Construirea unui pachet R pentru lucru cu variabile aleatoare continue

Proiect Probabilitati si Statistica

Anghelea Gabriel-Bogdan
Diyar Mert Daniel
Nicoi Alexandru
Ulmeanu Cristian

Cuprins

1 Introducere

Folosind documentul suport și orice alte surse de documentare considerați potrivite construiți un pachet R care să permită lucru cu variabile aleatoare continue. Pentru a primi punctaj maxim, pachetul trebuie să implementeze cel puțin 8 din următoarele cerințe(oricare 8!):

2 Exercitii

2.1 Exercitiul 1

Fiind dată o funcție f , introdusă de utilizator, determinarea unei constante de normalizare k. In cazul in care o asemenea constantă nu există, afișarea unui mesaj corespunzător către utilizator.

Pentru a calcula constanta de normalizare k, vom determina inversul valorii integralei functiei f de la $-\infty$ la $+\infty$, adica $\frac{1}{\int_{-\infty}^{\infty} f(x)dx}$.

Pentru realizarea calcului, am implementat functia "ConstantaDeNormalizareK", care primeste ca parametru functia. Daca valoarea integralei este egala cu 0, atunci afisam utilizatorului faptul ca functia nu admite constante de normalizare.

```
ConstantaDeNormalizareK <- function(functie)
{
  integrala <- integrate(vectorize(functie), -Inf, Inf)$value;
  if (integrala != 0){
    return(1 / integrala);
  }
  else{
    print("Aceasta functie nu admite constante de normalizare");
  }
}
#ConstantaDeNormalizareK(f1)
#the integral is probably divergent
#ConstantaDeNormalizareK(f2)
#"Aceasta functie nu adminte constante de normalizare"
```

2.2 Exercitiul 2

Verificarea dacă o funcție introdusă de utilizator este densitate de probabilitate.

Pentru realizarea acestui exercitiu am cautat documentatia din cadrul cursului cat si a laboratorului astfel am gasit si am folosit 2 proprietati ce caracterizeaza o functie drepta densitate de probabilitate. Cele 2 proprietati folosite pentru verificare sunt:

1. $f(x) \geq 0$ (Functia este pozitiva)
2. $\int_{-\infty}^{\infty} f(x)dx$

In cadrul verificarii ca functia este pozitiva metoda folosita nu este in totalitate precisa deoarece folosind functia “sequence” iau valorile din cadrul unui interval si calculez functia astfel verificand pozitivitatea, dar din pricina limitarilor functiei aceasta permite un interval finit, si pe langa acesta factor functia permite ca punctele de minim si de maxim ale intervalului sa fie maxim egale cu pas*10E±10.

Apoi am calculat integrala si am verificat ca rezultatul sa fie egal cu 1, am decis sa aproximez rezultatul deoarece am intampinat problema ca rezultatul sa fie foarte aproape de 1 dar sa nu fie acceptat pentru verificare, acesta nefiind egal exact cu 1 (ex. 0.999999999999292).

Functia de verificare am creat-o sa primeasca ca parametrii functia de testare, intervalul pentru verificarea pozitivitatii si pasul, toate fiind inputuri de utilizator.

```
## Verificarea daca o functie introdusa de utilizator este densitate de probabilitate.

densitate <- function(f,lower,upper,pas){
  val <- seq(lower,upper,pas);
  #Verificare functie pozitiva
  if(all(f(val)>=0)=='TRUE'){
    #Verificare integrabilitate
    if(typeof(integrate(Vectorize(f),-Inf,Inf))=='logical' && integrate(Vectorize(f),-Inf,Inf)==FALSE || integrate(Vectorize(f),-Inf,Inf) $ message!="OK"){
      return(FALSE)
    }
    else{
      i = integrate(Vectorize(f),-Inf,Inf) $ value
      if (round(i)==1){
        print("Functia este densitate de probabilitate.")
      }
      else{
        print("Functia nu este densitate de probabilitate.")
      }
    }
  }
  else{
    print("Functia este negativa pe intervalul ales.")
  }
}

# #Functie ce nu indeplineste conditiile
# f1 <- function(x){
# if (x > 0 && x < 4){
#     3/5 * (2*x-6*x^2)
#  }else{
#     0
#  }
# }
# #Functie ce indeplineste conditiile
# f2 <- function(x){
#   if (x > 0 && x < 4){
#     3/20*(x^2-2*x+1)
#   }else{
#     0
#   }
# }
# test <- integrate(Vectorize(f),-Inf,Inf) $ value
# print(test)
#
# densitate(f1,-10000,10000,1)
# densitate(f2,-10000,10000,0.01)
```

2.3 Exercitiul 4

Reprezentarea grafica a densitatii si a functiei de repartitie pentru diferite valori ale parametrilor repartitiei. In cazul in care funcția de repartitie nu este data intr-o forma explicita(ex. repartitia normala) se accepta reprezentarea grafica a unei aproximari a acesteia.

Pentru realizarea calculelor, am implementat urmatoarele functii:

- PDF_to_CDF(f, x) - integreaza functia densitate de probabilitate(PDF) pentru a obtine valoarea functiei de repetitie cumulativa(CDF)
- verif_densitate(f, a, b) - verifica daca functia de densitate respecta conditiile necesare (f pozitiva si probabilitate totala egala cu 1)
- plot_densitate(f, a, b, name) - realizeaza graficul densitatii
- plot_repartitie(F, a, b, name) - realizeaza graficul repartitiei

- plot_repartitie_gen(f, a, b) - functie care trateaza cazul in care este nevoie sa facem graficul pentru functii unde nu cunoastem functia de repartitie
- parse_repartitii_cunoscute(name, CDF, ...) - in functie de numele functiei de repartitie introdus in "name", verifica conditiile necesare ale acelei repartitii si realizeaza, dupa caz, ori graficul repartitiei, ori graficul densitatii.

Ultima functie listata are posibilitatea de a crea grafice pentru 4 tipuri de repartitii: uniform, exp, normal, cauchy.

Formulele necesare pentru repartitiile mentionate sunt urmatoarele:

$$\frac{1}{\pi}\arctan\left(\frac{x-x_0}{\gamma}\right)+\frac{1}{2}$$

Figure 1: Cauchy CDF

$$\frac{1}{\pi\gamma\left[1+\left(\frac{x-x_0}{\gamma}\right)^2\right]}$$

Figure 2: Cauchy PDF

$$1-e^{-\lambda x}$$

Figure 3: Exp CDF

$$\lambda e^{-\lambda x}$$

Figure 4: Exp PDF

$$\frac{1}{2}\left[1+\operatorname{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right]$$

Figure 5: Normal CDF

$$\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

Figure 6: Normal PDF

$$\begin{cases} 0 & \text{for } x < a \\ \frac{x-a}{b-a} & \text{for } x \in [a,b] \\ 1 & \text{for } x > b \end{cases}$$

Figure 7: Uniform CDF

$$\begin{cases} \frac{1}{b-a} & \text{for } x \in [a,b] \\ 0 & \text{otherwise} \end{cases}$$

