1

The Box-Muller technique is is based on the observation that if (X, Y) is a normal random variable in $\mathbb{R}^2$ then its representation (R, Θ) in polar coordinates is such that R^2 is exponential with mean $1/2$ and Θ uniform on $(0, 2\pi)$.

Now, if U, V are i.i.d. uniform on $(0, 1)$ then $R^2 = -2 \log U$ is exponential with mean $1/2$, and $\Theta = 2\pi V$ is uniform on $(0, 2\pi)$.

We first n independent samples from U and n from V:

```
n=10000; u=runif(n); v=runif(n)
```

We then compute R^2 and Θ:

```
rsquared=-2*log(u); theta=2*pi*v
```

Finally, we generate samples for (X, Y) by using the formula $X = R \cos \Theta$, $Y = R \sin \Theta$:

```
r=sqrt(rsquared); x=r*cos(theta); y=r*sin(theta)
```

We can concatenate x and y to obtain a size $2n$ sample:

```
z=c(x,y)
```

We estimate the mean and the variance

```
mean(z); var(z)
```

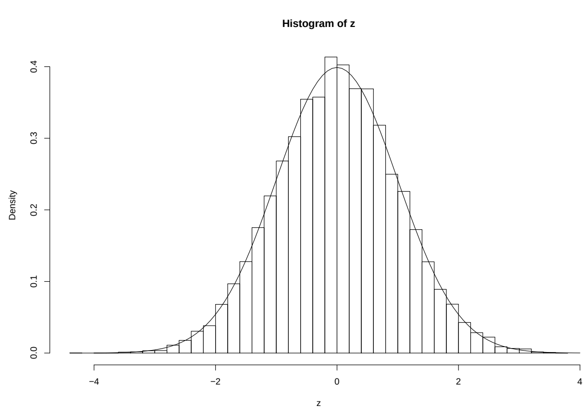
The answer is: 0.002601275 and 1.000343, respectively.

We generate a histogram for z and compare it against the theoretical density

$$f(z) = (2\pi)^{-1/2} \exp(-z^2/2)$$

as follows:

```
hist(z,breaks=50,probability=1)
f=function(x){(2*pi)^(-1/2)*exp(-x^2/2)}
plot(f,-4,4,add=TRUE)
dev.copy(postscript,'plot1.ps')
dev.off()
```



We obtain a good much and we're happy.

Define now

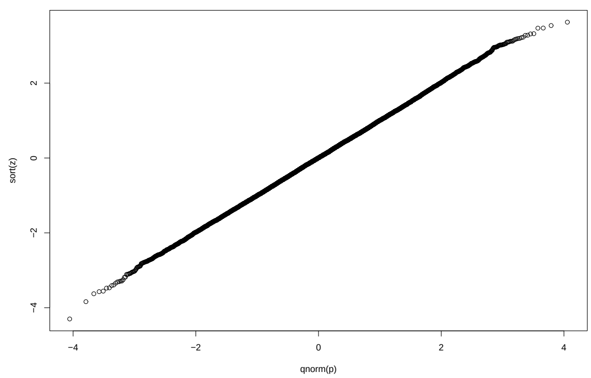
$$F(z) = \int_{-\infty}^{z} f(t)dt,$$

and its inverse function

$$F^{-1}(p) = \inf\{z \in \mathbb{R} : \ F(z) > p\} = \inf\{z \in \mathbb{R} : \ F(z) = p\}.$$

The function F^{-1} is called quantile function. The formula for $F^{-1}(p)$ in R is `rnorm(p)`. We have a sample z of size $2n$ and want to check if it comes from F. We generate $2n$ equally spaced points in the interval $(0, 1)$, store them in a vector p and plot $F^{-1}(p)$ against z:

```
p = ((1:(2*n))-0.5)/(2*n)
plot(qnorm(p),sort(z))
dev.copy(postscript,'plot2.ps')
dev.off()
```



We obtain a rather straight line and we're happy.

