1. Generați 100 000 de valori dintr-o variabilă aleatoare folosind **metoda transformării inverse** pentru repartițiile definite mai jos:

a) *Repartiția logistică* are densitatea de probabilitate $f(x) = \frac{1}{\beta} \frac{e^{-(x-\mu)/\beta}}{(1+e^{-(x-\mu)/\beta})^2}$ și funcția de repartiție $F(x) = \frac{1}{1+e^{-(x-\mu)/\beta}}$.

Rezolvare:

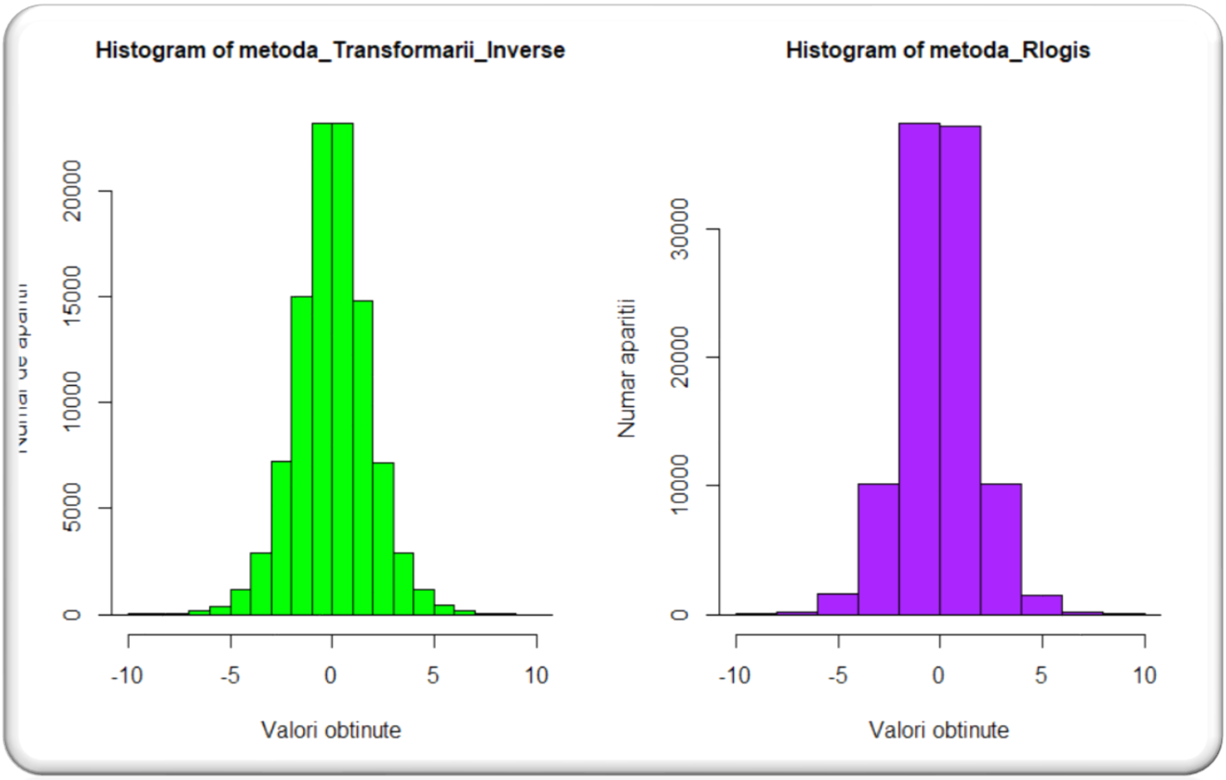
Știm despre funcția de repartiție (F(x)) că este continuă, drept urmare îi vom calcula inversa.

$$y = F^{-1}(x)$$
$$=> F(y) = x$$
$$=> x = \frac{1}{1+e^{-\frac{y-\mu}{\beta}}}$$
$$=> x\left(1+e^{-\frac{y-\mu}{\beta}}\right) = 1$$
$$=> \ x * e^{-\frac{y-\mu}{\beta}} = 1 - x => \ln\left(x * e^{-\frac{y-\mu}{\beta}}\right) = \ln(1-x) => \ln(x) - \frac{y-\mu}{\beta} = \ln(1-x)$$
$$=> y = \beta * \ln(x) - \beta * \ln(1-x) + \mu =>$$
$$=> F^{-1}(x) = \mu + \beta * \ln\left(\frac{x}{1-x}\right) = \mu + \beta * \ln(1-x) + \beta * \ln(x).$$

Codul:

```
n <- 100000

RepartitiaLogistica <- function(N, miu, beta)
{
  U <- runif(N)        #variabila aleatoare uniforma
  return (miu-beta*log(1-U)+beta*log(U)) #inversa funtiei de repartitie
}

metoda_Transformarii_Inverse <- RepartitiaLogistica(n, 0, 1)
metoda_Rlogis <- rlogis(n)

par(mfrow = c(1,2))
#o linie si 2 coloane pentru a afisa ambele histograme

hist( metoda_Transformarii_Inverse,
     #se poate folosi main pentru a pune titlul
     xlab = "Valori obtinute",
     ylab = "Numar de aparitii",
     xlim = c(-10, 10), #intervalul pe x
     cex.main = 1,
     col = "green" )

hist( metoda_Rlogis,
     xlab = "Valori obtinute",
     ylab = "Numar aparitii",
     xlim = c(-10, 10),
     cex.main = 1,
     col = "purple" )
```