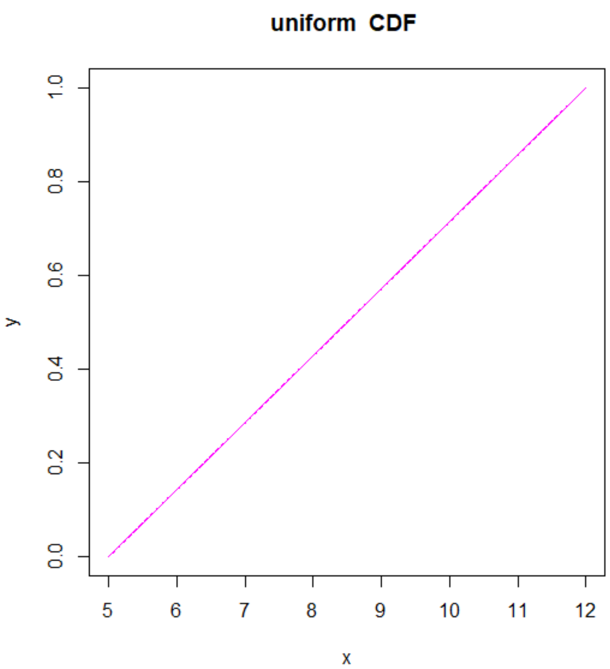
Figure 8: Uniform PDF

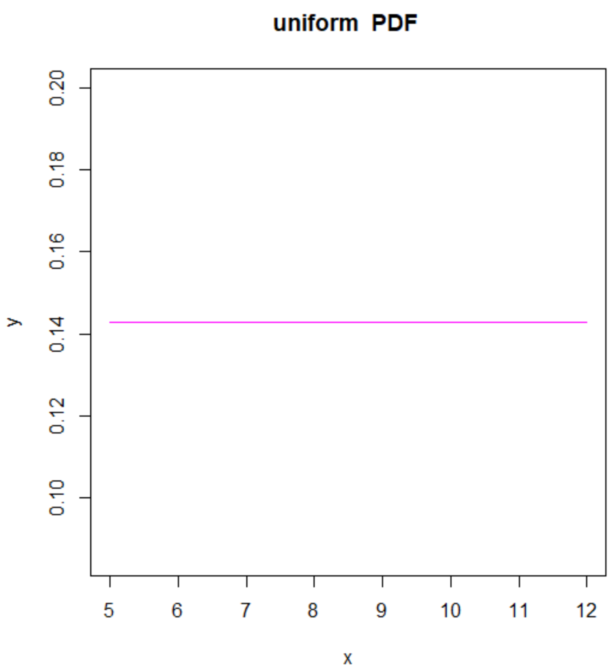
```
# Cerinta: Reprezentarea grafica a densita??ii ??i a func??iei de reparti??ie pentru diferite valori ale
```

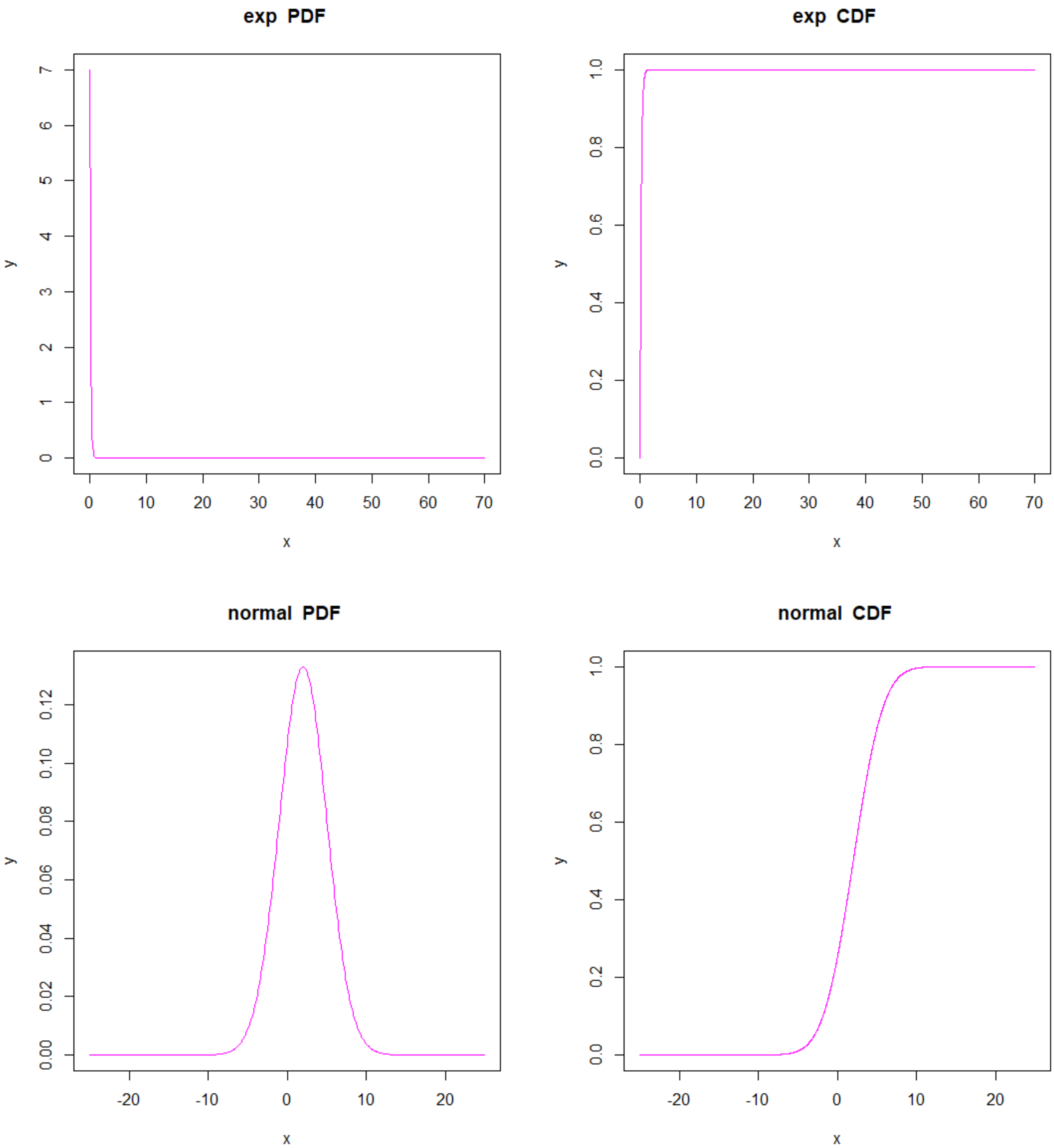
```
# parametrilor reparti??iei. ??n cazul ??n care func??ia de reparti??ie nu este data ??ntr-o forma
# explicita(ex. reparti??ia normala) se accepta reprezentarea grafica a unei aproximari a
# acesteia.

library(pracma)

# Integram functia densitate de probabilitate(cdf) ca sa obtinem
# valoarea functiei de repartitie cumulativa

PDF_to_CDF <- function(f, x)
{
  tryCatch(
    {
      integrate (f, 0, x)$value
    },
    error= function (e){
      print(e)
    }
  )
}

# Verificam conditiile functiei de densitate
# 1. Este pozitiva

verif_densitate <- function(f, a, b) {
  for (x in seq(a, b, 0.1)){
    if( f(x) < 0 ){
      print(paste("PDF nu poate sa aiba valori negative",))
      return(FALSE)
    }
  }

# 2.Probabilitatea totala trebuie sa fie 1

  total <- integral(Vectorize(f), max(-Inf, a), min(b, +Inf))

  if(abs(total - 1) >0.1){
    print ( paste ("Eroare, probabilitatea cumulata totala trebuie sa fie egala cu 1") )
   return ( FALSE )
  }

  return(TRUE)

}

# Afisam graficul densitatii

plot_densitate <- function(f, a, b, name=""){
  # Validam PDF
  if(!verif_densitate(f, a, b)){
    return()
  }

  #Afisam graficul PDF

  ax <- seq(a, b, 0.1)
  ay <- c()
  for (x in ax){
    ay= append(ay, f(x))
  }

  plot (ax , ay , type ="l", main = noquote ( paste ( name , " PDF ")) , col =" magenta ", xlab ="x", ylab="y")
}

#Afisam graficul repartitiei

plot_repartitie <- function(F, a, b, name){
  ax <- seq(a, b, 0.01)
  ay <- c()
  for (x in ax){
    ay = append (ay, F(x))
  }
  plot (ax, ay , col =" magenta ", type ="l", main = noquote ( paste ( name , " CDF ")) , xlab ="x", ylab="y")
```

```
    }
    #Cream o functie de afisare pentru functiile
    # unde nu cunoastem functia de repartitie

      plot_repartitie_gen <- function(f, a, b){

        if( !verif_densitate(f, a, b)){
          return()
        }

        ax <- seq(a, b, 0.01)
        ay <- c()
        for (x in ax) {
          ay = append(ay, PDF_to_CDF(f, x))
        }
        plot (ax , ay , col =" magenta ", type ="l", main ="CDF ", xlab ="x", ylab ="y")

    }

    parse_repartitii_cunoscute <- function(name, CDF=FALSE, ...) {
        params <- list(...)
        if (name == "uniform") {
          if (!is.null(params$a) && !is.null(params$b)) {
            a <- params$a
            b <- params$b
            if (a >= b) {
              return("parametri incorecti")
            }

            f <- function(x) 1 / (b - a)
            F <- function(x)(x - a) / (b - a)
            if (CDF) {
              plot_repartitie(F, a, b, name)
            }
            else {
              plot_densitate(f, a, b, name)
            }
          }
          else {
            return("parametrii necesari nu au fost pasati")
          }
        }
        else if (name == "exp") {
          if (!is.null(params$lambda)) {
            lambda <- params$lambda
            if (lambda <= 0) {
              return("parametrii incorecti")
            }

            f <- function(x) (lambda * exp(1)^(-lambda * x))
            F <- function(x) (1 - exp(1)^(-lambda * x))
            if (CDF) {
              plot_repartitie(F, 0, 70, name)
            }
            else {
              plot_densitate(f, 0, 70, name)
            }
          }
          else {
            return("parametrii necesari nu au fost pasati")
          }
        }
        else if (name == "normal") {
          if (!is.null(params$mu && !is.null(params$sigma))) {
            mu <- params$mu
            sigma <- params$sigma
            if (sigma <= 0) {
              return("parametrii incorecti")
            }

            f <- function(x) ((1 / (sigma * sqrt(pi * 2)))*(exp(1)^((-(x - mu)^2)/(2  * sigma ^ 2))))
            F <- function(x) (pnorm(x, mu, sigma))

