Proiect Probabilitati si Statistica

Alexandra Nanu
Luana Strimbeanu
Matei Mindruleanu
Radu Onea (leader echipa)

Februarie 2021

Contents

1 Introducere **2**

2 Cerinta 1: Constanta de normalizare **3**

3 Cerinta 2: Densitate de probabilitate **6**

4 Cerinta 3: Crearea unui obiect **9**

5 Cerinta 4: Reprezentarea grafica **10**

6 Cerinta 5: Calcularea mediei, dispersiei si momentelor **15**

7 Cerinta 6: Media si Dispersia **17**

8 Cerinta 7: Functia P **18**

9 Cerinta 8: Afisarea unei "fise de sinteza" **19**

10 Cerinta 9: Generare **21**

11 Cerinta 10: Covarianta si Coeficientul de corelatie **22**

12 Cerinta 11: Densitatea marginala/conditionata **24**

13 Cerinta 12: Convolutie **27**

1 Introducere

Am construit un pachet R care sa permita lucrul cu variabile aleatoare continue. Cateva dintre obiectivele atinse sunt: determinarea unei constante de normalizare, reprezentarea grafica a densitatii si a functiei de repartitie pentru diferite valori ale parametrilor repartitiei, calculul mediei și dispersiei unei variabile aleatoare g(X), calculul covariantei si coeficientului de corelatie pentru două variabile aleatoare continue si multe altele, in vederea eficientizarii lucrului in R.

Motivatie: intelegera utilitatii reale a unui pachet R similar cu $discreteRV$, dar pentru variabile aleatoare continue. Dorinta noastra a fost accea ca dupa terminarea proiectului, sa putem da ceva folositor inapoi celor pasionati de lucrul in R, astfel incat proiectul nostru sa capete o importanta practica.

Skill-uri dobandite: capacitatea de a cauta si descoperi solutii la problemele si provocarile aparute, capacitatea de lucru sub presiune, comunicare si lucrul in echipa.

Mod de lucru: Am impartit sarcinile de lucru, fiecare preluand 2-3 functii, paralelizand astfel task-urile de realizat.

2 Cerinta 1: Constanta de normalizare

Enunt: Fiind data o functie f , introdusa de utilizator, determinarea unei **constante de normalizare** k. In cazul in care o asemenea constantă nu exista, afisarea unui mesaj corespunzător către utilizator.

Constanta de normalizare: Conceptul unei constante de normalizare apare în teoria probabilității și într-o varietate de alte domenii ale matematicii. Constanta de normalizare este utilizată pentru a reduce orice funcție de probabilitate la o funcție de densitate de probabilitate cu probabilitate totală a uneia.
Constanta de normalizare - https://ro.qaz.wiki/wiki/Normalizing$_c onstant$

Fie o functie $func$ definita pe o reuniune de intervale. By default, in cazul in care utilizatorul nu introduce capetele domeniului, acestea sunt $(-inf, inf)$.

Cum calculam constanta de normalizare?

Parcurgem rand pe rand toate intervalele ce constituie domeniul functiei si realizam suma tuturor integralelor pe aceste intervale.

Cel mai important pas al rezolvarii noastre si iimlicit pasul care a necesitat cel mai mult research, a fost descoperirea posibilitatii de vectorizare a functiei. O functie vectorizata nu functioneaza doar ca o unica valoare, ci ca un intreg vector de valori. Astfel, nu a mai fost necesar sa parcurgem toate valorile pentru a folosi functia *integrate*, fiind suficient un singur apel al functiei. Un alt aspect interesant al sintazei in R este acela ca semnul "dolar" ce urmeaza functia integrate() pastreaza valoarea finala a integralei. Integrate returneaza un obiect de tipul "integrate" al carui element "value" va contine valoarea estimata a integralei.

Rezultatul nostru este asadar egal cu inversul sumei tuturor integralelor noastre pe aceste intervale: $\frac{1}{sum}$

In cazul aparitiei unei erori la integrare, toata structura noastra este incadrata intr-un tryCatch care rolul de a arunca erorile.