Histograme:



Histogram of metoda_Transformarii_Inverse
Valori obtinute
Histogram of metoda_Rlogis
Numar aparitii
Valori obtinute

1. Generați 100 000 de valori dintr-o variabilă aleatoare folosind **metoda transformării inverse** pentru repartițiile definite mai jos:

b) *Repartiția Cauchy* are densitatea de probabilitate $f(x)=\frac{1}{\pi\sigma}\frac{1}{1+\left(\frac{x-\mu}{\sigma}\right)^2}$ și funcția de repartiție $F(x)=\frac{1}{2}+\frac{1}{\pi}\arctan\left(\frac{x-\mu}{\sigma}\right)$.

Comparați rezultatele obținute cu valorile date de funcțiile **rlogis** și respectiv **rcauchy**(funcțiile de repartiție predefinite în **R** pentru repartițiile logistică și respectiv Cauchy). Ilustrați grafic aceste rezultate.

Rezolvare:

Exact ca în cazul precedent, vom calcula inversa funcției de repartiție.

$$y = F^{-1}(x)$$

$$=> F(y) = x$$

$$=> \frac{1}{2}+\frac{1}{n}\left(\frac{y-\mu}{\sigma}\right) = x$$

$$=> x-\frac{1}{2}=\frac{1}{\pi}\operatorname{atan}\left(\frac{y-\mu}{\sigma}\right) => \left(x-\frac{1}{2}\right)\pi = \operatorname{atan}\left(\frac{y-\mu}{\sigma}\right) => \tan\left(\left(x-\frac{1}{2}\right)\pi\right)=\frac{y-\mu}{\sigma}$$

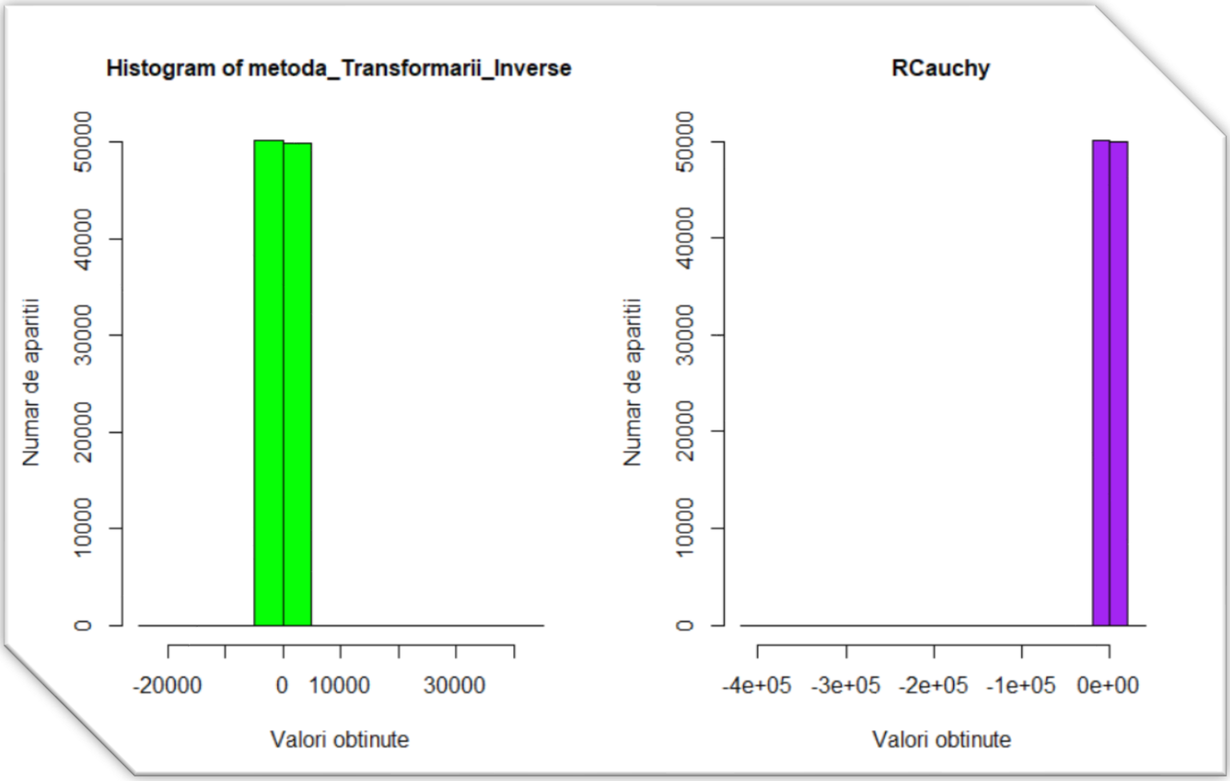
$$=> \sigma\tan\left(\left(x-\frac{1}{2}\right)\pi\right) = y-\mu => y = \mu+\tan\left(\left(x-\frac{1}{2}\right)\pi\right)$$

