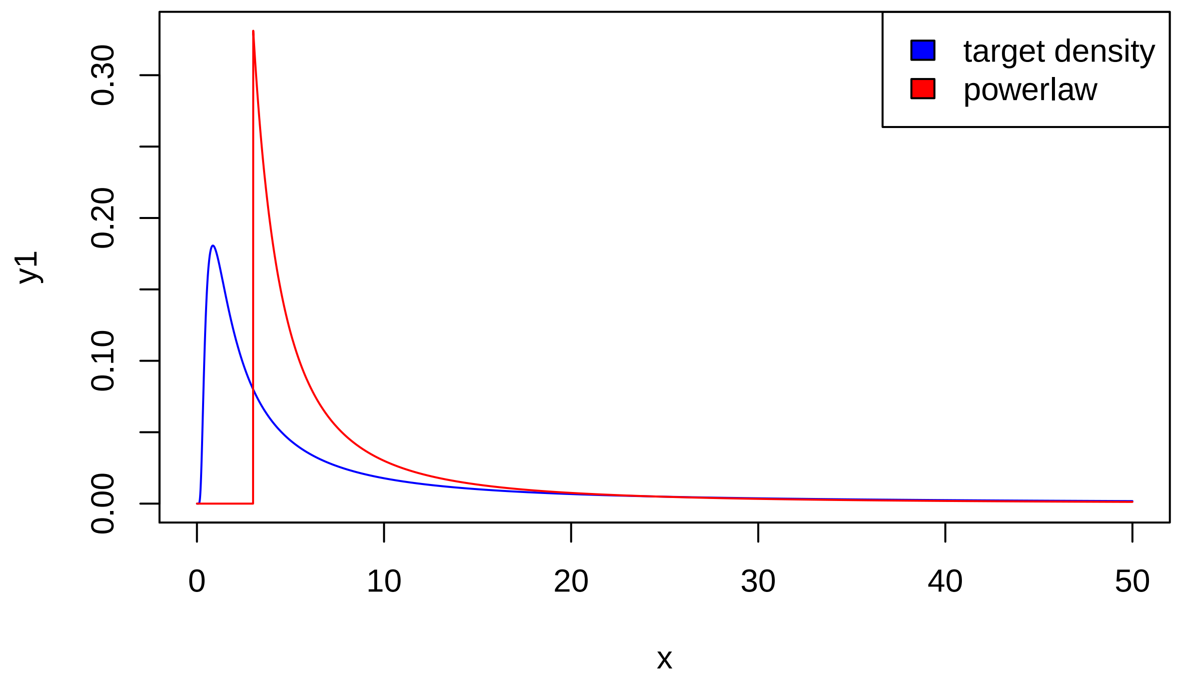
Computer Lab 3

Martynas Lukosevicius, Alejo Perez Gomez, Zahra Jalil Pour

17/11/2020

Question 1



Power-law distribution cannot be used just by itself because it does not support range from 0 to T_{min}. Because of this, we need to use mixture distribution. To support x from 0 to T_{min} we choose uniform distribution $Unif(0, T_{min})$. As we can see, powerlaw distribution is monotonically decreasing, so T_{min} should be equal to x where target density has maximum value.

Let's find maximum of target density function:

$$\frac{\partial}{\partial x}\frac{ce^{-\frac{c^2}{2x}}x^{-\frac{3}{2}}}{\sqrt{2\pi}} = \frac{ce^{-\frac{c^2}{2x}}\left(c^2-3x\right)}{2\sqrt{2\pi}x^{7/2}}$$

$$\frac{c\frac{e^{-c^2}}{2x}\left(c^2-3x\right)}{2\sqrt{2\pi}x^{7/2}} = 0$$

$$x = \frac{c^2}{3}$$

$$T_{min} = \frac{c^2}{3}$$

To make a mixture model we need to know the probability of taking uniform distribution and powerlaw distribution probability. That number will be in the range $[0, T_{min}]$:

$$\int_0^{T_{min}} cx^{-\frac{3}{2}} e^{-\frac{c^2}{2x}} \sqrt{2\pi}^{-1} dx = \frac{\Gamma\left(\frac{1}{2}, \frac{c^2}{2T_{min}}\right)}{\sqrt{\pi}}$$

as $T_{min} = \frac{c^2}{3}$ - $\frac{\Gamma\left(\frac{1}{2}, \frac{c^2}{2T_{min}}\right)}{\sqrt{\pi}} = \frac{\Gamma\left(\frac{1}{2}, \frac{3}{2}\right)}{\sqrt{\pi}} \sim 0.08326451666$