```

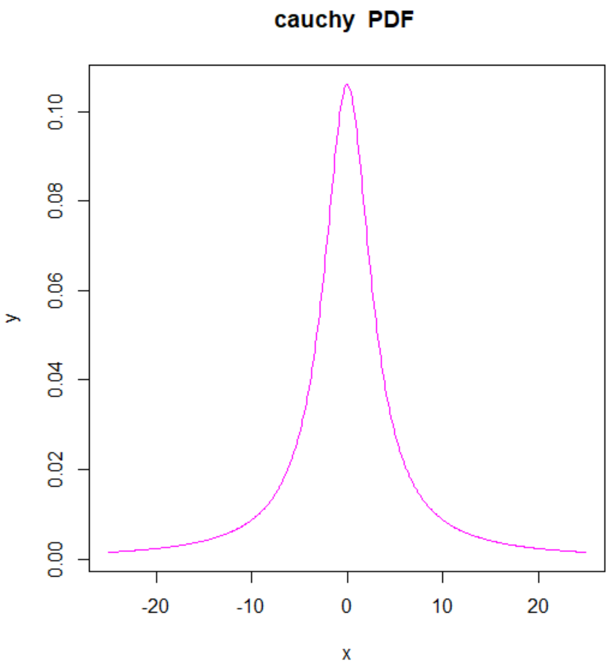
```
        if (CDF) {
          plot_repartitie(F, -30, 30, name)
        }
        else {
          plot_densitate(f, -30, 30, name)
        }
      }
      else {
        return("parametrii necesari nu au fost pasati")
      }
    }
    else if (name == "cauchy") {
      if (!is.null(params$location && !is.null(params$scale))) {
        location <- params$location
        scale <- params$scale
        if (scale <= 0) {
          return("parametrii incorecti")
        }

        f <- function(x) 1 / (pi * scale * (1 + ((x - location) / (scale))^2))
        F <- function(x) (1 / pi) * atan((x - location) / scale) + 1 / 2

        if (CDF) {
          plot_repartitie(F, -30, 30, name)
        }
        else {
          plot_densitate(f, -30, 30, name)
        }
      }
    }
    else {
      print("repartitie necunoscuta")
    }
  }

#EXEMPLE:

# parse_repartitii_cunoscute("uniform", FALSE, a=5, b=12)
# parse_repartitii_cunoscute("uniform", TRUE, a=5, b=12)
# parse_repartitii_cunoscute("exp", FALSE, lambda=7)
# parse_repartitii_cunoscute("exp", TRUE, lambda=5)
# parse_repartitii_cunoscute("normal", FALSE, mu=2, sigma=3)
# parse_repartitii_cunoscute("normal", TRUE, mu=2, sigma=3)
# parse_repartitii_cunoscute("cauchy", FALSE, location=0, scale=3)
# parse_repartitii_cunoscute("cauchy", TRUE, location=0, scale=3)
# plot_repartitie_gen(function(x) x / 22, 0, 2)
```

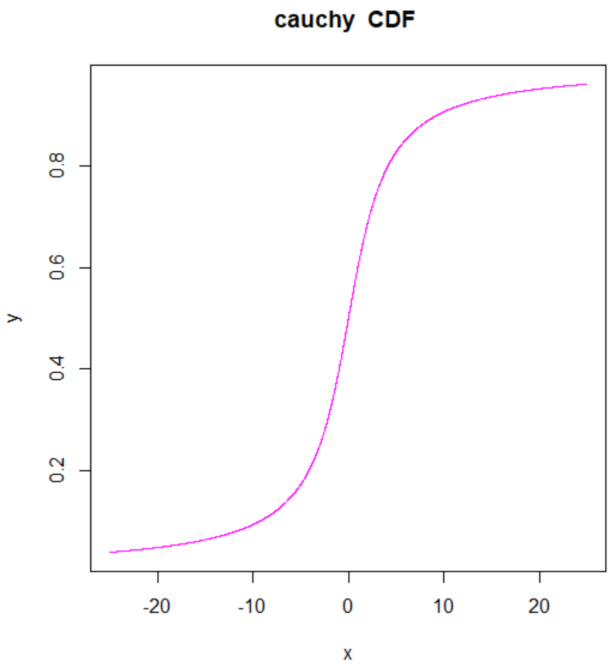






exp PDF
y
0
1
2
3
4
5
6
7
0
10
20
30
40
50
60
70
x
exp CDF
y
0.0
0.2
0.4
0.6
0.8
1.0
0
10
20
30
40
50
60
70
x
normal PDF
y
0.00
0.02
0.04
0.06
0.08
0.10
0.12
-20
-10
0
10
20
x
normal CDF
y
0.0
0.2
0.4
0.6
0.8
1.0
-20
-10
0
10
20
x





2.4 Exercitiul 5

Calculul mediei, dispersiei și a momentelor inițiale și centrate pană la ordinul 4(dacă există). Atunci cand unul dintre momente nu există, se va afișa un mesaj corespunzător către utilizator.