$$=> F^{-1}(x) = \mu+\tan\left(\left(x-\frac{1}{2}\right)\pi\right)$$

Codul:

```r
n <- 100000

RepartitiaCauchy <- function(N, miu, sigma)
{
  U <- runif(N)
  return(miu+sigma*tan((U-1/2)*pi))
}

metoda_Transformarii_Inverse <- RepartitiaCauchy(n,0,1)
metoda_RCauchy <- rcauchy(n,0,1)

par(mfrow = c(1,2))

hist(metoda_Transformarii_Inverse,
     xlab = "Valori obtinute",
     ylab = "Numar de aparitii",
     cex.main = 1,
     col = "green" )

hist(metoda_RCauchy,
     main = "RCauchy",
     xlab = "Valori obtinute",
     ylab = "Numar de aparitii",
     cex.main = 1,
     col = "purple" )
```

Histograme:



Histogram of metoda_Transformarii_Inverse
Numar de aparitii
0
10000
20000
30000
40000
50000
-20000
0
10000
30000
Valori obtinute
RCauchy
Numar de aparitii
0
10000
20000
30000
40000
50000
-4e+05
-3e+05
-2e+05
-1e+05
0e+00
Valori obtinute

2. $g \sim \mathcal{N}(\overset{\mu}{\overset{\shortparallel}{2}}, \overset{\sigma}{\overset{\shortparallel}{4}})$

$h(x) = e^{\left(-\frac{(x-2)^2}{8}\right)} \cdot (\sin 4x^2 - 2\cos x^2 \sin 9x^2 + 5)$

și $f(x) = c \cdot h(x)$

$g_\theta = \frac{1}{2\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{x-2}{2}\right)^2}$

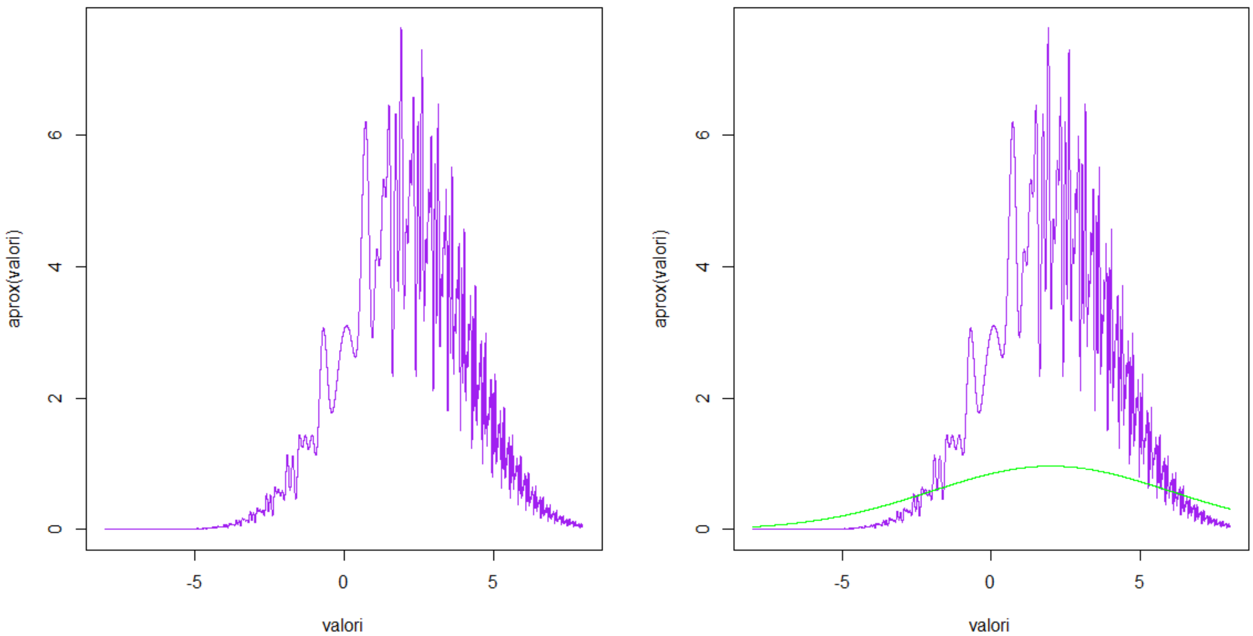
$\frac{h_x}{g_x} = 2\sqrt{2\pi} \cdot (\sin 4x^2 - 2\cos x^2 \sin 9x^2 + 5)$