As a result, majorising density function is:

$$g(x) = \frac{2 * 0.08326}{c^2} 1_{[0, T_{min}]} + (1 - 0.08326) * \frac{2^{1-a}(a-1)\left(\frac{x}{c^2}\right)^{-a}}{c^2} * 1_{(T_{min}, \infty)}$$

2.

Target density:

$$f(x) = c(\sqrt{2\pi})^{-1} e^{\frac{-c^2}{2x}} x^{\frac{-3}{2}} 1_{(0,\infty)}(x)$$

We need to find c_{maj}

$$c_{maj} > 0; sup_x(f(x)/g(x)) \leq c_{maj}$$

$$h(x) = \frac{f(x)}{g(x)}$$

$c_{maj} = h(x_{maj})$

if x < T_{min}

$$\frac{\partial}{\partial x} \frac{f(x)}{\frac{0.16652}{c^2}} = \frac{e^{-\frac{c^2}{2x}}\left(1.19788c^5 - 3.59364c^3 x\right)}{x^{7/2}}$$

$$\frac{e^{-\frac{c^2}{2x}}\left(1.19788c^5 - 3.59364c^3 x\right)}{x^{7/2}} = 0$$

$$x_{maj} = \frac{c^2}{3}$$

x for cmaj = $0.333333c^2 = T_{min}$

if x> Tmin

$$\frac{\partial}{\partial x}\frac{f(x)}{g(x)} = \frac{3^{a-1}c^3\frac{e^{-c^2}}{2x}\left(\frac{x}{c^2}\right)^a\left((2a-3)x+c^2\right)}{2\sqrt{2\pi}(a-1)x^{7/2}}$$

$$\frac{3^{a-1}c^3\frac{e^{-c^2}}{2x}\left(\frac{x}{c^2}\right)^a\left((2a-3)x+c^2\right)}{2\sqrt{2\pi}(a-1)x^{7/2}} = 0$$