2 If Z is standard normal then $X = \sigma Z + \mu$ has the NORMAL(μ, σ^2) law. In our case,

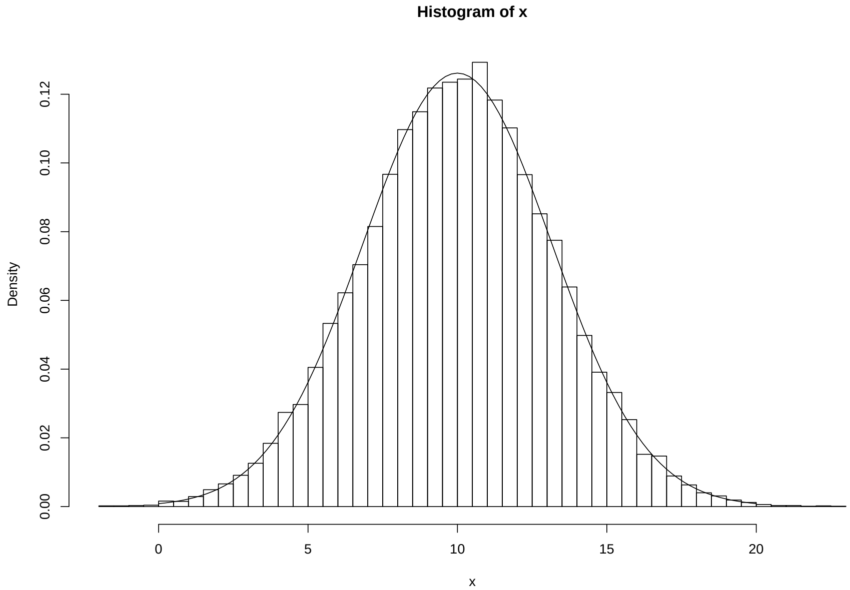
$$X = \sqrt{10}Z + 10.$$

Hence we do

```
n=10000; u=runif(n); v=runif(n)
rsquared=-2*log(u); theta=2*pi*v
r=sqrt(rsquared); x=r*cos(theta); y=r*sin(theta)
z=c(x,y)
x=sqrt(10)*z+10;
mean(x); var(x)
hist(x,breaks=50,probability=1)
f=function(x){(2*pi*10)^(-1/2)*exp(-(x-10)^2/(2*10))}
plot(f,0,20,add=TRUE)
p = ((1:(2*n))-0.5)/(2*n)
plot(qnorm(p),sort(x))
```

`mean(x)` gives 10.01072

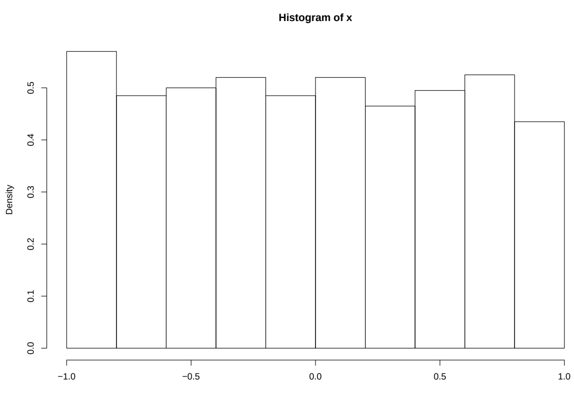
`var(x)` gives 10.03228



3 ______ If U is UNIFORM$(0,1)$ then $X = aU + b$ is UNIFORM$(b, a+b)$. Hence

$$X = 2U - 1 \sim \text{UNIFORM}(-1, 1).$$

```
n=1000
u=runif(n)
x=2*u-1
mean(x); var(x); median(x)
hist(x,probability=1)
```



`mean(x)` -0.01909700

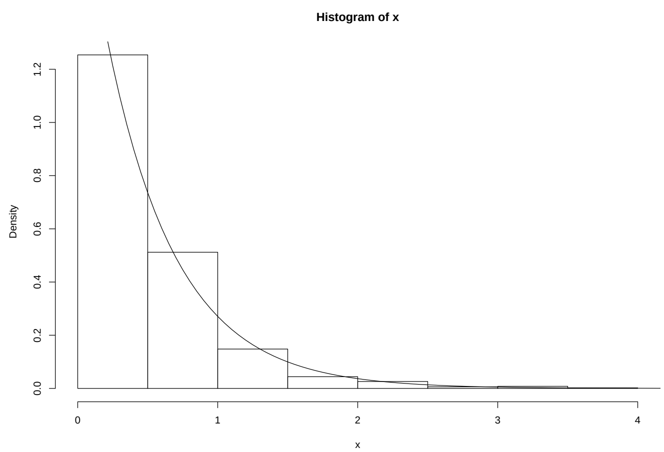
`median(x)` -0.02208797

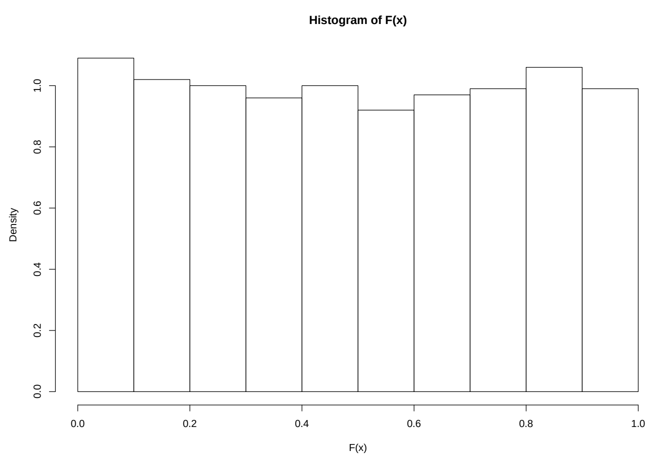
4 ______ If U is UNIFORM$(0,1)$ then $X = -\frac{1}{\lambda}\log U$ is EXPONENTIAL(λ). In our case,

$$X = -\frac{1}{2}\log U$$

does the job.

```
n=1000
u=runif(n)
lambda=2
x=-(1/lambda)*log(u)
hist(x,probability=1)
f=function(x){lambda*exp(-lambda*x)}
plot(f,0,5,add=TRUE)
F=function(x){exp(-lambda*x)}
plot(runif(n),F(x))
hist(F(x),probability=1)
```