Pentru realizarea cerintei, ne vom folosi de urmatoarele formule:

$$E(x) = \int_{-\infty}^{\infty} x * f(x)dx \qquad \text{Media unei v.a continue}$$

$$Var(x) = \int_{-\infty}^{\infty} (x - E(x))^2 * f(x)dx \qquad \text{Dispersia unei v.a continue}$$

$$\mu_r = \int_{-\infty}^{\infty} (x - E(x))^r * f(x)dx \qquad \text{Moment centrat de ordin r}$$

$$m_r = \int_{-\infty}^{\infty} x^r * f(x)dx \qquad \text{Moment initial de ordin r}$$

Pentru realizarea calculelor, am implementat urmatoarele functii:

- media(f) - primeste functia ca parametru, returneaza valoarea lui E(x)
- dispersia(f) - primeste functia ca parametru, returneaza valoarea lui Var(x)
- momentCentrat(f) - primeste functia si ordinul ca parametri, returneaza valoarea lui μ_r
- momentInitial(f) - primeste functia si ordinul ca parametri, returneaza valoarea lui m_r

Pentru realizarea celor 4 momente initiale si 4 momente centrate, am realizat functia patruMomente, care realizeaza pentru ordinele de la 1 la 4 cele 8 momente.

```
  media<-function(f){
    result=tryCatch({
      return(integrate(function(x){
        produs<-x*f(x);
```

```
      return(produs);
    },-Inf,Inf)$value);
  }, eroare = function(e){
    print("Nu se poate calcula media");
    return(0);
  });
}

dispersia<-function(f){
  medie<-media(f)
  if(medie==0){
    print("Nu se poate calcula dispersia fara medie");
    return(0);
  }
  result=tryCatch({
    return(integrate(function(x){
      produs<-((x-medie)^2)*f(x);
      return(produs);
    },-Inf,Inf)$value);
  }, eroare = function(e){
    print("Nu se poate calcula dispersia");
    return(0);
  });
}

momentCentrat<-function(f,ordin){
  medie<-media(f)
  if(medie==0){
    warning(c("Nu se poate calcula momentul centrat fara medie"));
    return(0);
  }
  result=tryCatch({
    return(integrate(function(x){
      produs<-((x-medie)^ordin)*f(x);
      return(produs);
    },-Inf,Inf)$value);
  }, eroare = function(e){
    print("Nu se poate calcula momentul centrat");
    return(0);
  });
}

momentInitial<-function(f,ordin){
  result=tryCatch({
    return(integrate(function(x){
      produs<-(x^ordin)*f(x);
      return(produs);
    },-Inf,Inf)$value);
  }, eroare = function(e){
    print("Nu se poate calcula momentul initial");
    return(0);
  });
}

patruMomente <- function(f)
{
  print('Primele 4 momente initiale sunt:')
  print(momentInitial(f, 1))
  print(momentInitial(f, 2))
  print(momentInitial(f, 3))
  print(momentInitial(f, 4))
  print('Primele 4 momente centrate sunt:')
  print(momentCentrat(f, 1))
  print(momentCentrat(f, 2))
  print(momentCentrat(f, 3))
  print(momentCentrat(f, 4))
}

#f1 <- function(x){
#   if(x > 0 && x < 20)
#     return(1/20)
#   else
#     return(0)
#}
#patruMomente(f1)
#[1] "Primele 4 momente initiale sunt:"
```

```
#[1] 5.625
#[1] 56.25
#[1] 632.8125
#[1] 7593.75
#[1] "Primele 4 momente centrate sunt:"
#[1] 1.40625
#[1] 16.69922
#[1] 84.04541
#[1] 780.5099
```

2.5 Exercitiul 6

Calculul mediei și dispersiei unei variabile aleatoare g(X), unde X are o repartiție continuă cunoscută iar g este o funcție continuă precizată de utilizator.

Pentru realizarea acestui exercitiu am cautat documentatia din cadrul cursului cat si a laboratorului astfel am gasit si am folosit formulele specifice mediei cat si dispersiei unei variabile aleatorii continue.

$$E(x) = \int_{-\infty}^{\infty} x * f(x)dx \qquad \text{Media unei v.a continue}$$

$$Var(x) = \int_{-\infty}^{\infty} (x - E(x))^2 * f(x)dx \qquad \text{Dispersia unei v.a continue}$$

Pe care le-am adaptat incat sa fie ablicabile in cazului unei variabile aleatoare de tip g(x), si unde X are o repartitie continua cunoscuta. Din moment ce X are o repartitie continua cunoscuta am presupus ca functia densitate de probabilitate a acestuia este cunoscuta, din acest motiv am ales ca functia pentru calcularea mediei cat si pentru calcularea dispersiei sa primeasca ca parametrii funtia g, functia densitate de probabiliate pentru X si intervalul acesteia. Cum x-ul din formula mediei va fi inlocuit cu o functie g(x) rezulta ca f(x) va fi reprezentata de functia densitate de probabilitate a lui X.

$E(x) = \int_{-\infty}^{\infty} g(x) * fX(x)dx$, unde fX(x) este functia densitate de probabilitate.

Acelasi lucru se va intampla si pentru formula dispersiei.

$Var(x) = \int_{-\infty}^{\infty} (g(x) - E(x))^2 * fX(x)dx$, unde fX(x) este functia densitate de probabilitate.

```
##Calculul mediei si dispersiei unei variabile aleatoare g(X), unde X are o repartitie continua cunoscuta iar g este o functie continua precizata de utilizator.

#Verificare functie de repartitie
#x1<x2=>f(x1)<f(x2)
#f(x-0)=f(x)
#lim f(x)=0 cand x - > -Inf
#lim f(x)=1 cand x - > Inf

#Calculare medie
calcul_medie<- function(g,functie_dens_prob,lower,upper){
  integrala_medie <- function(x){g(x)*functie_dens_prob(x)}
  medie <- integrate(Vectorize(integrala_medie),lower,upper) $ value
  return(medie)
}

#Calculare dispersie
calcul_dispersie <- function(g,functie_dens_prob,lower,upper){
  integrala_dispersie <- function(x){(g(x)-calcul_medie(g,functie_dens_prob,lower,upper))^2*functie_dens_prob(x)}
  dispersie <- integrate(Vectorize(integrala_dispersie),lower,upper) $ value
  return(dispersie)
```

```
}

#Testare
#
# f1 <- function(x)(2*(x^3))
# f2 <- function(x)(exp(1)^(-2*x))
# calcul_medie(f1,f2,0,Inf)
# calcul_dispersie(f1,f2,0,Inf)
```

2.6 Exercitiul 8

Afisarea unei “fise de sinteza” care sa contina informatii de baza despre respectiva repartitie(cu precizarea sursei informatiei!). Relevant aici ar fi sa precizati pentru ce e folosita in mod uzual acea repartitie, semnificatia parametrilor, media, dispersia etc.

