Simulation methods

Yifan Jin

28/06/2020

```
# install.packages("matlab")
library(matlab)
```

```
##
## Attaching package: 'matlab'
## The following object is masked from 'package:stats':
##
##     reshape
## The following objects are masked from 'package:utils':
##
##     find, fix
## The following object is masked from 'package:base':
##
##     sum
```

1. Transformation methods

Theory

If U uniformly distributed on (0,1), then 3U+5 is uniform on (5,8).

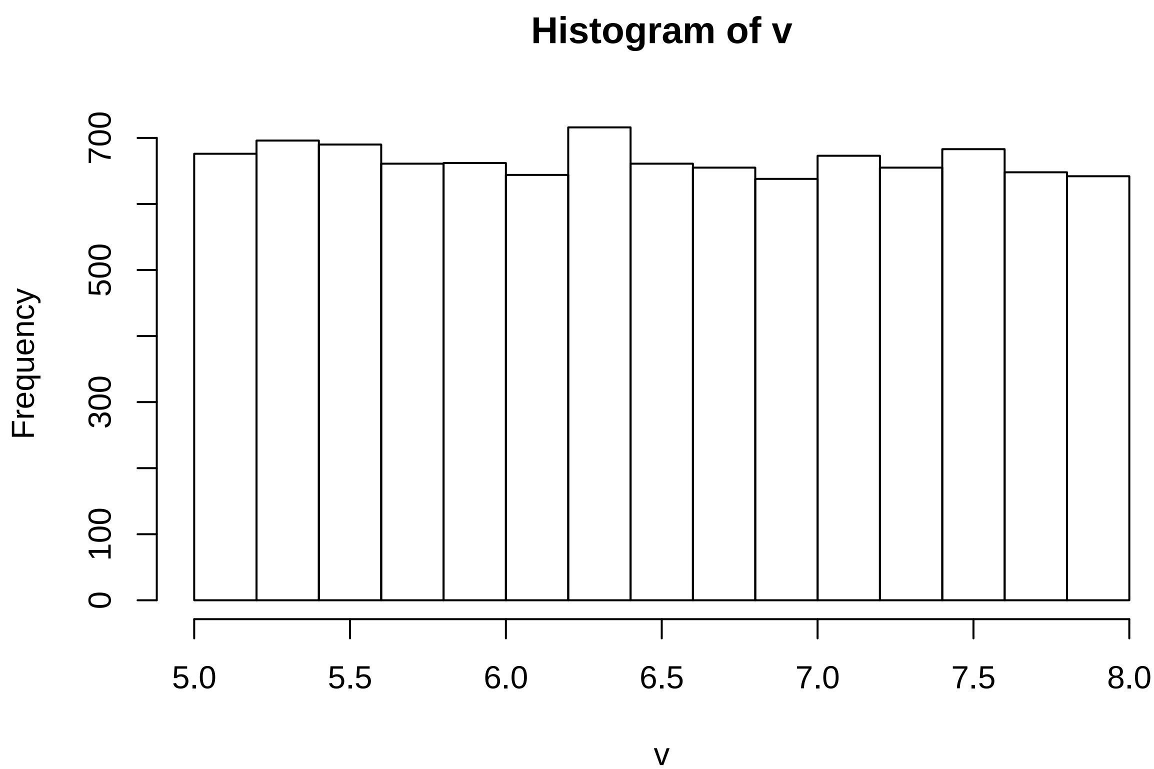
If Z is standard normal, then u+σZ is a normal distribution with mean u and variance σ^2