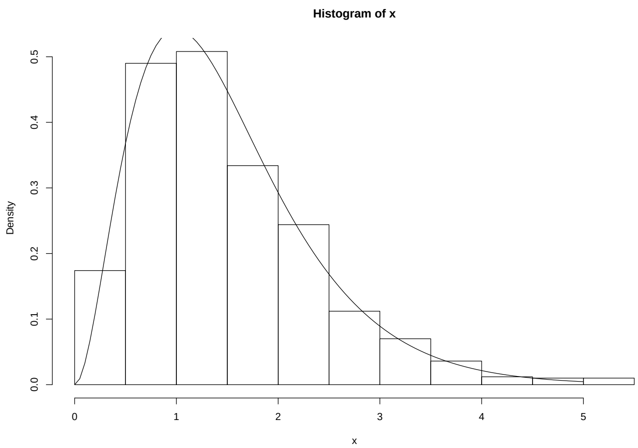
`mean(x)` 0.5050377

`var(x)` 0.2371565

5 A GAMMA(n, λ) random variable with n a positive integer has the law of the sum of n i.i.d. EXPONENTIAL(λ) random variables. So, if Z_1, Z_2, Z_3 are i.i.d EXPONENTIAL(2) then

$$X = Z_1 + Z_2 + Z_3 \sim \text{GAMMA(3,2)}.$$

```
n=1000
u1=runif(n); u2=runif(n); u3=runif(n)
z1=-(log(u1)/2); z2=-(log(u2)/2); z3=-(log(u3)/2)
x=z1+z2+z3
hist(x,probability=1)
f=function(x){dgamma(x,3,2)}
plot(f,0,5,add=TRUE)
mean(x)
var(x)
```



`mean(x)` 1.504518

`var(x)` 0.801717

6

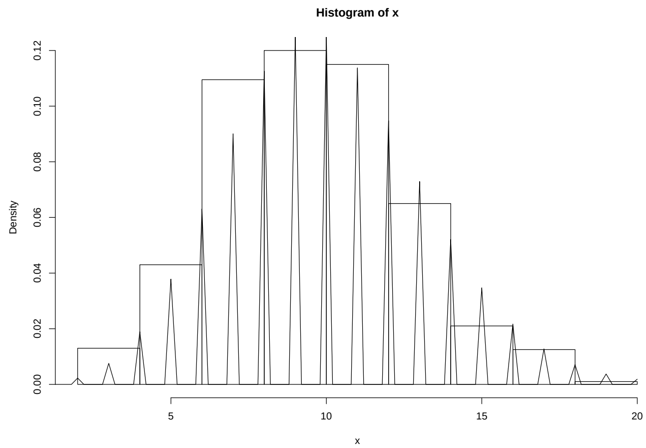
Let $Z_1, Z_2, \ldots$ be a sequence of i.i.d. EXPONENTIAL(λ) random variables. Let $T > 0$. Then

$$X(T) = \sum_{i=1}^{\infty} \mathbf{1}(Z_1 + \cdots + Z_i \leq T)$$

is POISSON(λT). So, to generate 1000 samples from a POISSON(10) distribution, we pick λ and T such that $\lambda T = 10$, for example $\lambda = 10$ and $T = 1$, and do this:

```
n=1000
x=c(1:n)
lambda = 10
T = 1
b = lambda*T
for(i in 1:n)
{
k=0
sum=0
while(sum < b)
{sum=sum-log(runif(1)); k=k+1}
x[i]=k-1
}
```

```
hist(x,probability=1)
f=function(x){dpois(x,b)}
plot(f,0,2*b,add=TRUE)
mean(x)
var(x)
```