Pentru a putea afisa o fisa de sinteza, am utilizat libraria ”R6”, mai exact obiecte de tip ”R6Class” alaturi de obiectul ”Dictionary” din libraria ”ml3misc”(`https://www.rdocumentation.org/packages/mlr3misc/versions/0.5.0/topics/Dictionary`).

Astfel, pe baza de object oriented programming, am realizat o clasa cu structura urmatoare:

- Sursa
- Notatie
- Parametri
- Domeniu
- PDF
- CDF
- Media
- Mediana
- Utilizari

Structura aceasta a fost organizata similar felului in care repartitiile sunt prezentate pe Wikipedia. In lista de simteza se regasesc repartitii, atat din curs, cat si din studiu suplimentar(Cauchy).

```
library(mlr3misc)
library(R6)

# Cerinta: Afi??area unei "fi??e de sinteza" care sa con??ina informa??ii de baza despre respectiva
# reparti??ie(cu precizarea sursei informa??iei!). Relevant aici ar fi sa preciza??i pentru ce e
# folosita ??n mod uzual acea reparti??ie, semnifica??ia parametrilor, media, dispersia etc.


data = Dictionary$new()

item = R6Class("Item")

            # UNIFORM #

data_ob = Dictionary$new()

data_ob$add("Sursa", R6Class(public = list(print = "Curs + Wikipedia")))

data_ob$add("Notatie", R6Class(public = list(print = "U(a, b)")))

```

```
    data_ob$add("Parametri", R6Class(public = list(print = "-Inf < a < b < Inf")))

    data_ob$add("Domeniu", R6Class(public = list(print = "x in [a, b]")))

    data_ob$add("PDF", R6Class(public = list(print = "1 / (b - a)")))

    data_ob$add("CDF", R6Class(public = list(print = "(x - a) / (b - a)")))

    data_ob$add("Media", R6Class(public = list(print = "(1 / 2) * (a + b)")))

    data_ob$add("Mediana", R6Class(public = list(print = "(1 / 2) * (a + b)")))

    data_ob$add("Utilizari", R6Class(public = list(print = "1. In domeniul economiei pentru inventariere, in special pentru analiza ciclului de viata a unui produs nou.
                                                          2. Solutie pentru erorile de cuantificare, un exemplu fiind conversia analog-to-digital.")))
    data$add("uniform", data_ob)


               # EXPONENTIAL #


    data_ob = Dictionary$new()
    data_ob$add("Sursa", R6Class(public = list(print = "Curs + Wikipedia")))

    data_ob$add("Parametri", R6Class(public = list(print = "labmda > 0")))

    data_ob$add("Domeniu", R6Class(public = list(print = "x in [0, +Inf]")))

    data_ob$add("PDF", R6Class(public = list(print = "lambda * exp(1)^(-lambda * x)")))

    data_ob$add("CDF", R6Class(public = list(print = "1 - exp(1)^(-lambda * x)")))

    data_ob$add("Media", R6Class(public = list(print = "1 / labmda")))

    data_ob$add("Mediana", R6Class(public = list(print = "ln2 / lambda")))

    data_ob$add("Utilizari", R6Class(public = list(print = "Pe baza unui set de date, aceasta distributie poate poate estima precis evenimente care vor avea lic in viitor.
                                                          De exemplu: durata de descompunere a unei particule radioactive;
                                                                      durata dintre 'click'-urile unui detector Geiger.")))

    data$add("exponential", data_ob)


                # NORMAL #

    data_ob = Dictionary$new()
    data_ob$add("Sursa", R6Class(public = list(print = "Curs + Wikipedia")))

    data_ob$add("Notatie", R6Class(public = list(print = "N(mu, sigma^2)")))

    data_ob$add("Parametri", R6Class(public = list(print = "mu in R, sigma ^ 2 >= 0")))

    data_ob$add("Domeniu", R6Class(public = list(print = "x in [-Inf, +Inf]")))

    data_ob$add("PDF", R6Class(public = list(print = "(1 / (sigma * sqrt(pi * 2)))*(exp(1)^((-(x - mu)^2)/(2  * sigma ^ 2)))")))

    data_ob$add("CDF", R6Class(public = list(print = "(1 / 2) * (1 + erf((x - mu) / (sigma * sqrt(2))))")))

    data_ob$add("Media", R6Class(public = list(print = "mu")))

    data_ob$add("Mediana", R6Class(public = list(print = "mu")))

    data_ob$add("Utilizari", R6Class(public = list(print = "Numeroase aplicatii, variand de la fizica pana la biologie, pana si la statisticarezultatelor unui examen. ")))

    data$add("normal", data_ob)


                # CAUCHY #

```

```r
data_ob = Dictionary$new()
data_ob$add("Sursa", R6Class(public = list(print = "Curs + Wikipedia")))

data_ob$add("Parametrii", R6Class(public = list(print = "x0, y > 0")))

data_ob$add("Domeniu", R6Class(public = list(print = "x in [-Inf, Inf]")))

data_ob$add("CDF", R6Class(public = list(print = "(1 / pi) * arctan((x - x0) / y) + (1/2)"
)))

data_ob$add("PDF", R6Class(public = list(print = "1 / (pi * y * (1 + (x - x0) / y)^2)")))

data_ob$add("Mediana", R6Class(public = list(print = "x0")))

data_ob$add("Media", R6Class(public = list(print = "undefined")))

data_ob$add("Utilizari", R6Class(public = list(print = "Utilizata in hidrologie,
distributia Cauchy poate fi aplicata fenomenelor extreme, precum furtuni de o zi sau
inundatii.")))

data$add("cauchy", data_ob)



         # FUNCTIE DE AFISARE #

afisare <- function(distribution_name) {

  out_data = data$get(distribution_name)

  for (key in out_data$keys()) {

    print(noquote(paste(key, ": ", out_data$get(key)$print, sep="")))

  }
}

# EXEMPLE
#afisare("uniform")
#afisare("normal")
#afisare("exponential")
#afisare("cauchy")
```

2.7 Exercitiul 9

Generarea a n valori(unde n este precizat de utilizator!) dintr-o repartiție de variabile aleatoare continue(solicitați material suport pentru partea de simulare).

Pentru realizarea cerintei, am aplicat metoda inversa pentru simularea variabilelor aleatoare conform materialului suport. In cod, am implementat urmatoarele functii:

- integrarePDF(f, x) - primeste functia si capatul din dreapta a integralei ca parametri, returneaza functia densitate de probabilitate
- generareInversa(f,y,stanga,dreapta) - primeste ca parametri functia, y care va fi noul parametru al functiei inverse, capetele din stanga si dreapta ale intervalului de definire , returneaza functia inversa
- generareNValori(f,n,stanga,dreapta) - primeste ca parametri functia, numarul de valori care urmeaza sa fie generate, capetele din stanga si dreapta ale intervalului de definire , afiseaza sirul generat de valori

```r
integrarePDF<-function(f, x){
  tryCatch({
    return(integrate(Vectorize(f), 0, x)$value);
    }, eroare = function(e){
    print(e);
    return(0);
  });
```

```r
  }

  generareInversa<-function(f, y, stanga, dreapta){
    uniroot((function (x) f(x) - y), lower=stanga, upper=dreapta)[1]
  }

  generareNValori<-function(f, n, stanga, dreapta){
    valori <- runif(n, stanga, dreapta)
    export <- c();
    for(y in valori){
      export <- append(export, generareInversa(function(x) (integrarePDF(f, x)), y, stanga, dreapta)$root)
    }
    print(export);
  }

  #f1<-function(x){
  #  if(x >= 0)
  #    return(1-exp(-2*x))
  #  else return(0)
  #}
  #generareNValori(f1,50,0.1,100)
  #[1] 25.446174 53.115397 74.557649 68.807575 30.584462 49.337176 71.337045 72.074684 18.379085 53.966918 76.665689 94.289155 95.916681
  #[14] 18.057581 74.451717 42.136329 62.783452 13.919849 95.225450 89.564206 74.636752 65.059950 11.614840 47.619160 89.173798 46.462880
  #[27] 66.338070 81.690513 30.974663 18.989209 32.714365 59.707631 86.227731 46.973578 93.553083 41.559197 28.137667 35.568942 44.815815
  #[40]  7.876199 81.646375 91.597387 71.536997 52.775758 53.720603 75.384789  3.531075 18.790562 77.463315 38.960205
```

2.8 Exercitiul 10

Calculul covariantei si coeficientului de corelație pentru doua variabile aleatoare continue (Atentie: Trebuie sa folositi densitatea comună a celor doua variabile aleatoare!)

Formula covariantei pentru variabilele aleatoare continue este: $cov(X,Y) = E(XY) - E(X)E(Y)$, unde $E[X]$ reprezinta media variabilei aleatoare continue. Pentru calcularea $E(XY)$ se calculeaza media folosind integrala dubla: $E(XY) = \int_c^d \int_a^b x * y * f(x,y)dxdy$

Pentru a calcula corelatia, folosim urmatoarea formula: $cor(X,Y) = \frac{cov(X,Y)}{D(X)D(Y)}$, unde D(X) reprezinta dispersia variabilei X.

```r
  library(pracma)

  media<-function(f,fst=-Inf,fdr=Inf){
    result=tryCatch({
      return(integrate(function(x){
        produs<-x*f(x);
        return(produs);
      },fst,fdr)$value);
    }, error = function(e){
      print("Nu se poate calcula media");
      return(0);
    });
  }

  dispersia<-function(f,fst=-Inf,fdr=Inf){
    medie<-media(f,fst,fdr)
    if(medie==0){
      print("Nu se poate calcula dispersia fara medie");
      return(0);
    }
    result=tryCatch({
      return(integrate(function(x){
        produs<-((x-medie)^2)*f(x);
        return(produs);
      },fst,fdr)$value);
    }, error = function(e){
      print("Nu se poate calcula dispersia");
```

```
      return(0);
    });
  }

  densitate_marginalaX <- function(f,a,b,c =-Inf,d = Inf)
  {
    a <- max(a,c) #se compara maximul intre limitele inferioare dintre intervalul variabilei opuse si cel conditional
    b <- min(b,d) #se compara minimul intre limitele superioare dintre intervalul variabilei opuse si cel conditional
    tryCatch(
      {
        if(a>b) 0 #se compara capetele intervalului pentru a nu utiliza un interval imposibil
        else function(x) {integrate((function(y) {f(x, y)}), a, b)$value} #folosim formula de calcul din teorema
      }, error = function(e){
        print("Functia de densitate comuna este incorecta!");
      }
    )
  }

  densitate_marginalaY <- function(f,a,b,c =-Inf,d = Inf)
  {
    a <- max(a,c) #se compara maximul intre limitele inferioare dintre intervalul variabilei opuse si cel conditional
    b <- min(b,d) #se compara minimul intre limitele superioare dintre intervalul variabilei opuse si cel conditional
    tryCatch(
      {
        if(a>b) 0 #se compara capetele intervalului pentru a nu utiliza un interval imposibil
        else function(y) {integrate((function(x) {f(x, y)}), a, b)$value} #folosim formula de calcul din teorema
      }, error = function(e){
        print("Functia de densitate comuna este incorecta!");
      }
    )
  }

  covarianta_corelatia<-function(f, xst, xdr, yst, ydr){
    fx = densitate_marginalaX(f, xst, xdr)
    fy = densitate_marginalaY(f, yst, ydr)

    medieX = media(fx, xst, xdr)
    medieY = media(fx, yst, ydr)

    functie<-function(x,y){return(x*y*f(x,y))}

    cov = integral2(functie, xst, xdr, yst, ydr)$Q - (medieX*medieY)

    dispersieX = dispersia(fx, xst, xdr)
    dispersieY = dispersia(fy, yst, ydr)

    cor = cov / (sqrt(dispersieX)*sqrt(dispersieY))

    print(cov)
    print(cor)
  }
  f<-function(x,y){
    return(3/2*(x^2+y^2))
  }
  #covarianta_corelatia(f, 0, 1, 0, 1)
  #[1] 0.125
  #[1] 1.5
```

2.9 Exercitiul 11

Vom calcula densitatea marginala folosind teorema acesteia. In cazul functiei, vom avea 5 parametrii, unde primul parametru reprezinta functia de densitatea comuna a celor doua variabile, al 2-lea si al 3-lea parametru sunt capetele intervalului domeniului variabilei opuse iar parametrii 4 si 5 sunt optionali si vor fi folositi pentru calculul densitatii conditionate, unde acestia reprezinta capetele intervalului conditional. Am facut doua functii, una pentru fiecare variabila aleatoare continua, pentru a calcula integrala in functie de pdf-ul variabilei alese.

Ne vom folosi de urmatoarea teorema: Fie X si Y doua variabile aleatoare continue, avand densitatea comuna continua f(x,y) cu domeniul [a,b] x [c,d]. Pdf-urile marginale sunt:

$$f_X(x) = \int_c^d f(x,y)dy \qquad \text{si} \qquad f_Y(y) = \int_a^b f(x,y)dx$$

```
densitate_marginalaX <- function(f,a,b,c =-Inf,d = Inf)
{
  a <- max(a,c) #se compara maximul intre limitele inferioare dintre intervalul variabilei opuse si cel conditional
  b <- min(b,d) #se compara minimul intre limitele superioare dintre intervalul variabilei opuse si cel conditional
  tryCatch(
    {
    if(a>b) 0 #se compara capetele intervalului pentru a nu utiliza un interval imposibil
    else function(x) (integrate(Vectorize(function(y) (f(x, y))), a, b)$value) #folosim formula de calcul din teorema
    }, eroare = function(e){
      print("Functia de densitate comuna este incorecta!");
    }
  )
}
densitate_marginalaY <- function(f,a,b,c =-Inf,d = Inf)
{
  a <- max(a,c) #se compara maximul intre limitele inferioare dintre intervalul variabilei opuse si cel conditional
  b <- min(b,d) #se compara minimul intre limitele superioare dintre intervalul variabilei opuse si cel conditional
  tryCatch(
    {
      if(a>b) 0 #se compara capetele intervalului pentru a nu utiliza un interval imposibil
      else function(y) (integrate(Vectorize(function(x) (f(x, y))), a, b)$value) #folosim formula de calcul din teorema
    }, eroare = function(e){
      print("Functia de densitate comuna este incorecta!");
    }
  )
}
```