din $f(x) = c \cdot h(x) \Rightarrow h(x) = \frac{f(x)}{c} \quad \Big| \int$

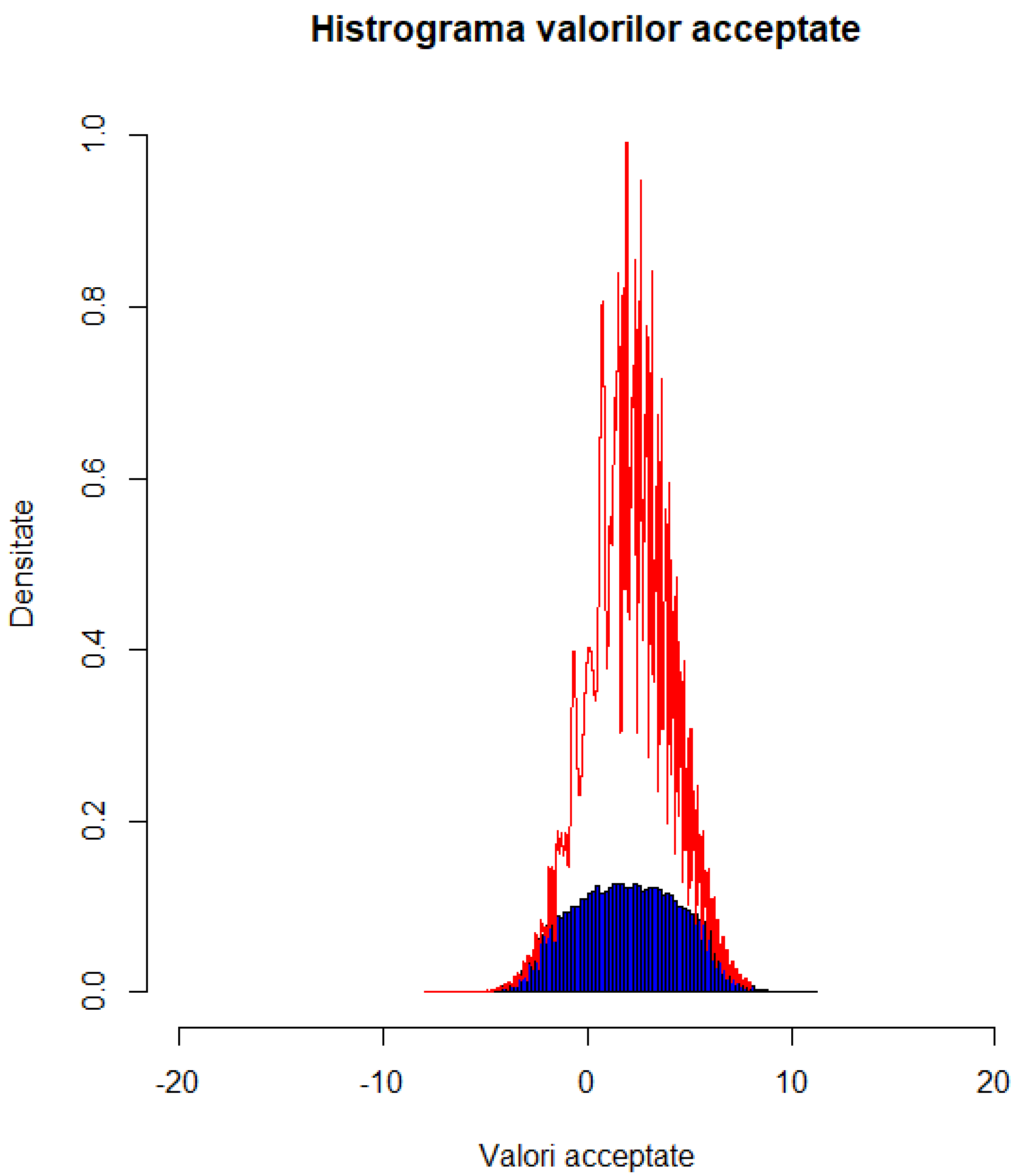
$\int h(x) = \frac{\int f(x)}{\int c}$; dar $\int f(x) = 1 \Rightarrow \overset{\text{valori obț. din cod}}{\overset{\shortparallel}{P.M.}} = \frac{1}{c}$

și aflăm c.

```r
aprox <- function(x)
{
  exp((-(x-2)^2)/8)*(sin((2*x)^2)-2*(cos(x^2))*sin(9*x^2)+5)
}

par(mfrow = c(1,2))

valori <- seq(-8,8,0.0005)
plot(valori, aprox(valori), type="l", col="purple")


raport <- function(x)
{
  return (2*sqrt(2*pi)*(sin(4*x^2)-2*cos(x^2)*sin(9*x^2)+5))
}#am impartit h la N(2,4)

M <- optimise(raport, c(-8,8), maximum = TRUE)
M[2] #in M[2] e val optima

valori <- seq(-8,8, 0.0005)

raport <- function(x)
{
  return (2*sqrt(2*pi)*(sin(2*x))^2-2*((cos(x))^2)*sin(3*x)^2+5)
}
M <- optimise(raport, c(-8,8), maximum = TRUE)
#optimise da 2 valori, de asta adaugam MAx, pt a o lua pe a doua

valori <- seq(-8,8, 0.0005)

plot(valori, aprox(valori), type="l", col = "purple")
lines(valori, dnorm(valori,2,4)*M[[2]], col = "green")

#
```