`mean(x)` 10.029

`var(x)` 9.11127

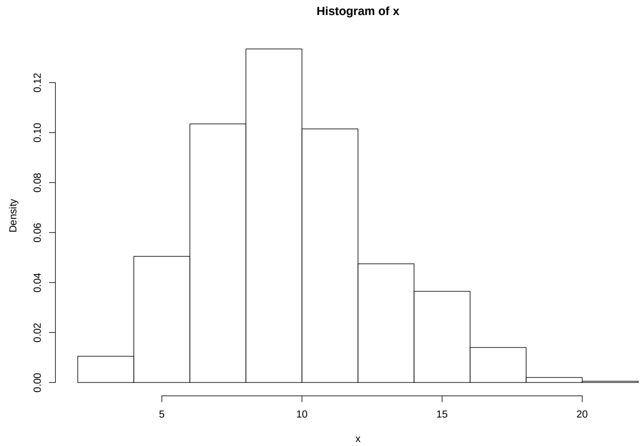
—

Another method is by inverting the distribution function $F(x) = P(X \leq x)$ of a Poisson(λ) random variable and setting

$$X = F^{-1}(U) = \inf\{x : F(x) > U\}.$$

We do this as follows:

```
n=1000
x=c(1:n)
lambda = 10
for(i in 1:n)
{
k=0
u=runif(1)
while(ppois(k,lambda) < u){k=k+1}
x[i]=k
}
```



`mean(x)` 10.038

`var(x)` 9.892448

7

Toss a fair coin 12 times and count the number of heads; this is BINOMIAL$(12, 1/2)$.

```
N=12
u=runif(N)
cond = (u < 1/2)
x = length(u[cond]); x
```

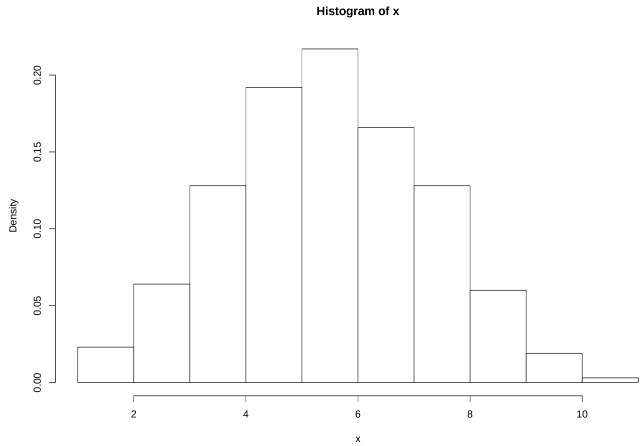
The command `cond = (u < 1/2)` generates a vector whose entries are TRUE or FALSE respectively if $(u < 1/2)$ is satisfied or not. The command `u[cond]` deletes all the entries of u which do not satisfy the condition; and the command `length(...)` obtains the length of the vector.

And here is how we generate 1000 BINOMIAL$(12, 1/2)$ samples, enabling us to check that all is correct.

```
N=12
n=1000
x=c(1:n)
for(i in 1:n)
{
u=runif(N)
cond = (u < 1/2)
x[i] = length(u[cond])
}
mean(x)
var(x)
hist(x)
```

`mean(x)` 5.958

`var(x)` 3.273510



8

To throw one die once we do:

```
p=c(1/6,1/6,1/6,1/6,1/6,1/6)
u=runif(1); k=1; s=p[1]; while(s<u){k=k+1;s=s+p[k]}; k
```

To throw $N = 10$ dice we do:

```
N=10
p=c(1/6,1/6,1/6,1/6,1/6,1/6)
x=c(1:N)
for(i in 1:N)
{u=runif(1); k=1; s=p[1]; while(s < u){k=k+1; s=s+p[k]}; x[i]=k}
sum = sum(x); sum
```

The last command computes the sum of the values of the 10 dice.

To compute the probability that this sum exceeds $a = 30$, we perform the above experiment a large number of times and estimate the fraction of times that the sum exceeds a.

```
n=1000
N=10
p=c(1/6,1/6,1/6,1/6,1/6,1/6)
x=matrix(0, n, N, byrow=T)
for(i in 1:n)
{
for(j in 1:N)
{u=runif(1); k=1; s=p[1]; while(s<u){k=k+1;s=s+p[k]}; x[i,j]=k}
}
s=c(1:n)
for(i in 1:n) {s[i]=sum(x[i,])}
probthatsumexceeds30 = length(s[s>30])/n; probthatsumexceeds30
```

The answer is

```
probthatsumexceeds30 0.801
```

Using normal approximation (which is not that great here), we have that, if X_i, $i = 1, \ldots, 10$ are i.i.d. uniformly distributed in $\{1, 2, 3, 4, 5, 6\}$ and if $S = \sum_{i=1}^{6} X_i$ then

$$P(S > 30) = P((S - ES)/\sqrt{\text{var}(S)} > (30 - 35)/\sqrt{29.2}) = 0.82.$$

9 Let ξ, η be two i.i.d. standard normals. We seek constants a, b, c so that

$$\begin{aligned} x &= a\xi + c\eta \\ y &= c\xi + b\eta \end{aligned}$$

are jointly normal with $var(x) = 1$, $\text{var}(y) = 9$, $\text{cov}(x, y) = 2$. We have

$$\begin{aligned} \text{var}(x) &= a^2 + c^2 = 1 \\ \text{var}(y) &= b^2 + c^2 = 9 \\ \text{cov}(x, y) &= ac + bc = 2 \end{aligned}$$

