实现从有限总体中有放回或者不放回两种抽样方式进行抽样.

```
#掷硬币
sample(0:1,size=10,replace=TRUE)
#字母a-z的一个置换
sample(letters)
#多项分布抽样
x<-sample(1:3,size=100,replace=TRUE,prob=c(0.2,0.3,0.5))
#> table(x)
#x
# 1  2  3
#12 30 58
```

The Inverse Transform Method

连续型场合

生成随机数的逆变换方法是基于以下熟知的定理

[Probability Integral Transformation]

若X为连续型随机变量, 其cdf 为F_X, 则 $U = F_X(X) \sim U(0,1).$

证: [折叠]

定义

$$F_X^{-1}(u) = \inf\{x : F_X(x) = u\}, \quad 0 < u < 1.$$

若随机变量$U \sim U(0,1)$, 则对所有$x \in \mathcal{R}$, 有

$$\begin{aligned} P(F_X^{-1}(U) \leq x) &= P(\inf\{t : F_X(t) = U\} \leq x) \\ &= P(U \leq F_X(x)) = F_U(F_X(x)) = F_X(x). \end{aligned}$$

因此$F_X^{-1}(U)$和随机变量X同分布.

从而, 若要产生X的一个随机观测x, 可以

1. 导出逆变换函数 $F_X^{-1}(u)$.
2. 从均匀分布$U(0,1)$中产生一个随机数u, 令$x = F_X^{-1}(u)$.

这种方法要求F的逆函数要容易求出.

例：使用逆变换方法产生连续型密度$f(x) = 3x^2 I(0 < x < 1)$的随机观测值.

此处$F_X(x) = x^3, (0 < x < 1)$, 因此 $F_X^{-1}(u) = u^{1/3}$, 因此

```
n<-1000
u<-runif(n)
x<-u^(1/3)
hist(x,prob=TRUE,main=expression(f(x)==3*x^2))
y<-seq(0,1,0.01)
lines(y,3*y^2)
```

绘图中数学符号的表示更多内容参看帮助文档**?plotmath**.

例: 使用逆变换方法产生指数分布的随机数.

$X \sim Exp(\lambda)$, 因此对$x > 0, F_X(x) = 1 - e^{-\lambda x}$, 从而 $F_X^{-1}(u) = -\frac{1}{\lambda} log(1-u)$. 注意到$U$和$1-U$同分布, 因此产生参数是$\lambda$的指数分布的长度为$n$的随机数命令为

```
-log(runif(n)) / lambda
```

在R中也可以使用**rexp**来产生.

离散型场合

设X为一离散型随机变量, 其可能取值记为

$$\cdots < x_{i-1} < x_i < x_{i+1} < \cdots$$

定义

$$F_X^{-1}(u) = \inf\{x : F_X(x) \geq u\}$$

则逆变换为$F_X^{-1}(u) = x_i$, 其中$F_X(x_{i-1}) < u \leq F_X(x_i)$. 从而产生一个随机数$x$的方式为

1. 从均匀分布$U(0,1)$中产生一个随机数u
2. 取$x = x_i$, 若$F_X(x_{i-1}) < u \leq F_X(x_i)$.

对某些分布来说, 计算$F_X(x_{i-1}) < u \leq F_X(x_i)$可能比较困难. 在离散型 场合应用逆变换方法产生随机数的不同方式可以参考**Devroye**第三章.

例: 使用逆变换方法产生0-1分布的随机数.

此时 $F_X(0)=f_X(0)=1-p$ 以及 $F_X(1)=1$. 因此若 $u>1-p$, 则 $F_X^{-1}(u)=1$; 若 $u\le 1-p$, 则 $F_X^{-1}(u)=0$. 因此

```
n<-1000
p<-0.4
u<-runif(n)
x<-as.integer(u>1-p)
mean(x) #理论值 p
var(x)  #理论值p(1-p)
```

在R中,我们使用**rbinom(n,1,p)**来产生0-1分布长为n的一个随机观测向量.

例: 使用逆变换方法产生几何分布的随机数.

由于几何分布的分布函数$F_X(x)=1-q^x,\ \ x=1,2,\cdots$. 因此每个随机数 都要计算

$$1-q^{x-1}<u\le 1-q^x.$$

这个不等式等价于$x-1<log(1-u)/log(q)\le x$, 其解为$x=\lceil log(1-u)/log(q)\rceil$, 这里$\lceil t\rceil$表示不小于$t$的最小整数(ceiling). 因此

```
n<-1000
p<-0.4
u<-runif(n)
x <- ceiling(log(1-u) / log(1-p)) - 1
```

注意到U和$1-U$同分布, 以及$log(1-U)/log(q)$取整数的概率为0, 从而上述代码最后一步可以等价为

```
x <- floor(log(u) / log(1-p))
```

在R中,我们使用**rgeom(n,p)**来产生参数为p的几何分布的长为n的一个随机观测向量.