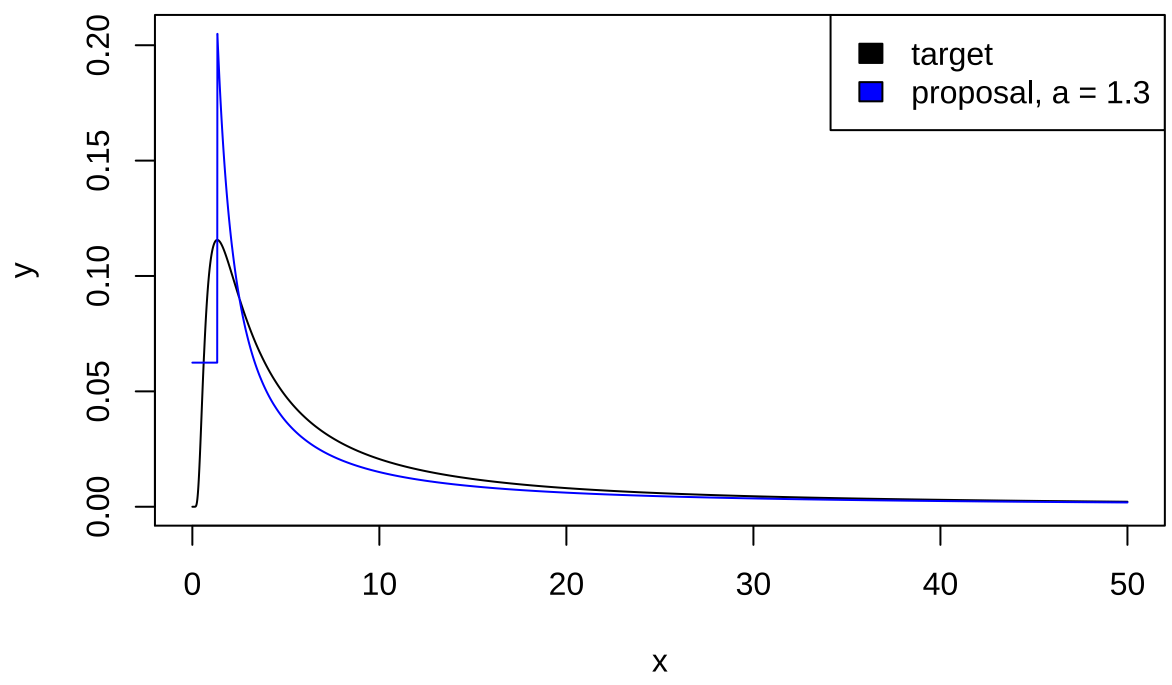
$$x_{maj} = \frac{c^2}{3-2a}$$

when $1 < a < 1.5$

$$c_{maj} = h(x_{maj})$$

As we can see a should be between 1 and 1.5 If $a >= 1.5$ majorising constant will be negative. We will fix $\alpha = 1.3$. Afterwards, the majorizing constant will ensure it gets above of the target.

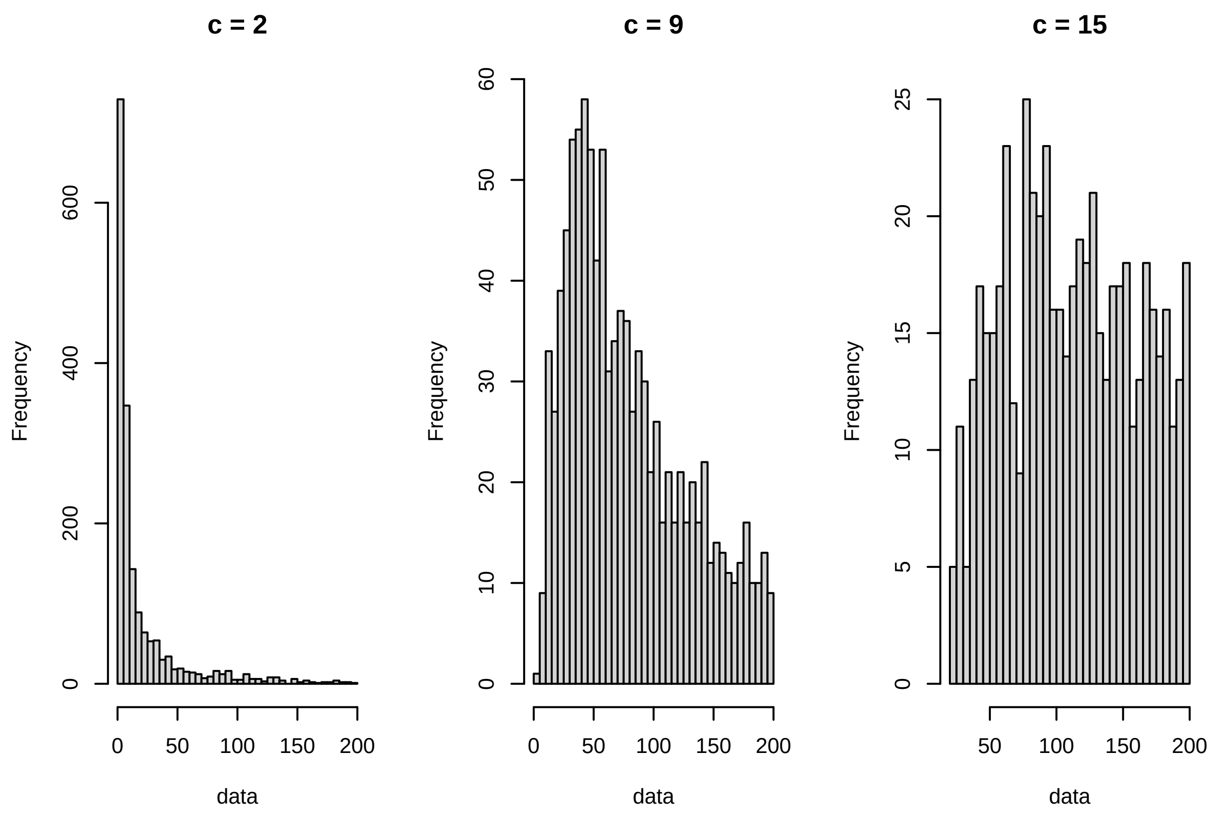
```r
majDensity <- function(x, c, a){
  Tmin <- (c^2)/3

  if(x>Tmin){
    return((1-0.083264) * powerlaw(x, a, Tmin))
  }
  else {return(0.083264 * dunif(x,0,Tmin))}
}
c <- 2
y3 <- sapply(x, majDensity, c = c, a = 1.3)
y1 <- sapply(x, distrib1, c = c)
plot(x,y1, type = "l", ylim = c(0,max(y3,y1)), ylab = "y")
lines(x,y3, col ="blue")
legend("topright", c("target", "proposal, a = 1.3"), fill=c("black", "blue"))
```