2.10 Exercitiul 12

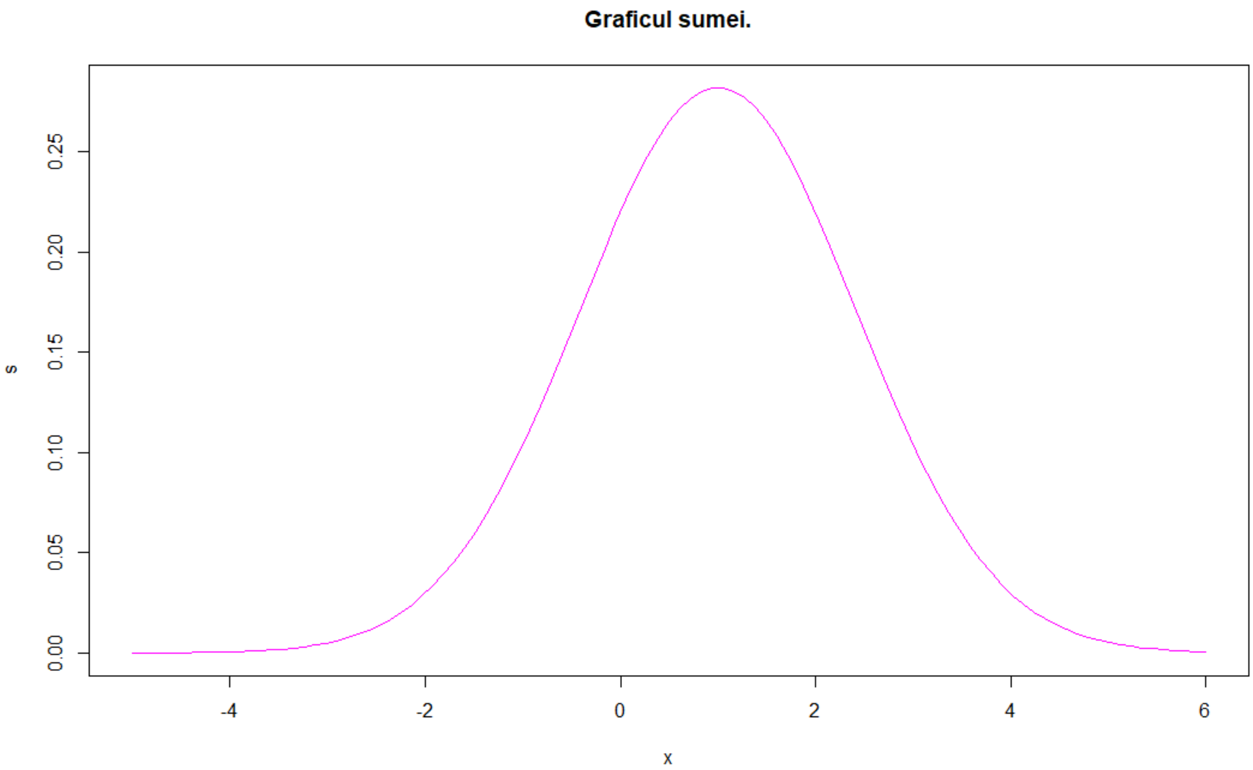
Construirea sumei și diferenței a două variabile aleatoare continue independente(folosiți formula de convoluție)

Pentru construirea sumei si diferentei folosind formula de convolutie m-am documentat de pe: `https://www.afahc.ro/ro/facultate/cursuri/ccg/MSE/C07%20-%20Convolutia%20semnalelor.pdf` pentru a obtine formula generala de convolutie analogica. M-am documentat despre suma si diferenta variabilelor aleatorii continue de pe: `https://math.stackexchange.com/questions/2628366/why-is-the-pdf-of-the-sum-of-two-continuous-random-variables-the-convolution-of`. Apoi am adaptat formula generala in functie de suma/diferenta variabilelor aleatorii continue.

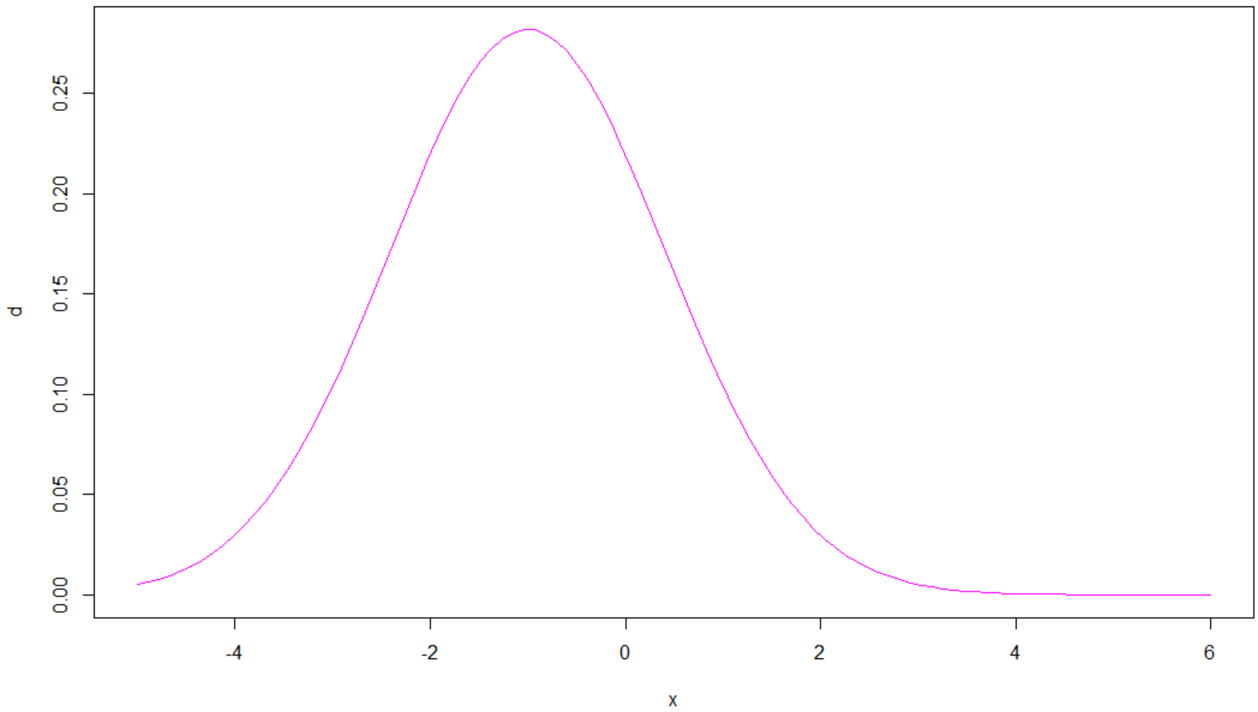
Formula generala de convolutie analogica este: $a(n) = \int_{-\infty}^{\infty} f(\tau) * g(n-\tau)d\tau$. Asadar deoarece suma a 2 variabile aleatorii este Z=X+Y atunci Y=Z-X deci formula de convolutie pentru suma va fi egala cu: $a(z) = \int_{-\infty}^{\infty} f(z-x) * g(x)dx$, iar pentru diferenta Z=X-Y atunci Y=X-Z formula devine: $a(z) = \int_{-\infty}^{\infty} f(x-z) * g(x)dx$.

```
##Construirea sumei si diferentei a doua variabile aleatoare continue independente(folositi formula de convolutie)