The last one gives

$$b + a = \frac{2}{c}.$$

Subtracting the second from the first,

$$b^2 - a^2 = 8 = (b - a)(b + a)$$

and so

$$b - a = 4c.$$

So

$$\begin{aligned} a &= \frac{1}{c} - 2c \\ b &= \frac{1}{c} + 2c. \end{aligned}$$

And so

$$1 = b^2 + c^2 = (c^{-1} + 2c)^2 + c^2.$$

Multiplying both sides by c^2 we obtain

$$c^2 = (1 + 2c^2)^2 + c^4,$$

which is a quadratic in c^2:

$$5c^4 - 5c^2 + 1 = 0.$$

Hence

$$c = \sqrt{\frac{5 \pm \sqrt{5}}{10}} = 0.526, \ 0.851.$$

Both values are acceptable. Choose, for example, the first one. We find

$$c = 0.526, \ a = \frac{1}{c} - 2c = 0.851, \ b = \frac{1}{c} + 2c = 2.954.$$

To generate X, Y as neededm we thus do

$$\begin{aligned} X &= 0.851\xi + 0.526\eta + 10 \\ Y &= 0.526\xi + 2.954\eta + 12. \end{aligned}$$

```
n=10000; u=runif(n); v=runif(n)
rsquared=-2*log(u); theta=2*pi*v
r=sqrt(rsquared); xi=r*cos(theta); eta=r*sin(theta)
X=0.851*xi+0.526*eta+10
Y=0.526*xi+2.954*eta+12
mean(X)
mean(Y)
var(X)
var(Y)
cov(X,Y)
```

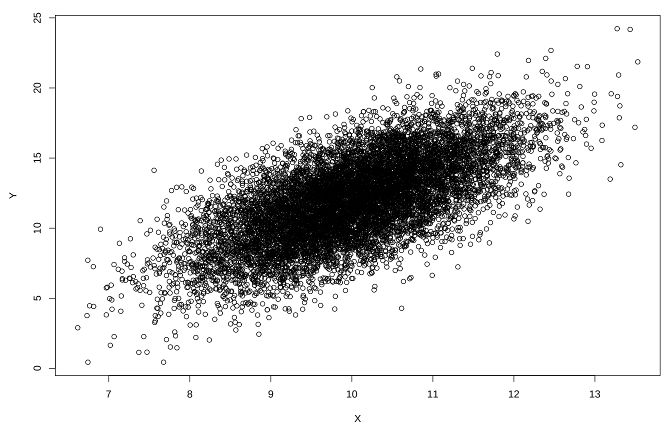
`mean(X)` 10.00332

`mean(Y)` 12.00170

`var(X)` 0.99564

`var(Y)` 9.054209

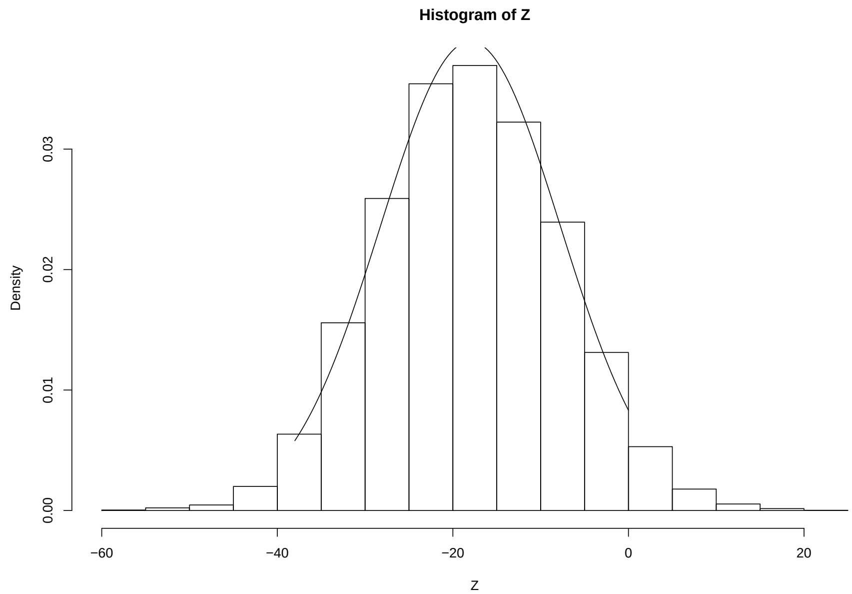
`cov(X,Y)` 2.003293



Now observe that $Z = 3X - 4Y$ has mean $30 - 48 = -18$ and variance $9\,\mathrm{var}(X) + 16\,\mathrm{var}(Y) - 24\,\mathrm{cov}(X,Y) = 105$. So we expect Z to be NORMAL$(-18, 105)$.

```
Z=3*X-4*Y
mean(Z)
var(Z)
hist(Z,probability=1)
f=function(x){dnorm(x,-18,sqrt(105))}
plot(f,-38,0,add=TRUE)
```

The plot is right on the money.