aprox(valori)
valori
0
2
4
6
-5
0
5
aprox(valori)
valori
0
2
4
6
-5
0
5

```r
valori_pe_care_le_retinem <- c()
n <- 100000 #observatii din densitate
contor <- 0 #numaram incercarile bune
i <- 1

while( i <= n )
{
  u <- runif(1,0,1) #generez din uniforma
  x <- rnorm(1,2,4) #generez din normala standard
  if( u <= aprox(x)/(M[[2]]*dnorm(x,2,4)))
  {
    valori_pe_care_le_retinem [contor] <- x
    contor <- contor + 1
  }
  i <- i + 1
}

# procentul de valori:
p <- contor/n

M[[2]]*p #integrala lui aprox
integrate(aprox, -Inf, Inf)

constanta <- 1/(M[[2]]*p) #calcuez valoarea aproximativa a constantei


hist(valori_pe_care_le_retinem , breaks=100 , freq = FALSE, col = "blue",
     xlab = "Valori acceptate",
     ylab = "Densitate",
     ylim = c(0,1),
     xlim = c(-20, 20),
     main = "Histrograma valorilor acceptate")

lines(valori, constanta*aprox(valori), col = "red", type="l")
#graficul functiei normalizate, breaks discretizare
# cand inmultesc M[2], nu-l ia; insa cu [[]], il ia ca valoare
```


Histrograma valorilor acceptate
Densitate
0.0
0.2
0.4
0.6
0.8
1.0
-20
-10
0
10
20
Valori acceptate

3.

Cerință:

Pentru funcția $h(x) = (1 - x^2)^{\frac{3}{2}}$ construiți o aproximare a integralei acesteia pe intervalul [0,1] după cum urmează:

Valoarea integralei poate fi văzută ca fiind media funcției $h(X)$ unde X este repartizată uniform pe [0,1]. Urmărind algoritmul dat de metoda Monte Carlo construiți programul **R** care determină aproximarea acestei integrale. Comparați rezultatul obținut cu cel analitic. Atașați reprezentările grafice pe care le considerați utile pentru a putea observa eficiența metodei.

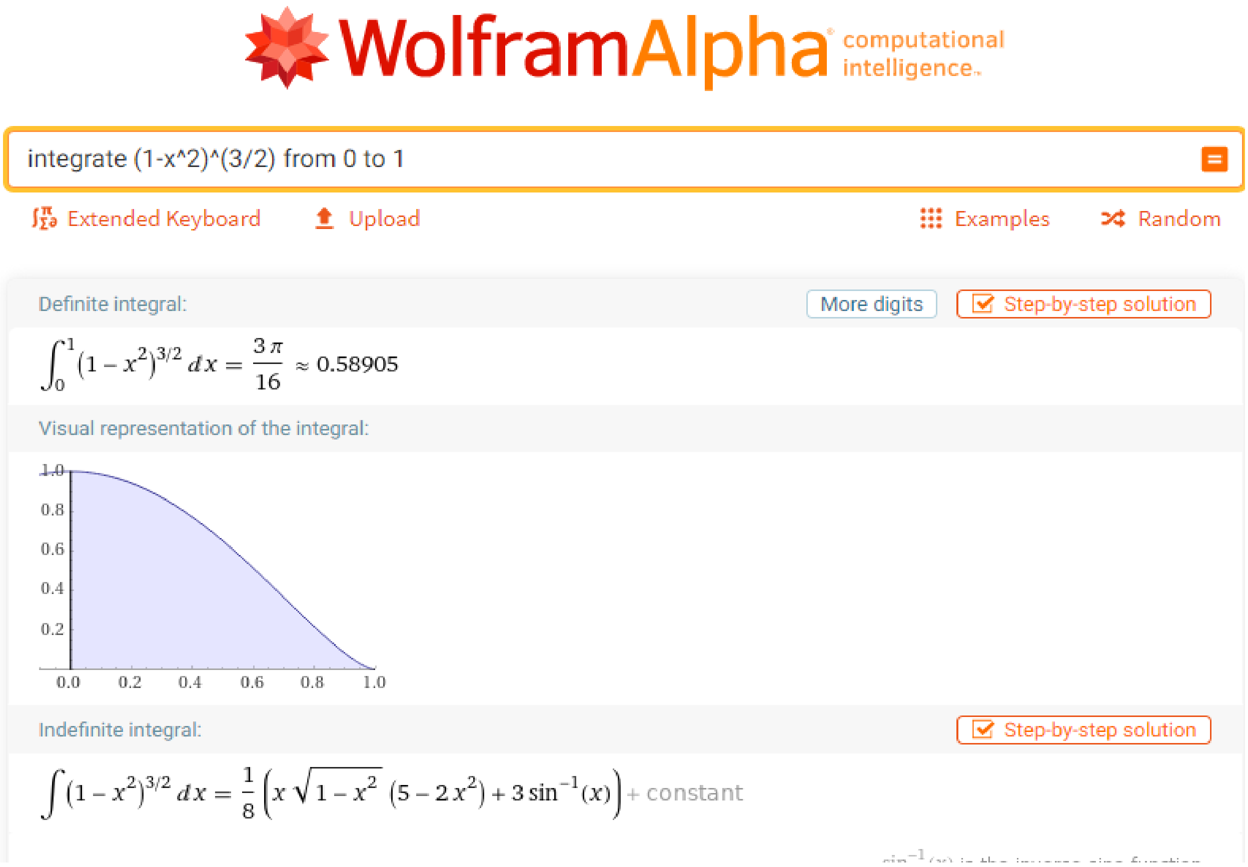
X fiind repartizată uniform pe [0,1], vom putea calcula valoarea integralei cu ajutorul metodei *Monte Carlo* prin $\frac{1}{n}\sum_{i=1}^{n} h(x_i), unde\ x_i este\ uniform\ repartizată\ pe\ intervalul\ mai\ sus\ menționat.$

```
n = 2000

fMonteCarlo = function(x)
{
  return ((1-x^2)^(3/2))
}
Sn = 0           #toate observatiile
valInt = c()    #integrala calculata pentru i observatii
valInt[0] = 0

for( i in 1:n)
{
  Xn = runif(1,0,1)
  Sn = Sn + fMonteCarlo(Xn)
  #fac suma, adun h(xn), cu h functia noastra
  valInt[i] = Sn/i
}
```