The Acceptance-Rejection Method

假设X和Y是随机变量, 其概率函数分别为f和g. 满足

$$\frac{f(t)}{g(t)}\le c,\quad \forall\ t\ \ s.t.\ \ f(t)>0$$

则舍选法(Acceptance-Rejection Method)可以用来生成X的随机数.

1. 找一个可以方便生成随机数的随机变量Y, 其概率函数g满足$f(t)/g(t)\le c, \forall\ t\ s.t.\ f(t)>0$.
2. 从g中产生一个随机数y.
3. 从均匀分布$U(0,1)$中产生一个随机数u.
4. 若$u<f(y)/(cg(y))$, 则接受$x=y$, 否则拒绝y. 重复2-4, 直至产生给定个数的x.

在离散型场合, 对每个使得$f(k)>0$的k有

$$P(k|A)=\frac{P(A|k)g(k)}{P(A)}=\frac{[f(k)/(cg(k))]g(k)}{1/c}=f(k)$$

对连续型随机变量场合, 即需证明$P(Y\le y|U\le \frac{f(Y)}{cg(Y)})=F_X(y)$. 事实上

$$
\begin{aligned}
P(Y \le y | U \le \frac{f(Y)}{cg(Y)}) &= \frac{P(U \le \frac{f(Y)}{cg(Y)}, Y \le y)}{1/c} \\
&= \int_{-\infty}^{y} \frac{P(U \le \frac{f(Y)}{cg(Y)} | Y = \omega \le y)}{1/c} g(\omega) d\omega \\
&= c \int_{-\infty}^{y} \frac{f(\omega)}{cg(\omega)} g(\omega) d\omega \\
&= F_X(y)
\end{aligned}
$$

舍选法的特点:

- 优点是计算时间不随着x增加;
- 缺点是寻求一个容易产生随机数且逼近f很好的分布g比较困难, 因此一个不好的g会导致接受概率很低,造成 运行时间过长.

例: 使用舍选法从如下分布产生随机数.

$$
f(x) = 6x(1-x), \quad 0 < x < 1
$$

从此分布中产生n个随机数需要循环的总数依赖于接受概率$1/c$的大小. 取$g(x)$为$U(0,1)$的密度, 则$c = 6$, 从而从g中产生的一个随机数x被接受, 除非

$$
\frac{f(x)}{cg(x)} = x(1-x) > u
$$

因此,平均来看需要cn次循环, 即$2cn$个随机数需要产生.

```
n<-1000
j<-k<-0
y<-numeric(n)
while(k<n){
  u<-runif(1)
  j<-j+1
  x<-runif(1) #从g中产生一个随机数
  if(x*(1-x)>u) {
    k<-k+1
    y[k]<-x
    }
  }
#>j
#[1] 6153
```

在这次模拟中, 需要6153次循环来产生$n = 1000$个需要的随机数. 比较其经验分位数和理论分位数

```
p<-seq(.1,.9,.1)
Qhat<-quantile(y,p)
Q<-qbeta(p,2,2)
se<-sqrt(p*(1-p)/(n*dbeta(Q,2,2)))
round(rbind(Qhat,Q,se),3)
```

当密度靠近0时需要大量的重复来估计分位数.

Transformation Methods

除前面介绍的方法外, 许多变换类型可以用来生成随机数. 比如

1. 若$Z \sim N(0,1)$, 则$Z^2 \sim \chi^2(1)$.
2. 若$U \sim \chi^2(m)$ 以及 $V \sim \chi^2(n)$, 则$F = \frac{U/m}{V/n} \sim F(m,n)$.
3. 若$Z \sim N(0,1)$ 以及 $V \sim \chi^2(n)$ 且相互独立, 则$T = \frac{Z}{\sqrt{V/n}} \sim t(n)$.
4. 若$U,V \sim U(0,1)$且相互独立, 则$Z_1 = \sqrt{-2logU}cos(2\pi V)$与$Z_2 = \sqrt{-2logU}sin(2\pi V)$ 相互独立的标准正态随机变量.

例: 使用变换法从产生对数分布的随机数.