10

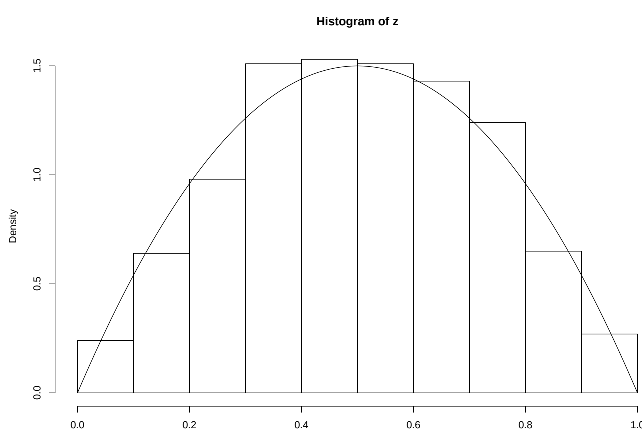
FIRST METHOD

If X, Y are independent GAMMA(a,λ), GAMMA(b,λ), respectively, then

$$Z = \frac{X}{X+Y} \sim \text{ BETA}(a,b).$$

Therefore:

```
n=1000
u1=runif(n); u2=runif(n); u3=runif(n); u4=runif(n)
x=-log(u1)-log(u2); y=-log(u3)-log(u4)
z=x/(x+y)
mean(z)
hist(z,probability=1)
f=function(x){dbeta(x,2,2)}
plot(f,0,1,add=TRUE)
```



SECOND METHOD:

The density of BETA$(2,2)$ is proportional to:

$$f(x) = x(1-x).$$

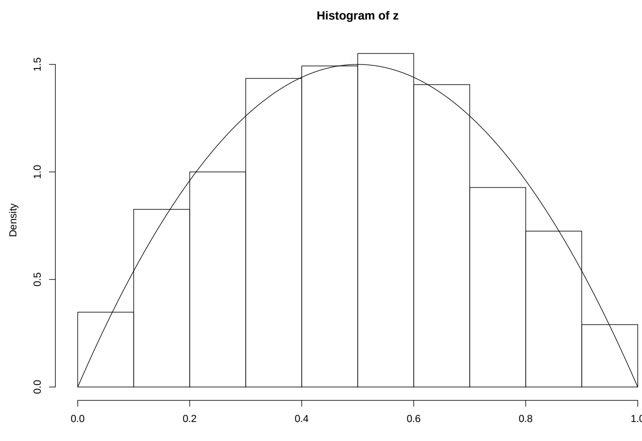
The maximum of $f(x)$ over $0 < x < 1$ occurs at $x = 1/2$ and equals $M = 1/4$. Let $g(x) = 1$, the density of a standard uniform random variable. We use

$$p(x) = \frac{f(x)}{Mg(x)} = 4x(1-x)$$

as the acceptance probability.

```
n=1000
x=runif(n)
u=runif(n)
z = x[u < 4*x*(1-x)]
mean(z)
hist(z,probability=1)
f=function(x){dbeta(x,2,2)}
plot(f,0,1,add=TRUE)
```

This works, but it is slower (due to time wasted for rejections).



THIRD METHOD:

We can apply the inversion method because R knows the inverse function of

$$F(x) = 3x^2 - 2x^3 \text{ (this is the distribution function of BETA}(2,2)$$

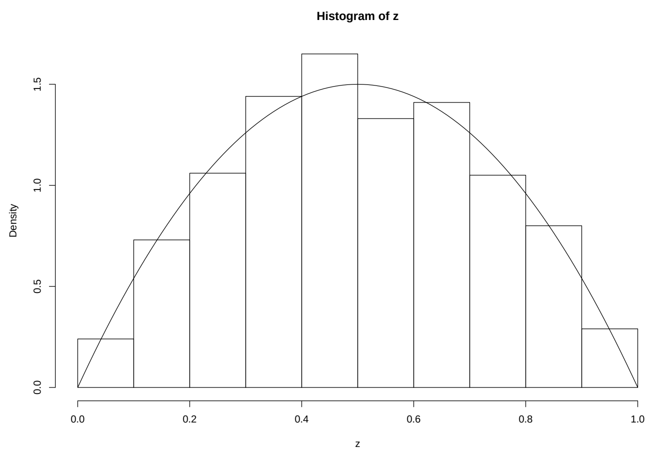
and calls it using the command `qbeta(u,2,2)`. In other words, the unique x for which

$$F(x) = u$$

is given by `qbeta(u,2,2)` in R. The inverse function is also called quantile function.

```
n=1000
u=runif(n)
```

```
z = qbeta(u,2,2)
mean(z)
hist(z,probability=1)
f=function(x){dbeta(x,2,2)}
plot(f,0,1,add=TRUE)
```



11. The goal is to draw a sample from some given distribution π on a set S by creating a MC whose limiting distribution is π. The MH algorithm does the following:

First define "arbitrary" transition probabilities $q(x, y)$ on S.