```

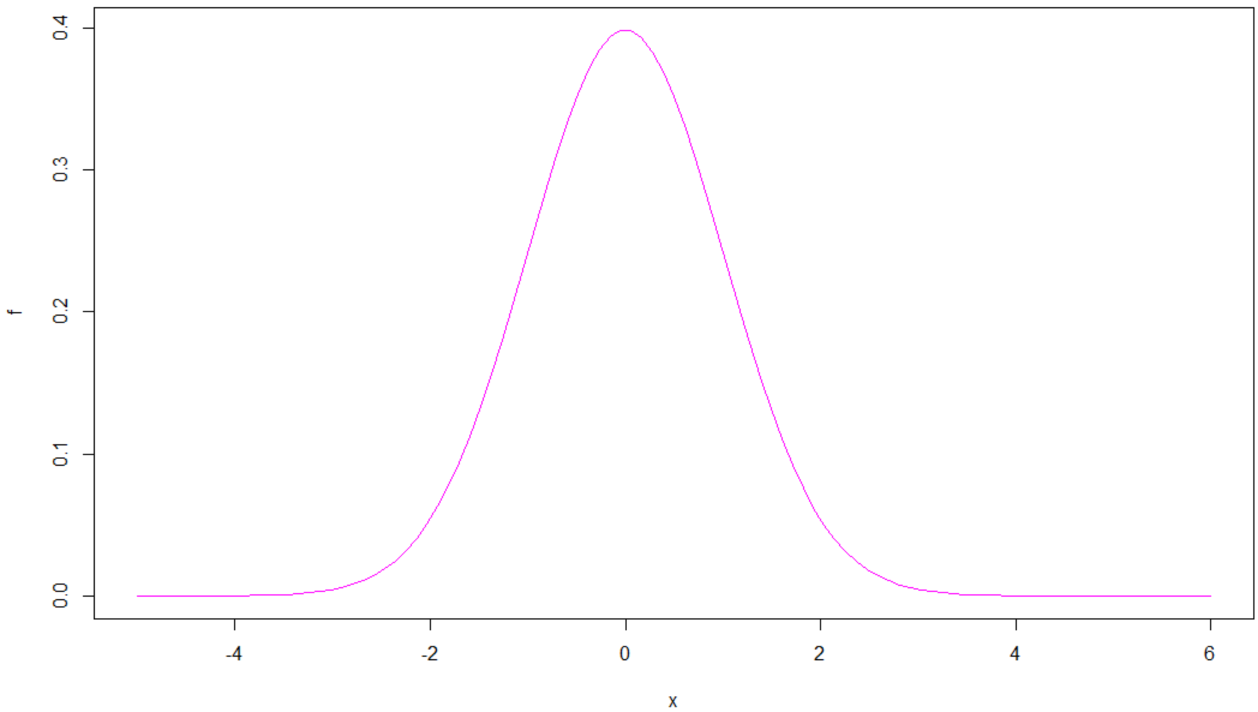
```r
#Formula de convolutie analogica
#a(n)=Integrala(x(t)*y(n-t))


#Calcularea sumei folosind functia de convolutie Z=X+Y unde X si Y sunt variabile aleatorii => Y=Z-X
suma_convolutie <- function (f,g){function(z){
  integrala_con_sum <- function(x){g(z-x)*f(x)}
  sum <- integrate(Vectorize(integrala_con_sum),-Inf,Inf) $ value
  return(sum)
}}

#Calcularea diferentei folosind functia de convolutie Z=X-Y unde X si Y sunt variabile aleatorii => Y=X-Z
diferenta_convolutie <- function (f,g){function(z){
  integrala_con_dif <- function(x){g(x-z)*f(x)}
  dif <- integrate(Vectorize(integrala_con_dif),-Inf,Inf) $ value
  return(dif)
}}

##Testare
# f <- function(x)(dnorm(x))
# g <- function(x) (dnorm(x,mean=1))
# s <- Vectorize(suma_convolutie(f,g))
# d <- Vectorize(diferenta_convolutie(f,g))
#
# plot(f,from=-5,to=6,type="l",col ="magenta")
# plot(g,from=-5,to=6,type="l",col ="magenta")
# plot(s,from=-5,to=6,type="l",col ="magenta")
# plot(d,from=-5,to=6,type="l",col ="magenta")
```

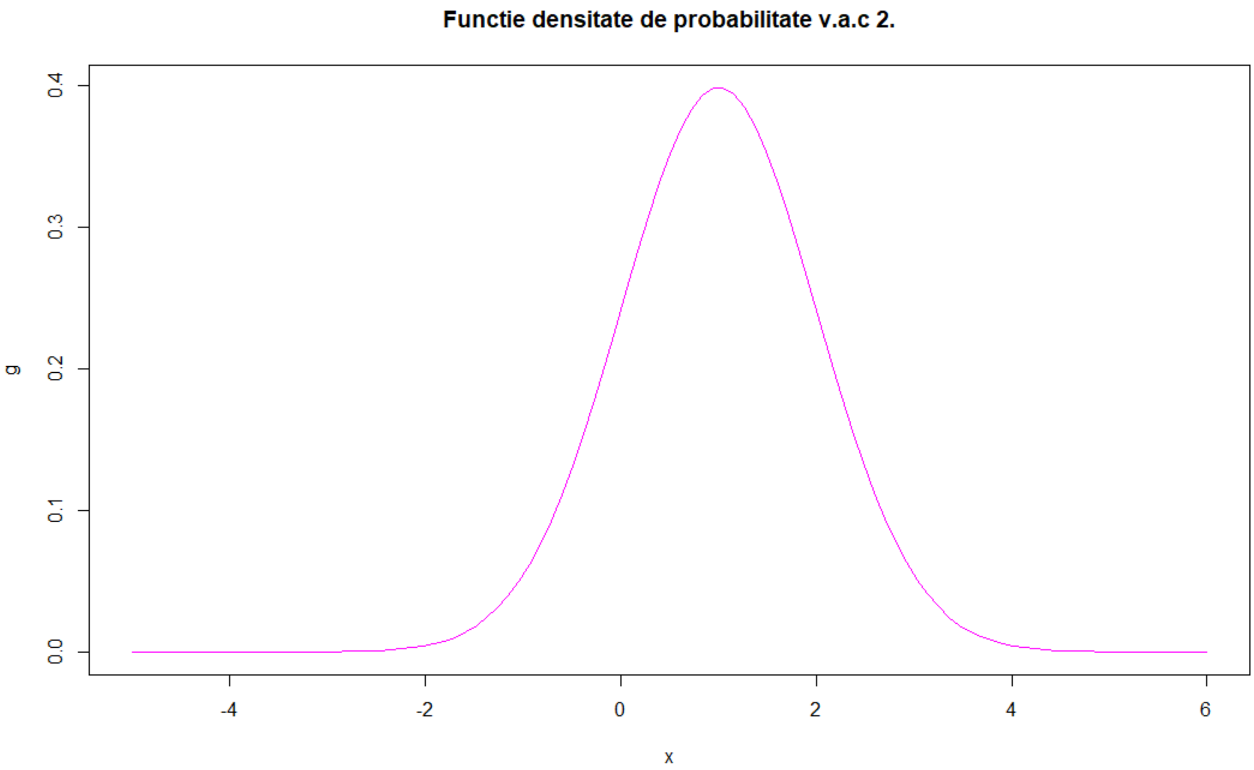


Graficul diferentei.



Functie densitate de probabilitate v.a.c 1.




Functie densitate de probabilitate v.a.c 2.
g
0.0
0.1
0.2
0.3
0.4
-4
-2
0
2
4
6
x