2

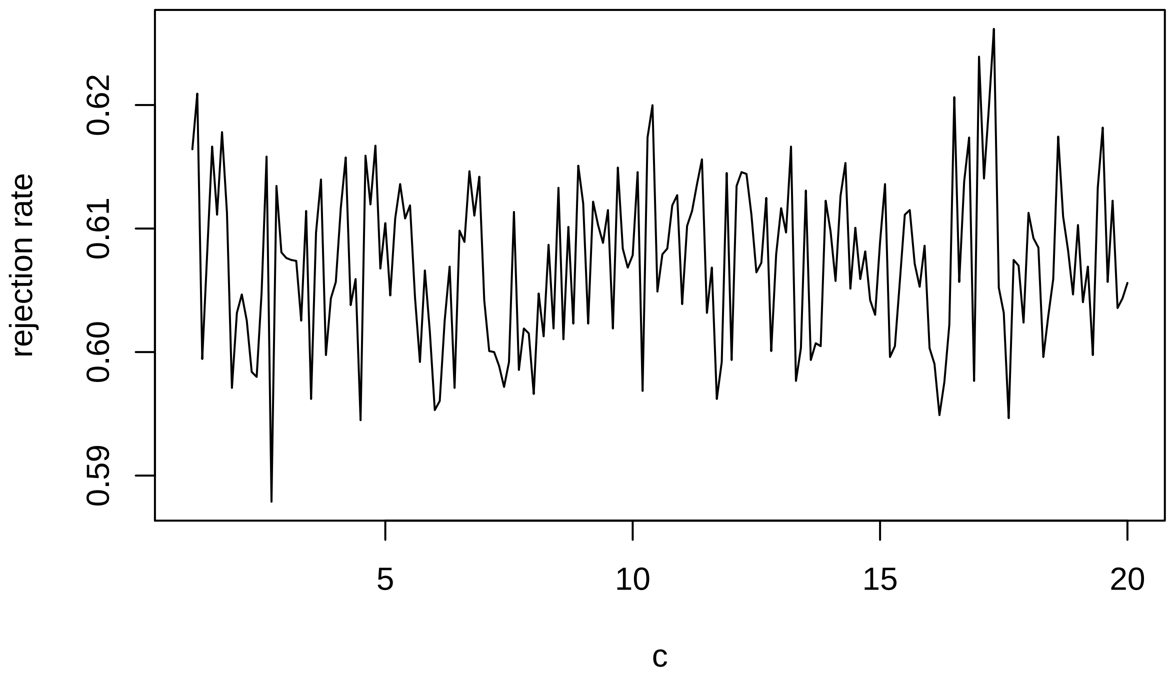
```r
library(poweRlaw)
randomnumber <- function(t,a){
  numb <- runif(1)
  if(numb<= 0.08326){
    return(runif(1,0,t))
  }
  else{
    return(rplcon(1,t,a))
  }
}

CompleteDist <- function(c, a, rej){
  z <- TRUE
  res <- 0

  Tmin <- (c^2)/3
  xmaj <- (c^2)/(3-2*a)
  cmaj <- distrib1(xmaj,c)/majDensity(xmaj,c, a)

  while (z == TRUE) {
    y <- randomnumber(Tmin,a)
    u <- runif(1)
    if(u <= distrib1(y, c) / (cmaj*majDensity(y,c,a))){
      res <- y
      z <- FALSE
    }
    if(rej){
    rejected <<- rejected + 1
    }
  }
  return(res)
}

rDist <- function(n,c,a ,rej = FALSE){
 return(replicate(n, CompleteDist(c, a, rej)))
}
```