Sum of squared normals is chi-squared disrtibution with df=number of normals

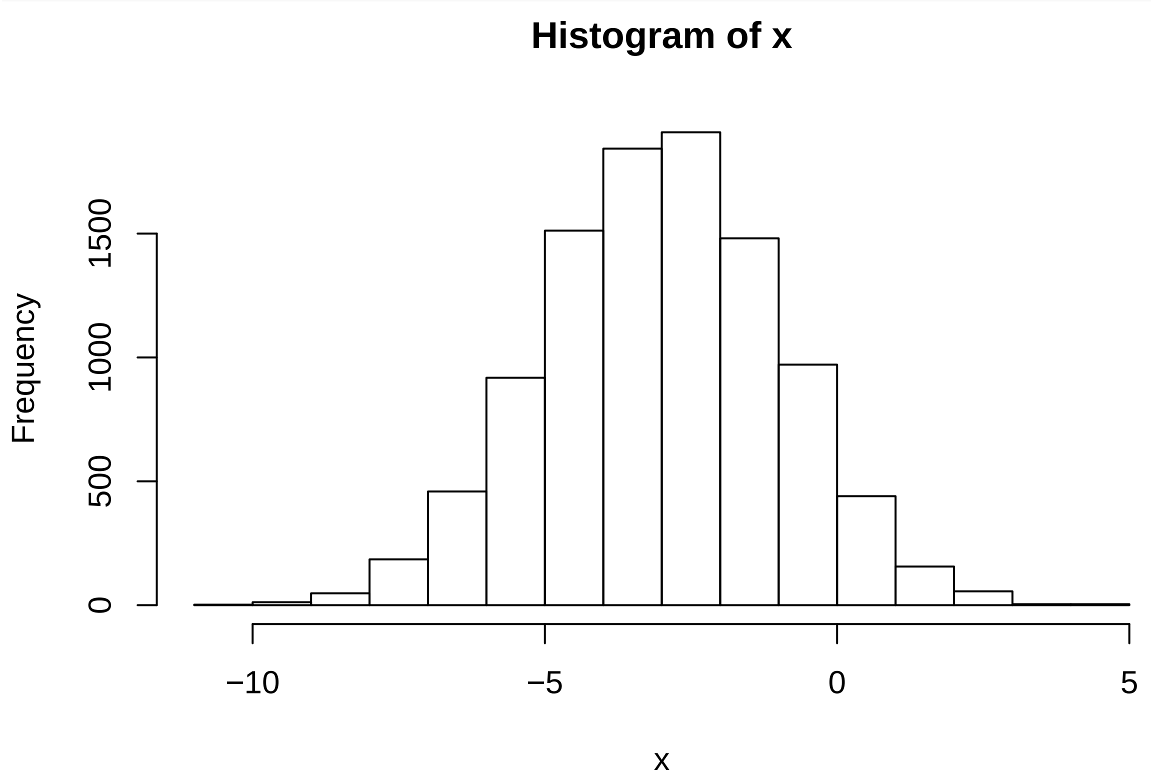
```
set.seed(999)
S=10000
3*runif(1)+5 #This gives the draw from U(5,8)
```

```
## [1] 6.167214
```

```
v=3*runif(S)+5 #10000 draws from U(5,8)
hist(v)
```