注意到若$U,V \sim U(0,1)$ 且相互独立, 则 $X = \left\lfloor 1 + \frac{logV}{log(1-(1-\theta)^U)} \right\rfloor$ 服从参数为θ的对数分布. 从而

1. 从$U(0,1)$中产生u
2. 从$U(0,1)$中产生v
3. 取$x = \lfloor 1 + log(v)/log(1-(1-\theta)^u) \rfloor$

实现代码为

```
n<-1000
theta<-0.5
u<-runif(n)
v<-runif(n)
x<-floor(1+log(v)/log(1-(1-theta)^u))
k<-1:max(x)#计算对数分布的概率
p<--1/log(1-theta)*theta^k/k
se<-sqrt(p*(1-p)/n)
p.hat<-tabulate(x)/n
round(rbind(p.hat,p,se),3)
```

相比于逆变换方法, 变换法更有效率(为什么?)

```
rlogarithmic<-function(n,theta){
  stopifnot(all(theta>0 & theta<1))
  th<-rep(theta,length=n)
  u<-runif(n)
  v<-runif(n)
  x<-floor(1+log(v)/log(1-(1-theta)^u))
  return(x)
  }
```

Sums and Mixtures

随机变量的和或者混合是一种特殊类型的变换. 比如

例: 使用随机变量的和产生χ^2分布的随机数.

ν个标准正态随机变量的平方和为$\chi^2(\nu)$分布随机变量, 因此

```
n<-1000
nu<-2
X<-matrix(rnorm(n*nu),n,nu)^2
#方法1
y<-rowSum(X)
#方法2
y<-apply(X,MARGIN=1,FUN=sum)
```

对随机变量的混合, 我们先看定义:

称一个随机变量为离散混合, 如果其分布为某些随机变量$X_1, X_2, \cdots$分布的加权:

$$F_X(x) = \sum \theta_i F_{X_i}(x).$$

$\theta_i > 0$且$\sum \theta_i = 1$为权重.

而对连续型随机变量, 类似的有

称一个随机变量为连续混合, 如果其分布为某个分布族$X|Y = y$的加权:

$$F_X(x) = \int F_{X|Y=y}(x) f_Y(y) dy.$$

其中$\int f_Y(y)dy = 1.$

例: 产生如下混合分布的随机数.

$$F_X(x) = pF_{X_1}(x) + (1-p)F_{X_2}(x)$$

产生此随机数的方法显然

1. 产生一个整数$k \in \{1,2\}$, 这里$P(1) = p, P(2) = 1-p.$
2. 若$k = 1$, 则从F_{X_1}中产生x_1, 并令$x = x_1$.
3. 若$k = 2$, 则从F_{X_2}中产生x_2, 并令$x = x_2$.

例：Γ分布的混合.

$$F_X = \sum_{i=1}^5 \theta_j F_{X_j}$$

其中$X_j \sim Gamma(r = 3, \lambda_j = 1/j)$相互独立, $\theta_j = j/15,\ j = 1, \cdots, 5.$

```
n<-1000
k<-sample(1:5,size=n,replace=TRUE,prob=(1:5)/15)
rate<-1/k
x<-rgamma(n,shape=3,rate=rate)
#画混合的密度以及分量的密度
plot(density(x),xlim=c(0,40),ylim=c(0,.3),lwd=3,xlab=''x'',main='''')
for(i in 1:5)
lines(density(rgamma(n,3,1/i)))
```

例：Γ密度的混合.

$$f(x) = \sum_{j=1}^5 \theta_j f_j(x), x > 0$$

其中f_j为$Gamma(3, \lambda_j)$的密度. 为画此混合密度的图形, 需要计算其值

```
f<-function(x,lambda,theta){
 sum(dgamma(x,3,lambda)*theta)
 }
```

这里$dgamma(x, 3, lambda) * theta$是向量$(\theta_1 f_1(x), \cdots, \theta_5 f_5(x))$.

```
x<-seq(0,8,length=200)
dim(x)<-length(x) #使用apply函数需要
p<-c(.1,.2,.2,.3,.2)
lambda<-c(1,1.5,2,2.5,3)
#计算混合密度在x处的值
y<-apply(x,1,f,lambda=lambda,theta=p)
plot(x,y,type=''l'';,ylim=c(0,.85),lwd=3,ylab=''Density'')
for(j in 1:5){
 y<-apply(x,1,dgamma,shape=3,rate=lambda[j])
```

```
lines(x,y)
}
```

若$X|\Lambda=\lambda\sim Pois(\lambda),\Lambda\sim Gamma(r,\beta)$,则 $X\sim NB(r,p=\beta/(1+\beta))$. 本例说明一个Poisson-Gamma混合,并和负二项分布的样本相比较.

```
#Poisson-Gamma Mixture
n<-1000
r<-4
beta<-3
lambda<-rgamma(n,r,beta) #lambda是随机的
x<-rpois(n,lambda)
#compare with negative binomial
mix<-tabulate(x+1)/n
negbin<-round(dnbinom(0:max(x),r,beta/(1+beta)),3)
se<-sqrt(negbin*(1-negbin)/n)
round(rbind(mix,negbin,se),3)
```

本页面最后编辑于2019年10月10日 (星期四) 21:23。