Astfel, apelând funcția `valR = integrate(fMonteCarlo, 0, 1)` vom obține valoare integralei folosind metoda MonteCarlo, în timp ce apelând `valIntMC = Sn/n` obținem valoarea prin metoda numerică.

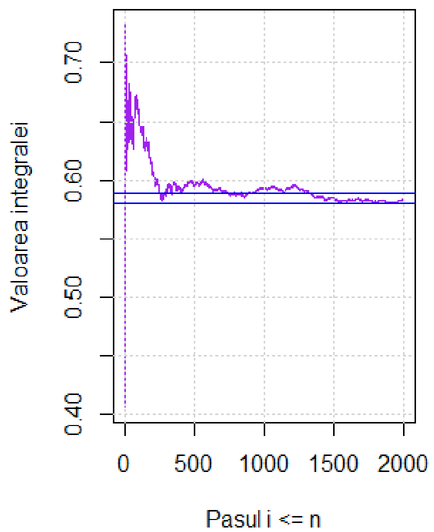
Valoarea analitică o aflăm rezolvând integrala

$\int(1 - x^2)^{\frac{3}{2}} = \frac{1}{8}\left(x\sqrt{1 - x^2}(5 - 2x^2) + 3 * \sin^{-1}(x)\right)$ cu o valoare aproximativă de 0.58905 fiind luată de la 0 la 1, după cum se poate observa mai jos:



Rezultate returnate de R sunt după cum urmează:

```
> valIntMC = Sn/n #valoarea integralei
> valIntMC
[1] 0.5933583
> valR = integrate(fMonteCarlo, 0, 1)
> valR
0.5890486 with absolute error < 1.3e-05
```



Exercitiul 4

4. Construiţi două funcţii în R ***frcpois*** şi respectiv ***frcexp*** care să calculeze **marginea inferioară Rao-Cramer (MIRC)** pentru varianţa estimatorilor parametrilor repartiţiilor ***Poisson*** şi respectiv ***Exponenţială*** pentru un eşantion de dimensiune n (generaţi voi un asemenea eşantion într-o manieră corespunzătoare şi folosiţi-l în apelul funcţiei!).

Repartitia Poisson

Logaritmam functia de repartitie Poisson si calculam cea de a doua derivata a acesteia.

$$p_k = \frac{\lambda^k}{k!} \cdot e^{-\lambda}$$

```
lnf <- expression(log((lambda^x*exp(-lambda))/factorial(x)))

deriv1 <- D(lnf,'lambda')
deriv2 <- D(deriv1,'lambda')
```

Avem de a doua derivate a functiei pentru a o folosi pentru calcularea formulei MIRC, asa ca o vom pastra intr-o functie pentru a fi mai usor de folosit.

```
frcpois <- function(n,lambda,esantion)
{
  derivata2 <- function(lambda,esantion)
  {
    (lambda^((esantion - 1) - 1) * (esantion - 1) * esantion * exp(-lambda) -
       lambda^(esantion - 1) * esantion * exp(-lambda) -
       (lambda^(esantion - 1) * esantion * exp(-lambda) -
        lambda^esantion * exp(-lambda)))/factorial(esantion)/(lambda^esantion * exp(-lambda)/factorial(esantion)) -
        (lambda^(esantion - 1) * esantion * exp(-lambda) - lambda^esantion * exp(-lambda))/
         factorial(esantion) * ((lambda^(esantion - 1) * esantion * exp(-lambda) -
          lambda^esantion * exp(-lambda))/factorial(esantion))/(lambda^esantion *
           exp(-lambda)/factorial(esantion))^2
  }
  #rezultatul derivatei a doua, adaugat intr-o maniera foarte bruta

  MIRC <- 1/(-n*mean(eval(derivata2(lambda,esantion))))
  return(MIRC)
}
```

Iar apoi generam un esantion cu functia predefinita in R *rpois()* pentru functia creata mai sus *frcpois().*

```
X <- rpois(n,12)
MIRC <- frcpois(n,12,X)
```

Valorile date esantionului X vor fi aproximativ 1000, iar MIRC, in cazul nostrum va avea valoarea:

```
> MIRC
[1] 0.01202505
```

Repartitia Exponentiala

Logaritmam functia de repartitie exponentiala si calculam cea de a doua derivata a acesteia.

$$f(x) = \lambda \cdot e^{-\lambda \cdot x}, \lambda > 0$$

```
lnf <- expression(log(lambda*exp(-lambda*x)))

deriv1 <- D(lnf,'lambda')
deriv2 <- D(deriv1,'lambda')
```