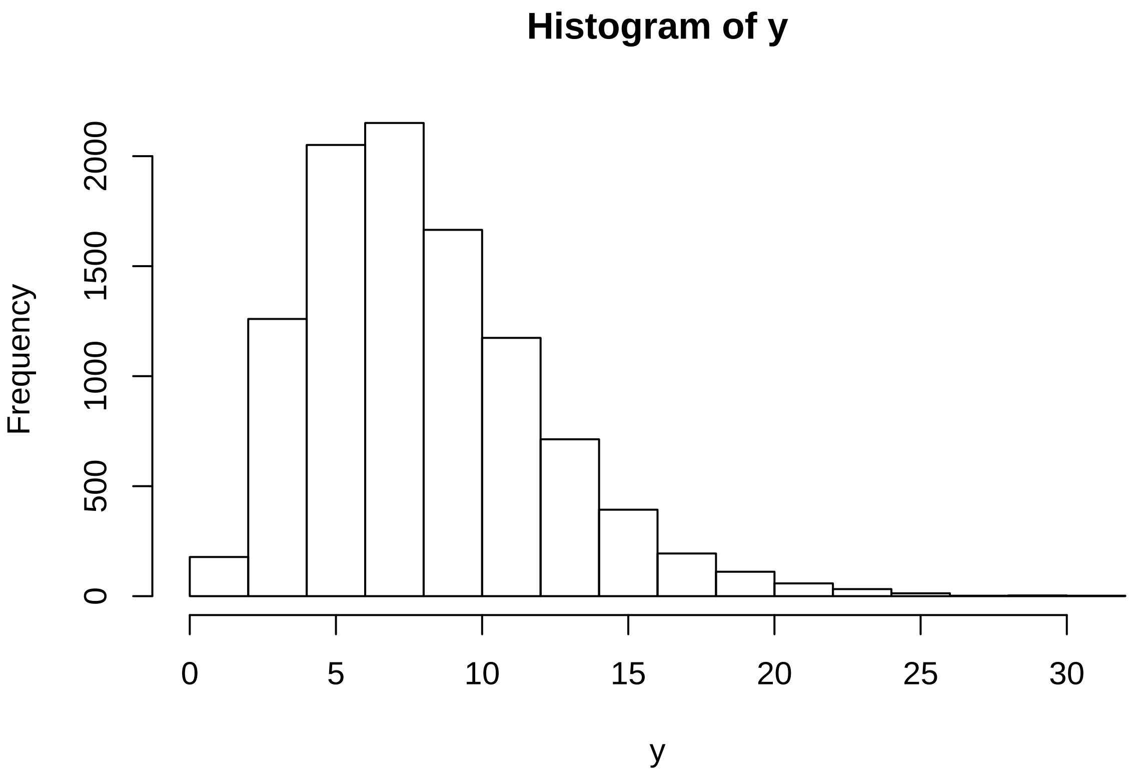



```r
# from N(0,1) to N(-3,2^2)
x=2*rnorm(S)-3
hist(x)
```




```r
z=matrix(rnorm(S*8),nrow=S,ncol=8)
y=matrix(rowSums(z^2),nrow = S)
```

```r
hist(y)
```



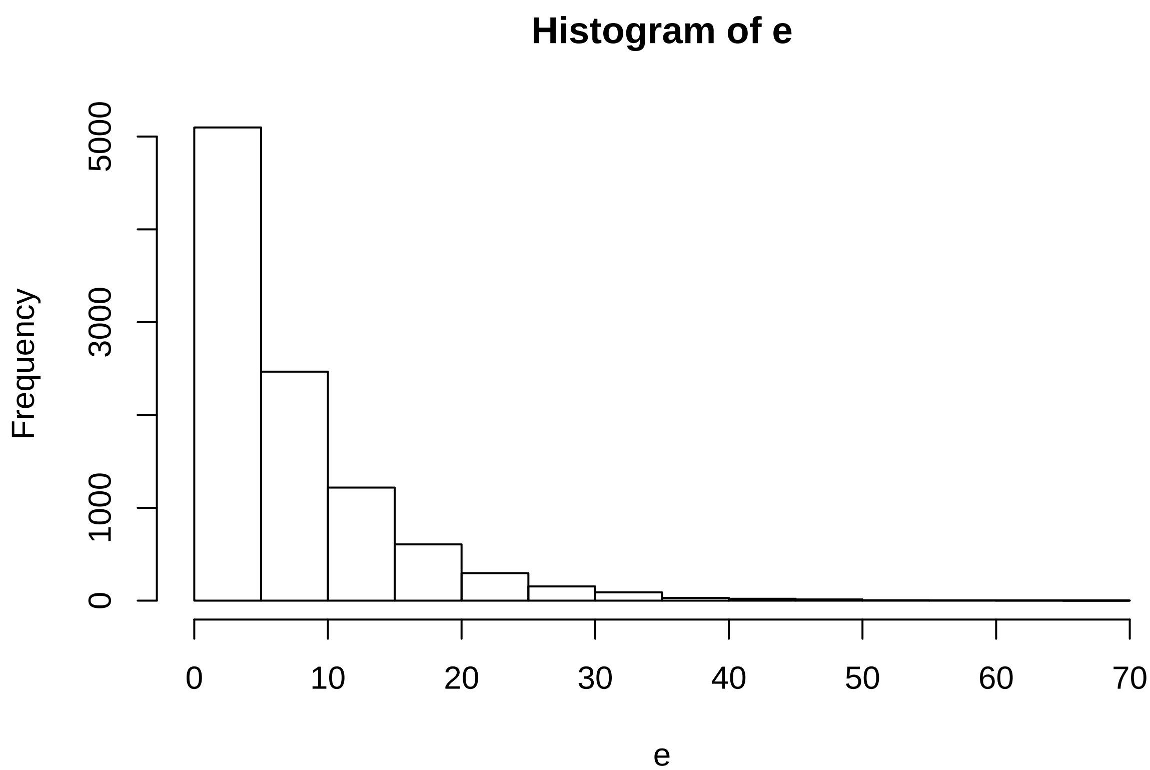

```r
# multivariate normal
mu=matrix(c(-3,1),nrow=2)
sigma=matrix(c(1,0.8,0.8,1),nrow=2)

x=ones(S,1)%*%t(mu)+matrix(rnorm(S*2),nrow = S)*diag(sigma)
```