Second, define acceptance probability $\alpha(x, y) \in (0, 1]$ via the formula

$$\alpha(x, y) = h(\pi(y)q(y, y)/\pi(x)q(x, y)).$$

If the current state is x, then generate a y from the distribution $q(x, \cdot)$.

Toss a coin which has probability of heads $\alpha(x, y)$.

If heads come up, set the new state equal to y.

If tails come up, set the new state equal to the old one, i.e., x.

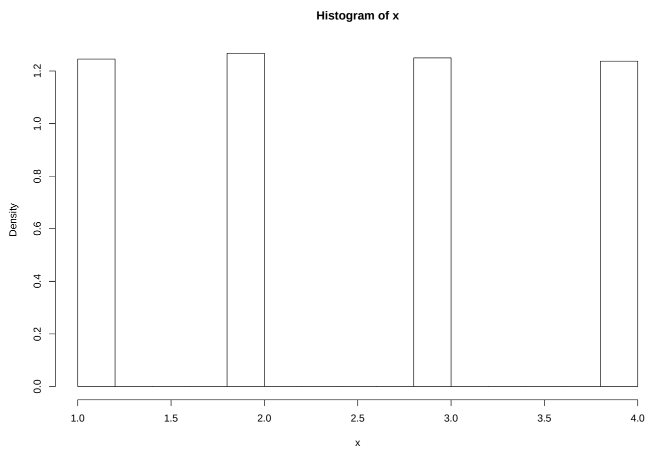
For the specific exercise, we have a uniform distribution on a set S with four elements, so a reasonable way to move is by moving to the neighbours, viz.,

$$Q = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1 & 0 \end{pmatrix}.$$

We do this in R as follows:

```
p=c(0.25,0.25,0.25,0.25);
q=matrix(c(0,1,0,0,1/2,0,1/2,0,0,1/2,0,1/2,0,0,1,0),nrow=4,ncol=4,byrow=T);
N=10000
x=c(1:N)
x[1]=1
for(i in (1:N))
```

```
{
u=runif(1); y=1; s=q[x[i],1]; while(s<u){y=y+1; s=s+q[x[i],y]}
alpha=min(1,q[y,x[i]]/q[x[i],y])
V=runif(1)
if(V<alpha){x[i+1]=y}else{x[i+1]=x[i]}
}
hist(xprobability=1)
```



12 The Gamma(2,1) law has density $\pi(x) = xe^{-x}$, supported on $x > 0$. The method (see Explanatory Note 4) works as follows: We are asked to use steps distributed as NORMAL(0,ε^2). Let f be the density of NORMAL(0,ε^2). (We can play with the standard deviation ε).

When the chain is currently at x we generate z from f, and attempt to move to $y = x + z$. The density of moving from x to y is thus

$$q(x, y) = f(y - x).$$

If $y < 0$, we reject it. Otherwise, we define acceptance probability

$$\alpha(x, y) = \min(1, \pi(y)q(y, x)/\pi(x)q(x, y)) = \min(1, \pi(y)/\pi(x)).$$

We toss a coin with probability of heads equal to $\alpha(x, y)$. If we get heads then we move to y. Otherwise, we stay at x and repeat the procedure. The code in R is:

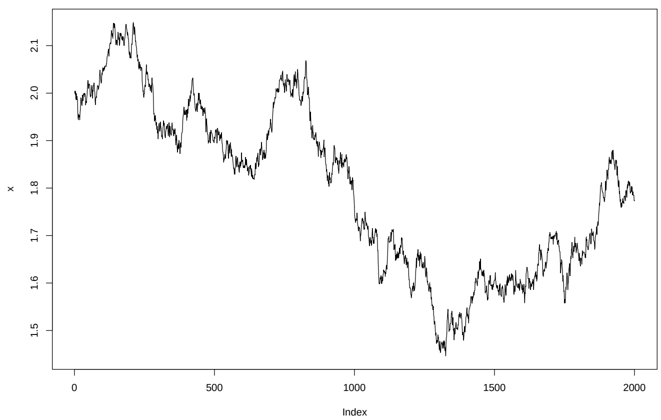
```
pi=function(x){x*exp(-x)}
N=2000
x=c(1:N)
epsilon=1.4
x[1]=2
for(i in 1:N){
z=rnorm(1,0,epsilon)
y=x[i]+z
if(y<0){x[i+1]=x[i]}
else
{
```

```
alpha=min(1,pi(y)/pi(x))
V=runif(1,0,1)
if(V<alpha){x[i+1]=y}else{x[i+1]=x[i]}
}
}
mean(x[N/4:N])
var(x[N/4:N])
hist(x,probability=1)
plot(pi,0,6,add=TRUE)
```

There are two things to play with: ε and the initial state.