3.



	c=2	c=9	c=15
mean	2.152240e+04	2.470089e+05	5.133534e+05
variance	2.913419e+11	3.948446e+13	1.302555e+14

Variance and mean is increase due to c



From plot we can see that rejection rate does not depend on the constant c of the target density.

Question 2

1.

$$DE(\mu, \alpha) = \frac{\alpha}{2} e^{-\alpha|x-\mu|}$$

- μ - location parameter
- $b > 0$ - scale parameter

inverse CDF of DE:

Source - https://en.wikipedia.org/wiki/Laplace_distribution

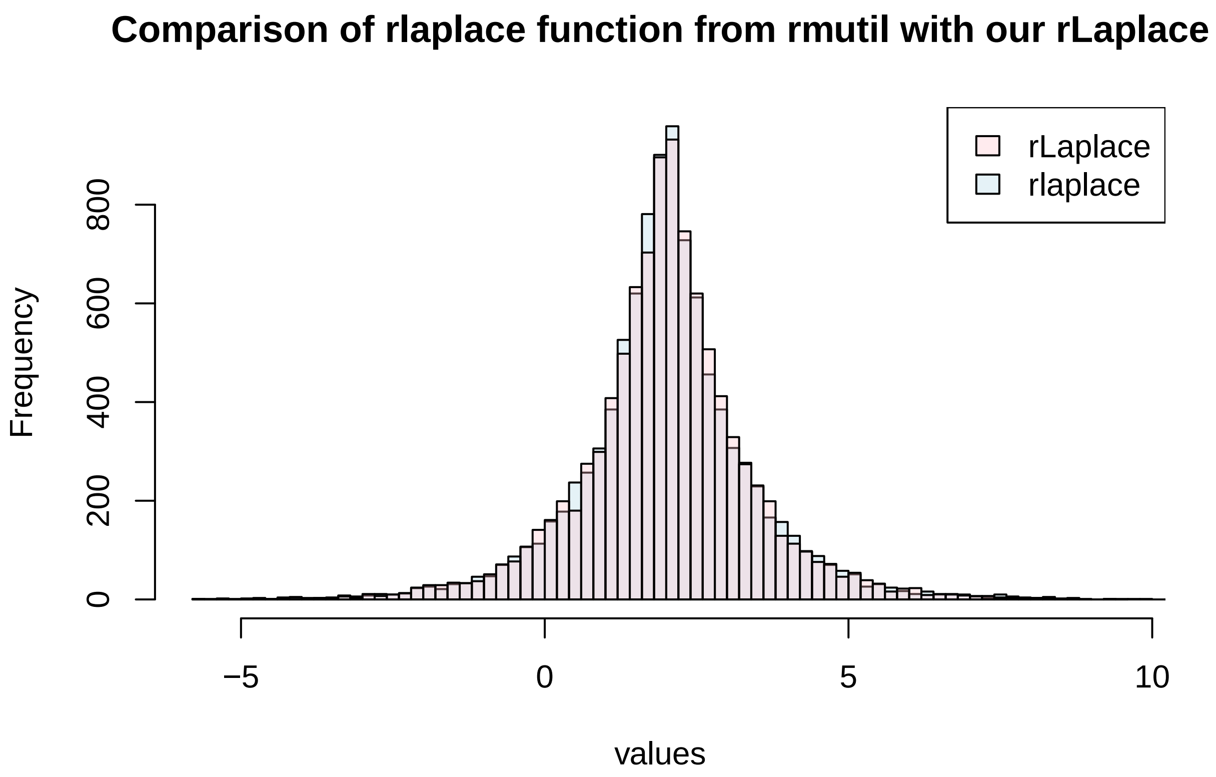
$$F^{-1}(p) = \mu - bsgn(p - 0.5)ln(1 - 2|p - 0.5|)$$

where $b = \frac{1}{\alpha}$

```r
rLaplace <- function(n, mean = 0, alpha = 1){
  b <- 1/alpha
  u <- runif(n)
  res <- mean - (b*sign(u-0.5) * log(1-(2*abs(u-0.5))))
  return(res)
}
```

meaning:

1. calculate b.
2. take n random variables from uniform distribution [0,1]
3. calculate random numbers from inverse CDF of laplace distribution where x is a random variable from uniform distribution



From histogram we can see that inverse CDF method approximated Laplace distribution reasonably, mean and variance similarly, in turn.

2.

```r
DE <- function(x, mean = 0,alpha = 1){
  return((0.5*alpha)*exp((-alpha)*abs(x-mean)))
}

genNorm <- function(c, rej){
  z <- TRUE
  res <- 0
  while (z == TRUE) {
    y <- rLaplace(1)
    u <- runif(1)
    if(u <= pnorm(y) / (c*DE(y))){
      res <- y
      z <- FALSE
    }
    if(rej){
    rejected <<- rejected + 1
    }
  }
  return(res)
}

rNorm <- function(n,c,rej = FALSE){
 return(replicate(n, genNorm(c, rej)))
}
```

algorithm:

1. write Laplace probability function
2. assign 0 to result value res
3. generate random number y from rLaplace function
4. generate random number u from uniform distribution
5. check if u is less or equal to probability of y in normal distribution / c * probability of y in laplace distribution
    a) if yes, return y
    b) repeat steps from 3

$$c > 0; sup_x(f(x)/g(x)) \le c$$

$$h(x) = \frac{f(x)}{g(x)}$$

$$f(x) = N(0,1) = \frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$$

$$g(x) = \frac{1}{2}e^{-|x|}$$

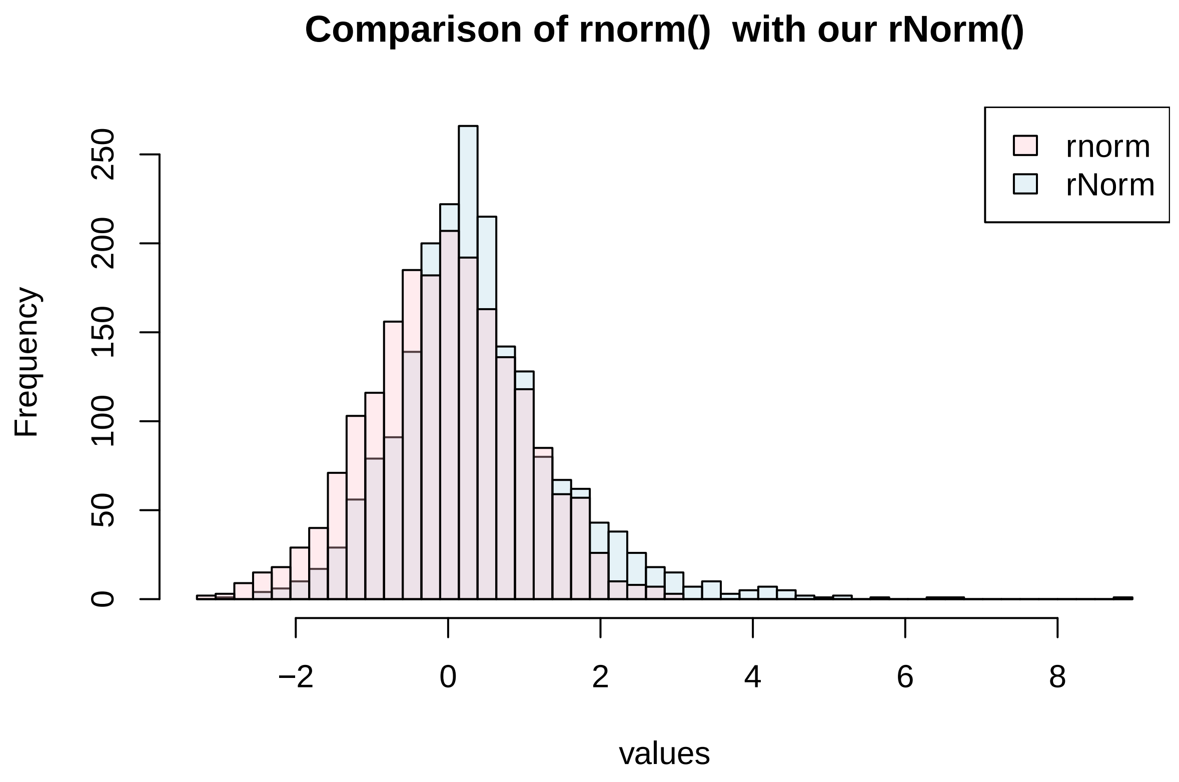
$$h(x) = \sqrt{\frac{2}{\pi}}e^{|x|-\frac{x^2}{2}}$$

$$\frac{d}{dx}\sqrt{\frac{2}{\pi}}e^{|x|-\frac{x^2}{2}} = \sqrt{\frac{2}{\pi}}xe^{|x|-\frac{x^2}{2}}\left(\frac{1}{|x|}-1\right)$$

$$\frac{\sqrt{2}e^{x-\frac{x^2}{2}}(x-1)}{\pi} = 0$$

$$x = \pm 1$$

$$c = h(1) = 1.3154892$$



	mean	variance
rNorm()	0.4032887	1.248962
rnorm()	-0.0219314	1.001522

rejection rate: 0.2019154, expected rejection rate $= 1 - \frac{1}{c} = 0.2398265$, difference - -0.0379111

We can see that mean and variance of generated distribution slightly differs.

Appendix

```
knitr::opts_chunk$set(echo = TRUE)


distrib1 <- function(x,c){
  if(x>0){
    return(c*(sqrt(2*pi)^(-1))*(exp(-(c^2)/(2*x))) * (x^(-3/2)))
  }
  else return(0)
}

powerlaw <- function(x,a,t){
  if(x>t){
    return(((a-1)/t) * ((x/t)^(-a)))
  }
  else return(0)

}

x <- seq(0,50, by=0.01)
c <- 1.6
y1 <- sapply(x, distrib1, c=c)
y2 <- sapply(x, powerlaw, a = 2,t = 3)
ymax <- max(y1,y2)
b <- min(c(y1,y2))
e <- max(c(y1,y2))
ax <- seq(b,e,by=(e-b)/200)

plot(x,y1, type="l", col = "blue", ylim = c(0,ymax))
lines(x,y2, col = "red")
legend("topright", c("target density", "powerlaw"), fill=c("blue", "red"))


majDensity <- function(x, c, a){
  Tmin <- (c^2)/3

  if(x>Tmin){
    return((1-0.083264) * powerlaw(x, a, Tmin))
  }
  else {return(0.083264 * dunif(x,0,Tmin))}
}
c <- 2
y3 <- sapply(x, majDensity, c = c, a = 1.3)
y1 <- sapply(x, distrib1, c = c)
plot(x,y1, type = "l", ylim = c(0,max(y3,y1)), ylab = "y")
lines(x,y3, col ="blue")
legend("topright", c("target", "proposal, a = 1.3"), fill=c("black", "blue"))

library(poweRlaw)
randomnumber <- function(t,a){
  numb <- runif(1)
  if(numb<= 0.08326){
    return(runif(1,0,t))
  }
  else{
    return(rplcon(1,t,a))
  }
}

CompleteDist <- function(c, a, rej){
  z <- TRUE
  res <- 0

  Tmin <- (c^2)/3
  xmaj <- (c^2)/(3-2*a)
  cmaj <- distrib1(xmaj,c)/majDensity(xmaj,c, a)

  while (z == TRUE) {
    y <- randomnumber(Tmin,a)
    u <- runif(1)
    if(u <= distrib1(y, c) / (cmaj*majDensity(y,c,a))){
      res <- y
      z <- FALSE
    }
    if(rej){
    rejected <<- rejected + 1
    }
  }
  return(res)
}

rDist <- function(n,c,a ,rej = FALSE){
 return(replicate(n, CompleteDist(c, a, rej)))
}




rejected <- 0
par(mfrow=c(1,3))
datarDist1 <- rDist(2000, 2, 1.3, TRUE)
datarDist12 <- datarDist1[datarDist1 < 200]
rejected1 <- rejected

rejected <- 0
datarDist2 <- rDist(2000, 9, 1.3, TRUE)
datarDist22 <- datarDist2[datarDist2 < 200]
rejected2 <- rejected

rejected <- 0
datarDist3 <- rDist(2000, 15, 1.3, TRUE)
datarDist32 <- datarDist3[datarDist3 < 200]
rejected3 <- rejected

hist(datarDist12, breaks = 30, main = "c = 2", xlab = "data")
hist(datarDist22, breaks = 30, main = "c = 9", xlab = "data")
hist(datarDist32, breaks = 30, main = "c = 15", xlab = "data")



results <- matrix(c(mean(datarDist1), var(datarDist1), mean(datarDist2), var(datarDist2), mean(datarDis
colnames(results) <- c("c=2", "c=9", "c=15")
row.names(results) <- c("mean","variance")

knitr::kable(results)

resrejected <- numeric()
c <- seq(1.1,20, by = 0.1)
for (ci in c) {
  rejected <- 0
  rDist(2000, ci, 1.12, TRUE)
  RR <- (rejected / 2000)
  resrejected <- append(resrejected ,1-(1/RR))
}

plot(c,resrejected, type="l", ylab= "rejection rate")
rLaplace <- function(n, mean = 0, alpha = 1){
  b <- 1/alpha
  u <- runif(n)
  res <- mean - (b*sign(u-0.5) * log(1-(2*abs(u-0.5))))
  return(res)
}
library(rmutil)
c1 <- rgb(173,216,230,max = 255, alpha = 80, names = "lt.blue")
c2 <- rgb(255,192,203, max = 255, alpha = 80, names = "lt.pink")

hist(rlaplace(10000, 2,1), 100,
     col = c1,
     main = "Comparison of rlaplace function from rmutil with our rLaplace",
     xlab = "values")
hist(rLaplace(10000, 2,1), 100, col = c2, add = TRUE)
legend("topright", c("rLaplace", "rlaplace"), fill=c(c2, c1))
DE <- function(x, mean = 0,alpha = 1){
  return((0.5*alpha)*exp((-alpha)*abs(x-mean)))
}

genNorm <- function(c, rej){
  z <- TRUE
  res <- 0
  while (z == TRUE) {
    y <- rLaplace(1)
    u <- runif(1)
    if(u <= pnorm(y) / (c*DE(y))){
      res <- y
      z <- FALSE
    }
    if(rej){
    rejected <<- rejected + 1
    }
  }
  return(res)
}

rNorm <- function(n,c,rej = FALSE){
 return(replicate(n, genNorm(c, rej)))
}
# x <- seq(-5,5, by=0.1)
#
# test <- function(x){ return((dnorm(x)/DE(x)))}
# cde <- sapply(x, test)
# cnorm <- dnorm(x)
#c <- max(cnorm/cde)

## why it calculates 1,35 ???
```

```r
c <- 1.3154892

rejected <- 0
datarNorm <- rNorm(2000, c, TRUE)
datarnorm <- rnorm(2000)

b <- min(c(datarNorm,datarnorm))
e <- max(c(datarNorm,datarnorm))
ax <- seq(b,e,by=(e-b)/50)



hist(datarNorm, breaks = ax,
     col = c1,
     main = "Comparison of rnorm()  with our rNorm()",
     xlab = "values",
     xlim = range(datarNorm,datarnorm))

hist(datarnorm, breaks = ax, col = c2,  xlim = range(datarNorm,datarnorm),  ylim = c(0,250), add = TRUE
legend("topright", c("rnorm", "rNorm"), fill=c(c2, c1))
compare <- matrix(c(mean(datarNorm), var(datarNorm), mean(datarnorm), var(datarnorm)), nrow = 2, byrow =
row.names(compare) <- c("rNorm()", "rnorm()")
colnames(compare) <- c("mean", "variance")
knitr::kable(compare)
RR <- (rejected / 2000)
```