2. The inverse CDF method

Theory

A very general technique for transforming from standard uniform to the distribution with cdf F is simply to compute inverse of F(u). $F^{-1}(u) = \inf\{x \mid F(x) \geq u\}$

```r
# exponential case
e=-7*log(1-runif(S))
hist(e)
```



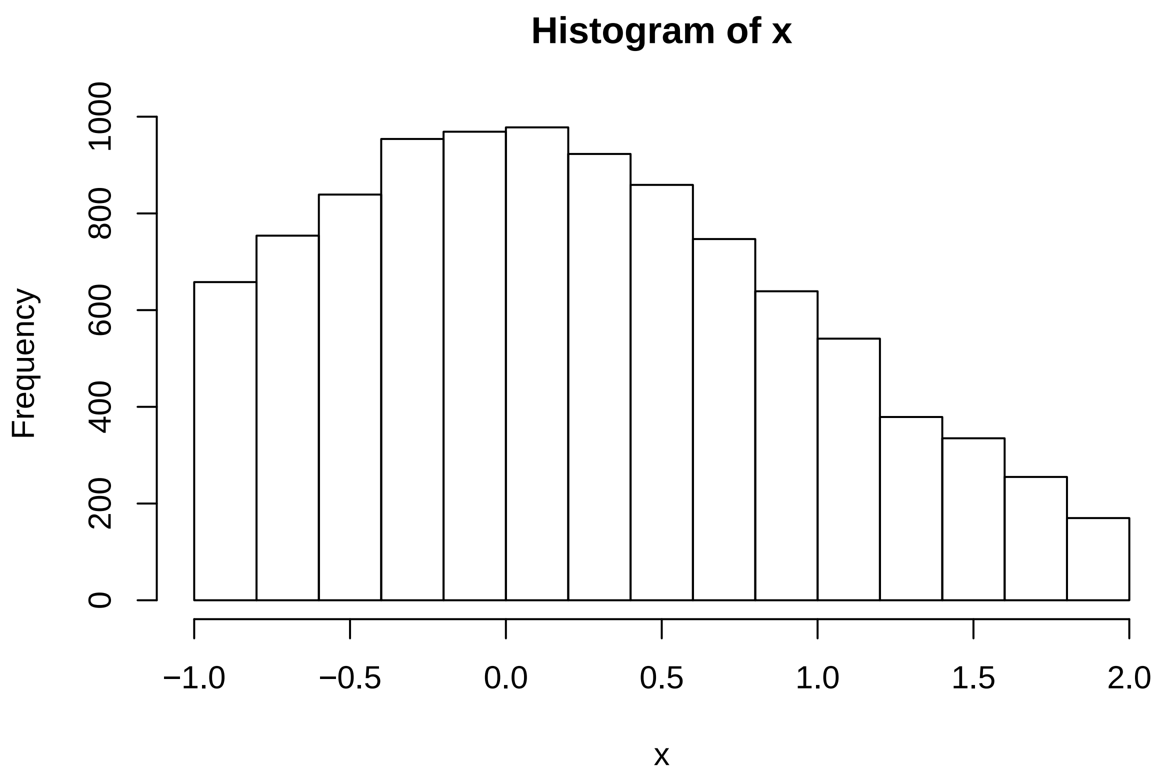
3. Acceptance/rejection rule

Theory

We wish to draw from a distribution F with density f but need to draw from an auxiliary distribution G with density g instead. The only requirement on G is that there exists a known and finite number c such that $f(x) \leq c \cdot g(x)$ for all $x \in \mathcal{D}$