Avem de a doua derivate a functiei pentru a o folosi pentru calcularea formulei MIRC, asa ca o vom pastra intr-o functie pentru a fi mai usor de folosit.

```
frcexp <- function(n,esantion,lambda)
{

  derivata2 <- function(lambda,esantion)
  {

    -((exp(-lambda * esantion) * esantion +
      ((exp(-lambda * esantion) * esantion) -
        lambda * (exp(-lambda * esantion) * esantion * esantion)))
        /(lambda * exp(-lambda * esantion)) +
          (exp(-lambda * esantion) - lambda * (exp(-lambda * esantion) * esantion))
          *(exp(-lambda * esantion) - lambda * (exp(-lambda * esantion) * esantion))
          /(lambda * exp(-lambda * esantion))^2)
  }
  #rezultatul derivatei a doua, adaugat intr-o maniera foarte bruta din nou

  MIRC <- 1/(-n*mean(eval(derivata2(lambda,esantion))))
  return(MIRC)
}
```

Iar apoi generam un esantion cu functia predefinita in R *rexp()* pentru functia creata mai sus *frcexp().*

```
X <- rexp(n)
MIRC <- frcexp(n,12,X)
```

Valorile date esantionului X vor fi aproximativ 1000, iar MIRC, in cazul nostrum va avea valoarea:

```
> MIRC
[1] 3.167687e-08
```

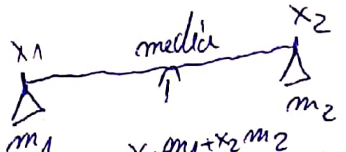
5 a), b) media pentru o variab. discretă:

fie $X \sim \begin{pmatrix} x_1 & x_2 & \dots \\ p_1 & p_2 & \dots \end{pmatrix}$

not $p_i := \mathbb{P}(X = x_i)$

- media $:= E[X] = \sum_{k=0} x_k \cdot \mathbb{P}_k = \sum_{k=0} x_k \mathbb{P}(X = x_k)$ (v.d).

exemplu:



$$\frac{x_1 m_1 + x_2 m_2}{m_1 + m_2}.$$

prin inducție până la n: pt x_i atribui un y_i deoarece repet experimentul de n ori.

$$\left.\begin{array}{l} i)\ y_1 \in x_i \\ ii)\ y_2 \in x_i \\ \vdots \\ y_n \in x_i \end{array}\right\} \Rightarrow \frac{y_1 + \dots + y_n}{n} \simeq \frac{1}{n} \sum_{x} x n \mathbb{P}(X = x) = \sum_{x} x \mathbb{P}(X = x).$$

$$:= E[X]$$

- variab. continuă: $\to$ adm. densitate i.e. $E[X] = \int_{\mathbb{R}} x f(x)\, dx$, $f(x)$ - densitate

- varianța la v.d: $Var(X) = \sum_{x} (x - E[X])^2 \mathbb{P}(X = x)$

$$:= E[X^2] - E[X]^2$$

- varianța la v.c: $var(X) = \int_{\mathbb{R}} (x - E[X])^2 f(x)\, dx.$

Scanned with CamScanner

5c), d) fct. generatoare de momente:

$E[x] \to E[e^{tx}]$ (procedez asemănător la medie (folosind $\sum$, însă $x \to e^{tx}$)).

media: $\int_{-\infty}^{\infty} e^{tx} \cdot \underbrace{f(x)}_{\text{densitatea}} dx = E[x]$

prin LNM: $x_1, \dots, x_n \sim f_{\theta \to \text{estimator}}$, media $= \mu$ și $\sigma^2 =$ dispersia;
(analogie)

fie $\underbrace{\mu_n = \overline{x_n}}_{\text{media empirică}}$ și $\underbrace{\sigma_n^2 = S_n^2}_{\text{variație eșantion}}$ (estimatori)

$$\Rightarrow \begin{cases} \overline{x_n} \xrightarrow{\mathbb{P}} \mu = E_\theta[x_1] \\ S_n^2 \xrightarrow{\mathbb{P}} \sigma^2 = Var_\theta[x_1]. \end{cases}$$

Scanned with CamScanner

6. $F(x) = \frac{1}{1+e^{-\frac{x-\mu}{\beta}}}$

Calc. fct. inversă a repartiției: $F(x) = u \Leftrightarrow 1 + e^{-\frac{x-\mu}{\beta}} = \frac{1}{u}$

$\Leftrightarrow e^{-\frac{x-\mu}{\beta}} = \frac{1}{u} - 1 \quad \Big| \cdot \ln \Leftrightarrow \ln\left(\frac{1}{u} - 1\right) = -\frac{x-\mu}{\beta} \Leftrightarrow x - \mu = -\beta \ln\left(\frac{1-u}{u}\right)$

$\Rightarrow x = \mu - \beta \ln(1-u) + \beta \ln(u)$

fct. de verosim: $L(x_1, \dots, x_n | \theta) = \prod_{i=1}^{n} \frac{e^{-\frac{(x-\mu)}{\beta}}}{\beta\left(1+e^{-\frac{(x-\mu)}{\beta}}\right)^2}$