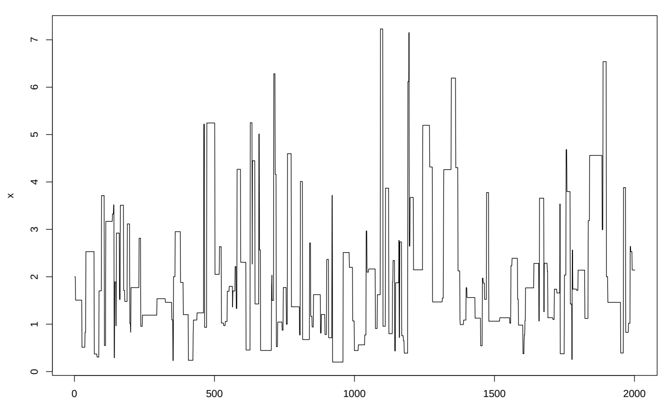
If we choose ε too small then we will need a long time to converge.

• With $\varepsilon = 0.01$ we have the following plot for $x[i]$ as a function of $i = 1, \ldots, N$:



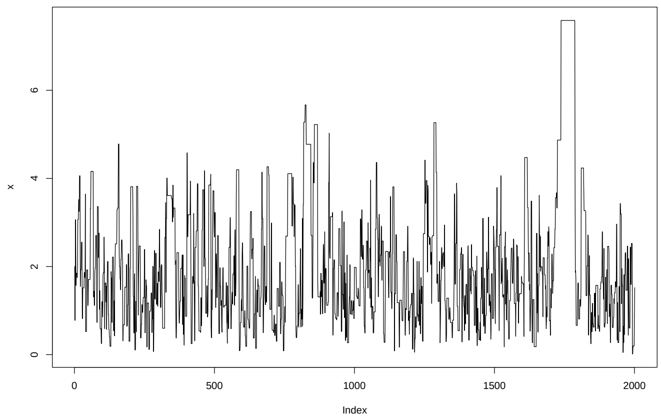
If we choose ε too large then, again, we will waste a lot of time getting very large values of z which will wither be rejected (either either due to $x + z < 0$ or because $\alpha(x, y) = \pi(y)/\pi(x)$ is far too small at a large y). Hence, again, convergence will be slow.

• With $\varepsilon = 10$ we have the following plot for $x[i]$ as a function of $i = 1, \ldots, N$:



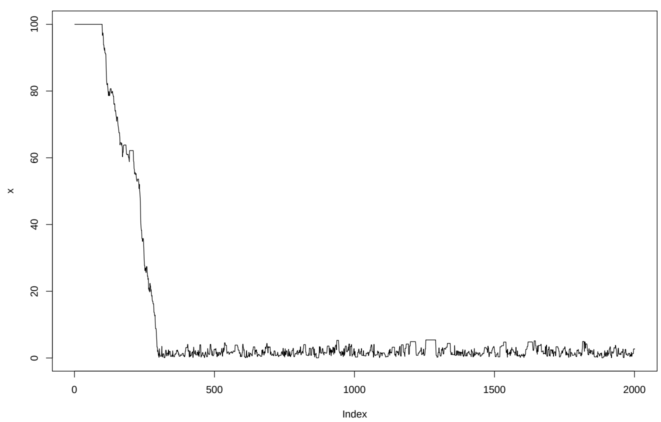
Since we know that π has variance 2 it is reasonable to choose ε^2 comparable to it.

With $\varepsilon = 1.4$ we have the following plot for $x[i]$ as a function of $i = 1, \ldots, N$:



As for the initial state, we should not start from a very large value because we will waste time going down. Since the mean of π is 2, it is best to start from an initial value close to 2.

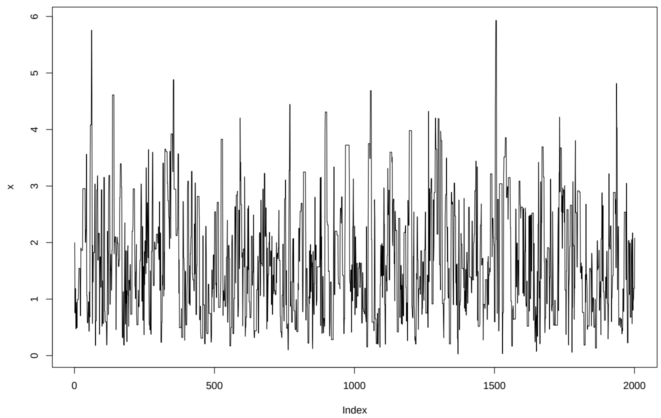
With initial state equal to 100 and $\varepsilon = 1.4$ we have



—

So we decide to pick initial state equal to 2 and $\varepsilon = 1/4$.

This gives the following plot for $x[i]$:



and the following histogram: