Numerical and Statistical Methods for Finance

Random Numbers

Pierpaolo De Blasi

University of Torino
email: `pierpaolo.deblasi@unito.it`
webpage: `sites.google.com/a/carloalberto.org/pdeblasi/`

Lecture no. 5
7 November 2013

Pseudorandom Number Generation

The building block of a simulation study is the ability to generate random numbers,

$$U_1, U_2, \ldots \sim \text{iid Unif}(0, 1)$$

We shall be contented to use *pseudorandom numbers*, i.e. a sequence of values which, although they are deterministically generated, have all the appearances of being independent Unif$(0, 1)$ random variables.

One of the most common approaches to generate pseudorandom numbers starts with

- an initial seed x_0;
- two integers a and m;
- recursively compute successive values of x_n, $n = 1, 2, \ldots$, as

$$x_n = (ax_{n-1}) \bmod m$$

- set $u_n = x_n/m$

Integer division and modulus

Let x, y be two integers. x mod y is the remainder of the integer division of x by y. For example, for $x = 17$ and $y = 4$,

```
> x%/%y   # integer division = 4
> x%%y    # x mod y = 1
```

questo fa la divisione di x con y e poi lo arrotonda al più piccolo numero intero

Questo e quello che ci interessa per generare un PRN
(aXn-1)%%m ------aXn-1 mod m

according to the formula

```
x == y * ( x %/% y )+(x %% y)
```

It is clear that x mod y is either $0, 1, \ldots, y - 1$.

Pseudorandom Number Generation

Since each of the numbers x_n assumes one of values $0, , \ldots, m-1$, after some finite number (of at most m) of iterations a value must repeat itself; and once this happens, the whole sequence will begin to repeat.

```
N<-101
x.0<-17
x<-array(0,N)
a<-2;m<-100

x[1]<-(a*x.0)%%m
```

sarebbe modulo, per es
x %% y

```
for(i in 2:N){
    x[i]<-(a*x[i-1])%%m
}
u<-x/m
```

```
> matrix(x[2:N],nrow=20)

      [,1] [,2] [,3] [,4] [,5]
 [1,]   68   68   68   68   68
 [2,]   36   36   36   36   36
 [3,]   72   72   72   72   72
 [4,]   44   44   44   44   44
 [5,]   88   88   88   88   88
...
[16,]   24   24   24   24   24
[17,]   48   48   48   48   48
[18,]   96   96   96   96   96
[19,]   92   92   92   92   92
[20,]   84   84   84   84   84
```

It follows $u_1, \ldots, u_N$ with $N > 20$ will feature ties, which is not consistent with them being independent Unif$(0, 1)$.

Pseudorandom Number Generation

The goal is then to use constants a and m to satisfy

1. for any initial seed, the resultant sequence has the appearance of being a sequence of independent $\text{Unif}(0, 1)$ random variables;
2. for any initial seed, the number of values that can be generated before repetition begins is large;
3. the values can be computed efficiently on a computer.

It turns out that m should be chose to be a large prime number. It has been shown that for a 32-bit word machine, $m = 2^{31} - 1$ results in desirable properties. See help pages

```
> ?randu
> ?RNGkind()
```

In the following, we rely on the random generator used by R as a "black block" that gives a random number on request.

Monte Carlo Integration

Let $g(x)$ be a function and suppose we wanted to compute

$$\theta = \int_0^1 g(x)\mathrm{d}x$$

Note that if $U \sim \text{Unif}(0, 1)$, we can express the integral θ as

$$\theta = E[g(U)]$$

If $U_1, \ldots, U_N$ are iid $\text{Unif}(0, 1)$, then $g(U_1), \ldots, g(U_N)$ are iid random variables with mean θ. By the strong law of large number

$$\frac{1}{N}\sum_{n=1}^{N} g(U_n) \to E[g(U)] = \theta \quad \text{as } N \to \infty$$

Hence we can approximate θ by generating a large number of random numbers u_n and taking as our approximation the sample mean of $g(u_n)$.

This approach to approximating integrals is called *Monte Carlo integration*.

Ex 3 page 46, Ross (2006)

$$\theta = \int_0^1 \exp(e^x)\mathrm{d}x$$

e^x

```
> set.seed(1)
> N<-1000
> u<-runif(N)
> g<-function(x) exp(exp(x))
> mean(g(u))
[1] 6.325002
```

Compare the result with numerical integration:

```
integrate(f, lower, upper, ..., subdivisions=100,
          rel.tol = .Machine$double.eps^0.25, abs.tol = rel.tol)
```

where

- `f` is a R function taking a numeric first argument and returning a numeric vector of the same length
- `lower` and `upper` are the limits of integration. When integrating over infinite intervals, use `Inf` or `-Inf`.
- `rel.tol` and `abs.tol` are the relative and absolute accuracy, respectively.

```
> integrate(g,0,1)
[1] 6.316564 with absolute error < 7e-14
```

Monte Carlo Integration

$x = a + [b-a]y$

If we want to compute

$$\theta = \int_a^b g(x)\mathrm{d}x$$

then, by making the substitution $\boxed{y = (x-a)/(b-a)}$, $\mathrm{d}y = \mathrm{d}x/(b-a)$,

$$\frac{d}{dy} = b-a$$

$$\begin{aligned}\theta &= \int_0^1 g(a+[b-a]y)\,(b-a)\mathrm{d}y \\ &= \int_0^1 h(y)\mathrm{d}y\end{aligned}$$

where $h(y) = (b-a)\,g(a+[b-a]y)$.

Ex 5 page 47, Ross (2006)

$$\theta = \int_{-2}^{2} e^{x+x^2}\mathrm{d}x$$

```
> set.seed(3)
> N<-10000
> u<-runif(N)
> g<-function(x) exp(x+x^2)
> h<-function(y) (2-(-2))*g(-2+(2-(-2))*y)
> mean(h(u))
[1] 92.35895
```

Monte Carlo Integration

When one of the limit of integration is ∞, as in

$$\theta = \int_0^\infty g(x)\mathrm{d}x$$

we can make the substitution $y = 1/(x+1)$,
$\mathrm{d}y = -\mathrm{d}x/(x+1)^2 = -y^2\,\mathrm{d}x$ to obtain the identity

$$\theta = \int_0^1 h(y)\mathrm{d}y = \int_0^1 g\left(\left(\frac{1}{y}-1\right)\right)/y^2$$

where $h(y) = g(\frac{1}{y} - 1)/y^2$.

Ex 7 page 47, Ross (2006)

$$\theta = \int_{-\infty}^{\infty} e^{-x^2}\mathrm{d}x$$

$$\left(=\sqrt{\frac{2\pi}{2}}\right)$$

knowing this

genera N uniform random numers

$= e^{-x^2}$

```
> set.seed(1)
> N<-100000
> u<-runif(N)
> g<-function(x) exp(-x^2)
> h<-function(y) g(1/y-1)/(y^2)
> 2*mean(h(u))    # since g(-x)=g(x)
[1] 1.770098
> sqrt(2*pi/2)
[1] 1.772454
```

!!! !!!

simile al risult precedent

Monte Carlo Integration

Suppose that g is a function of d variables (say $d = 2$) and we are interested in computing the 2-dimensional integral

$$\theta = \int_0^1 \int_0^1 g(x_1, x_2) \mathrm{d}x_1 \, \mathrm{d}x_2$$

We can estimate θ by expressing it as the following expectation

$$\theta = E[g(U_1, U_2)]$$

where U_1 and U_2 are independent $\text{Unif}(0, 1)$ random variables.

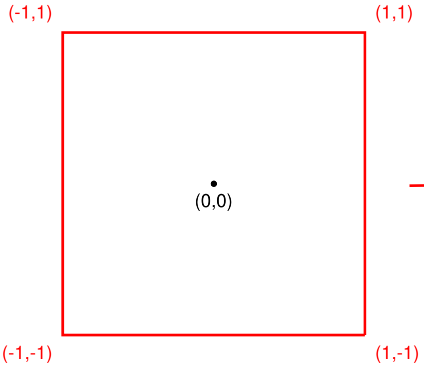
If we generate N independent sets, each consisting of $d = 2$ iid $\text{Unif}(0, 1)$ random variables:

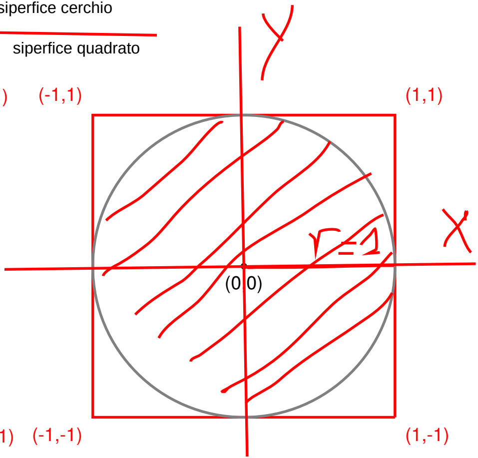
$$\begin{array}{cc} U_1^1, & U_2^1 \\ U_1^2, & U_2^2 \\ \vdots & \vdots \\ U_1^N, & U_2^N \end{array}$$

since $g(U_1^n, U_2^n)$ are all iiid random variables with mean θ, we can approximate θ by $N^{-1} \sum_{n=1}^{N} g(U_1^n, U_2^n)$.

Estimation of π

$$P(x^2+y^2 \leq 1) = \frac{\text{siperfice cerchio}}{\text{siperfice quadrato}}$$





$$x^2+y^2=r^2$$

circonferenza

Estimation of π

Suppose that the random vector (X, Y) is uniformly distributed in the square of area 4 centered at the origin. What is the probability that this random point is contained within the circle of radius 1?

$$\theta = \Pr\{(X, Y) \text{ is in the circle}\} = \Pr\{X^2 + Y^2 \leq 1\}$$

$x^2 + y^2 = r^2$

$r = 1$

$$= \frac{\text{Area of the circle}}{\text{Area of the square}} = \frac{\pi}{4}$$

i.e. the double integral $\int_{-1}^{1} \int_{-1}^{1} g(x, y)\, \mathrm{d}x\, \mathrm{d}y$ where

$$\boxed{g(x, y) = \mathbb{1}_{(x^2+y^2 \leq 1)} = \begin{cases} 1 & \text{if } x^2 + y^2 \leq 1 \\ 0 & \text{otherwise} \end{cases}}$$

Hence if we generate a large number of random points in the square, the proportion of points that fall within the circle will be $\approx \pi/4$.

AREA $= \pi \cdot r^2 = \pi$

$x \sim U(-1\ 1) \sim 2 \cdot U(0,1) - 1 \sim$
$\sim U(0,2) - 1 \sim U(-1\ 1)$

Estimation of π

If X, Y are independent and both Unif$(-1, 1)$, their joint density is

$$f(x, y) = f(x)f(y) = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}, \quad -1 \leq x, y \leq 1$$

Hence (X, Y) is uniformly distributed in the square (*why?*)

We can generate Unif$(-1, 1)$ as $2U - 1$ (*why?*), hence if we generate $U_1, U_2 \sim$ iid Unif$(0, 1)$ and we set $X = 2U_1 - 1$, $Y = 2U_2 - 1$, then

$$E[g(X, Y)] = E[\mathbb{1}_{(X^2+Y^2 \leq 1)}] = \frac{\pi}{4}$$

generate many numbers inside the circle that are distrib as unif 01 and such that

$\boxed{x^2 + y^2 \leq 1}$

```
> set.seed(1)
> N<-1000
> u1<-runif(N);u2<-runif(N)
> Indic<-( (2*u1-1)^2+(2*u2-1)^2 ) <= 1
> mean(Indic)
[1] 0.77
> pi/4
[1] 0.7853982
```

calculate the mean of this numbers inside the circle and this will give us the probability teta.

$x = 2U_1 - 1$
$y = 2U_2 - 1$

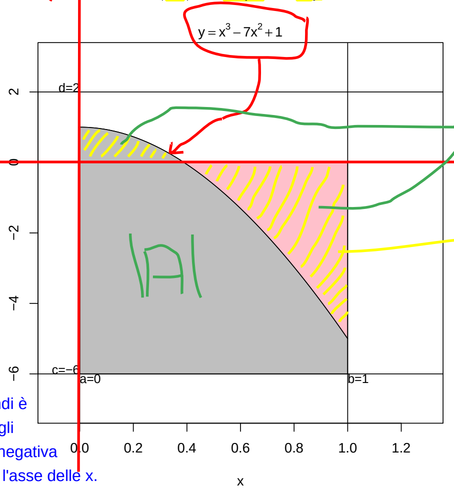
Hit-and-miss Method

Alternative (and slower) way to calculate $\theta = \int_a^b f(x)\mathrm{d}x$.

Let c and d be such that $c < f(x) < d$ for all $x \in [a, b]$. Let A be the set bounded above by the curve and by the box $[a, b] \times [c, d]$, then

$$\theta = |A| + c(b - a)$$

= alla parte rigata di gialla nel grafico.



questa area in giallo sarebbe il teta ovvero quello che vogliamo trovare. (non solo quella sopra anche quella sotto)

consideriamo la parte negativa di A che sarebbe quella sotto 0, noi lo prendiamo in valore assoluto quindi è positiva. Se a questa parte positiva gli aggiungiamo c(b-a) che di per se è negativa allora otteniamo la parte gialla sotto l'asse delle x.

Hit-and-miss Method

Thus if we estimate $|A|$, then we can estimate θ.

To estimate $|A|$, we take $X \sim \text{Unif}(a, b)$ and $Y \sim \text{Unif}(c, d)$ so that (X, Y) is uniformly distributed over the box $[a, b] \times [c, d]$, and

$$\boxed{\Pr\{(X, Y) \in A\} = \frac{|A|}{(b-a)(d-c)}}$$

---from this we can calculate |A|
since we are able to calc
Pr(XYelemA)
this is the box

Let $Z = \mathbb{1}_{((X,Y)\in A)}$, then $E(Z) = \Pr\{(X, Y) \in A\}$ and we have

$$\theta = E(Z)(b-a)(d-c) + c(b-a)$$

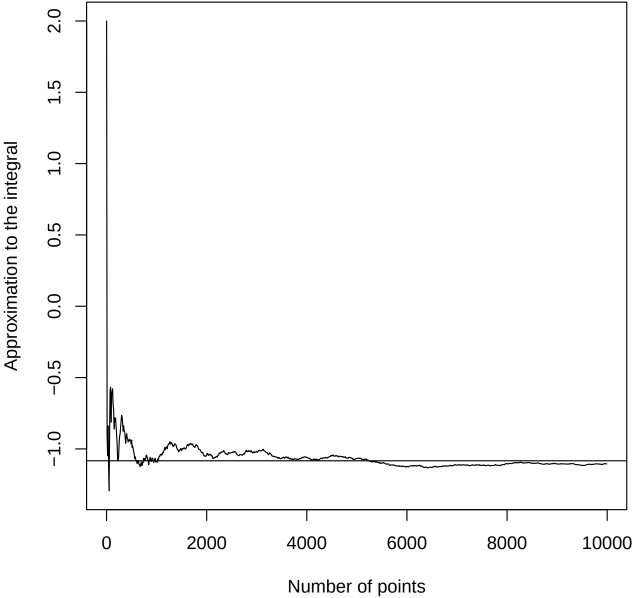
```
set.seed(1)
N<-10000
ftn<-function(x) x^3-7*x^2+1
a<-0;b<-1;c<--6;d<-2
X<-a+runif(N)*(b-a)
Y<-c+runif(N)*(d-c)
Z<-(ftn(X) >= Y)
I<-(b-a)*c+mean(Z)*(b-a)*(d-c)
cat("I=",I)
I= -1.1072
```

Note that

$$\int_0^1 (x^3 - 7x^2 + 1)\mathrm{d}x$$
$$= (x^4/4 - 7x^3/3 + x)_0^1$$
$$= -13/12 = -1.0833$$

and $f(0) = 1 < 2$, $f(1) = -5 > -6$.

Hit-and-miss Method



Exercises

- Chapter 3, Ross (2006)
  Ex 1, Ex 2, Ex 4, Ex 6, Ex 7, Ex 8, Ex 9
- Chapter 19, Owen, et al. (2007)
  Ex 2, Ex 3

Resources

- BOOKS
  - Owen J., Maillardet R. and Robinson A. (2009).
    Introduction to Scientific Programming and Simulation Using R.
    Chapman & Hall/CRC.
  - Ross, S. (2006).
    Simulation. 4th edn. Academic Press.
- WEB
  - R software:
    `http://www.r-project.org/`
  - Owen, et al. (2009):
    `http://www.ms.unimelb.edu.au/spuRs/`