În mod analog pt Cauchy: $F(x) = u$

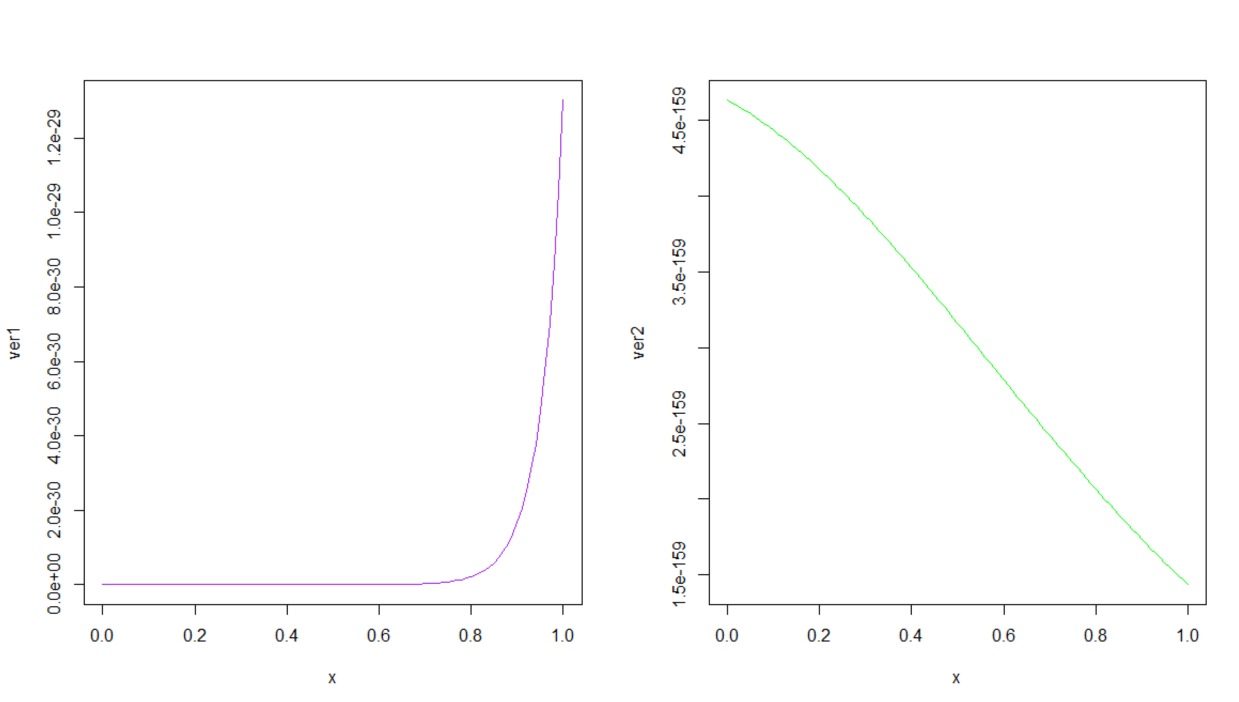
$x = \mu + \delta \cdot tg\left(\pi \frac{(2u-1)}{2}\right)$

fct. de verosim: $L(x_1, \dots, x_n | \theta) = \prod_{i=1}^{n} \frac{1}{\pi\delta} \cdot \frac{1}{1+\left(\frac{x-\mu}{\delta}\right)^2}$.

```r
#a
n <- 100

##Repartitie logistica
rLogistica <- function(n,miu,beta)
{
  U <- runif(n)
  X <- miu+beta*log(U)-beta*log(1-U)     #inversa functiei
  return(X)
}

f <- rLogistica (n,0,1)
L <- 1;
beta <- 10;


#functia de verosimilitate
ver1  <- function(miu)
{
  for(i in 1:n)
  {
     L <- L*exp(-(f[i]-miu)/beta)/beta*(1+exp(-(f[i]-miu)/beta))^2;
  }
  return(L)
}

par(mfrow = c(1,2))
#o linie si 2 coloane pentru a afisa ambele histograme

```

```r
##Repartitie Cauchy
rCauchy <- function(n,miu,sigma)
{
  U <- runif(n)
  X <- miu+sigma*tan(pi*(2*U-1/2))    #inversa functiei
  return(X)
}

g <- rCauchy(n,0,1)
sigma <- 10;

#functia de verosimilitate
ver2 <- function(miu)
{
  for(i in 1:n)
  {
    L <- L*(1/(pi*sigma))*1/(1+((g[i]-miu)/sigma)^2)
  }
  return(L)
}
```

```r
#b
#folosim functia optimise pentru estimatia lui miu
plot(ver1, col="purple")
niu_optim1 <- optimise(ver1, lower=0, upper=1,maximum = TRUE)
niu_optim1


plot(ver2, col="green")
niu_optim2 <- optimise(ver2, lower=0, upper=1,maximum = TRUE)
niu_optim2
```



ver1
0.0e+00
2.0e-30
4.0e-30
6.0e-30
8.0e-30
1.0e-29
1.2e-29
0.0
0.2
0.4
0.6
0.8
1.0
x
ver2
1.5e-159
2.5e-159
3.5e-159
4.5e-159
0.0
0.2
0.4
0.6
0.8
1.0
x