Sample y from G and u from $U(0,1)$. If u $u \leq \frac{f(y)}{c \cdot g(y)}$, define x=y ("accept") and stop; otherwise, ignore whatever happened so far ("reject") and start all over

```
# Truncated normal, [-1,2], using acceptance/rejection rule and standard normal g
x=zeros(S,1)
count=0
for (i in 1:S){
  done=0
  while (done==0){
    y=rnorm(1)
    count=count+1
    if (y>-1&y<2){
      x[i]=y
      done=1}
  }
}
hist(x)
```




```r
count
```

```
## [1] 12225
```

4. Antithetic acceleration

Theory

Variance reduction: $Var(\frac{1}{2}\left(\hat{\theta}_1+\hat{\theta}_2\right))=\frac{1}{4}\sigma^2+\frac{1}{4}\sigma^2+2\cdot\frac{1}{4}\rho\sigma^2=\frac{\sigma^2}{2}(1+\rho)$

Antithetic acceleration: $\frac{1}{2}\left(\frac{1}{S}\sum_{s=1}^{S}v\left(x^{(s)}\right)+\frac{1}{S}\sum_{s=1}^{S}v\left(x^{*(s)}\right)\right)$

```r
howmany=100
u=matrix(runif(S*howmany),nrow = S,ncol=howmany)
e=-7*log(1-u)
theta=colMeans(e)
mean(theta)
```

```
## [1] 7.004498
```

```r
std(theta)
```

```
## [1] 0.06789412
```

```r
# Using twice as many draws decreases the standard error by a factor of sqrt(2)
u_more=matrix(runif(S*howmany),nrow=S,ncol=howmany)
e_more=-7*log(1-u_more)
dim(e_more)
```

```
## [1] 10000   100
```

```
e_double=matrix(c(e,e_more),nrow = 2*S,ncol=howmany)
theta_double=colMeans(e_double)
std(theta_double)
```

```
## [1] 0.04817811
```

```
#But now we don't need double size of draws
# antithetic acceleration perform better
e_star=-7*log(u)
e_anti=(e+e_star)/2
theta_anti=colMeans(e_anti)
std(theta_anti)
```

```
## [1] 0.02747244
```

5.Importance sampling

Theory

We are trying to estimate $\int v(x)f(x)dx/$. Because f(x) is a density function so $\int f(x)dx = 1$

Therefore, $\int v(x)f(x)dx = \frac{\int v(x)f(x)dx}{\int f(x)dx} = \frac{\int v(x)\frac{f(x)}{g(x)}g(x)dx}{\int \frac{f(x)}{g(x)}g(x)dx}$.

Define w=f/g. If X~G, then

$\int v(x)\frac{f(x)}{g(x)}g(x)dx = E[w(X)v(X)]$.

$\int \frac{f(x)}{g(x)}g(x)dx = E[w(X)]$ By some Russian's probability limit law, we then find $\frac{\frac{1}{S}\sum_{s=1}^{S} w\left(x^{(s)}\right)v\left(x^{(s)}\right)}{\frac{1}{s}\sum_{s=1}^{S} w\left(x^{(s)}\right)} \rightarrow \frac{E[w(X)v(X)]}{E[w(X)]}$

```
x=matrix(-3*log(1-runif(S*howmany)),nrow = S,ncol = howmany)
w=(exp(-x/7)/7)/(exp(-x/3)/3)
ta_is=sum(w*x)/sum(w)
mean(ta_is)
```

```
## [1] 6.962204
```

```
std(ta_is)
```

```
## [1] 0.9015568
```