De unde stim ca functioneaza?
Am realizat 3 teste pentru verificare:

```
test1 <- function (x)
{
    return (3*x*x + 2*x+7)
}

test2 <- function (x)
{
  # ifelse este un operator ternar
  # daca se indeplineste conditia, intra pe prima ramura (7 * exp(7^x))
  # altfel pe a doua (0)
  ifelse(x > 0, 7 * exp(-7*x), 0)
}
```

Iar rezultatele print-urilor au fost urmatoarele:

```
# 0.1111111
print (constNorm (test1, domeniu = list (c(0, 1))))
# eroare
print (constNorm (test1))
# 1
print (constNorm (test2))
```

Functia noastra este urmatoarea:

```
# constanta de normalizare este insusi valoarea integralei functiei pe R

constNorm <- function (func, domeniu = list (c (-Inf, Inf)))
{
  # integram functia pe intervalele suport
  # daca este posibil, altfel aruncam eroare
  tryCatch (
    {

      suma <- 0

      # pentru fiecare interval pe care e definita functia
```

```
      # adaugam integrala corespunzatoare

      for (interval in domeniu)
      {

        # IMPORTANT: Vectorizam functia inainte
        # de a o integra pe intervalul curent
        suma <- suma + integrate (Vectorize (func), lower = interval[1], upper = i
      }

      # constanta de normalizare
      return (1 / suma)
    },
    error = function (e)
    {
      # Daca am ajuns aici inseamna ca
      # in blocul tryCatch a fost aruncata o eroare.
      message (paste("Eroare! ", e))
    })
}
```

3 Cerinta 2: Densitate de probabilitate

Enunt: Verificarea daca o functie introdusa de utilizator este densitate de probabilitate.

Definitie: O variabila aleatoare X este continua $\iff \exists$ o functie $f(x)$ astfel incat $\forall\ c \leq d$ avem: $P(c \leq X \leq d) = \int_c^d f(x)\,dx$, unde f este functia densitate de probabilitate.

O functie este densitate de probabilitate daca satisface urmatoarele proprietati:

$$\textbf{1.}\ f(x) \geq 0$$

$$\textbf{2.}\ \int_{-\infty}^{\infty} f(x)\,dx = 1$$

Asadar, in implementarea functiei noastra am testat cele doua rporpietati mentionate mai sus. Am inceput cu proprietatea 2. Dupa vectorizarea functiei, am integrat-o folosindun-ne de functia predefinita *integrate* si am verifcat daca dupa integrare obtinem valoarea 1. In cazul obtinerii unei erori la integrare, aruncam eroarea si printam mesajul corespunzator (EROARE!). (toata structura noastra este scrisa intr-un block $tryCatch$.

Pentru a testa prima proprietate, in schimb, aceea ca $f(x) \geq 0$, am luat un numar de $CONST - INTERVALS = 10000$ de intervale aleatoare pe care am testat proprietatea. Desi prima abordare a verificarii acestui pas era prin calcularea derivatei si a punctelor critice, dupa research am descoperit proprietatile definite de Riemann si Darboux si ne-am folosit de **partitionare a unui interval in subintervale**.

Verificam pe intervale daca integrala este negativa. In acest caz, functia nu este pozitiva, deci nu este nici functie densitate de probabilitate. Altfel, este PDF!

Functia noastra este urmatoarea:

```
# De cate ori verificam sa fie PDF pe fiecare interval al functiei date.
CONST_INTERVALS <- 10000

# O functie este densitate de probabilitate daca indeplineste 2 conditii:
# 1. este pozitiva
# 2. integrala ei pe R este 1
densProb <- function (func, left = -Inf, right = Inf)
{
  # verificam cele doua conditii
  # in caz de esec la integrare, este returnata valoarea FALSE
  tryCatch (
    {

      # integrala functiei pe intervalul primit
      # IMPORTANT: Vectorizam functia inainte de a o integra
      # pe intervalul curent
      integrala <- integrate (Vectorize (func), left, right)$value

      # verificam ca valoarea integralei sa fie egala cu 1
      if (integrala == 1)
      {
        # cream intervalul [L, R]
        L <- max(left, -2e9)
        R <- min(right, 2e9)
        dist = R / 10 - L / 10


        # functia runif() genereaza devieri aleatoare
        # ale distributiei uniforme
        intervals <- runif (CONST_INTERVALS, L, R - dist)

         # Cautam pe intervale (CONST_INTERVALS)
         # pentru a afla daca functia este pozitiva
        for (i in intervals)
        {
          j = i + dist
```

```
          # in cazul in care integrala este negativa,
          # atunci functia nu este pozitiva
          # <=> nu este PDF
          # IMPORTANT: Vectorizam functia inainte de a o integra
          # pe intervalul curent
          if (integrate (Vectorize (f), i, j)$value < 0)
            return (FALSE)
        }
        return (TRUE)
      }
      else
        return (FALSE)
    },

    error = function (err)
    {
      message(paste("EROARE! ", err))
      return (FALSE)
    })
}

#FALSE
densProb(function(x) (3*x*x))
#TRUE
densProb(function(x) (3*x*x), 0, 1)
```

4 Cerinta 3: Crearea unui obiect

Enunt: Crearea unui obiect de tip variabilă aleatoare continuă pornind de la o densitate de probabilitate introdusă de utilizator. Funcția trebuie să aibă opțiunea pentru variabile aleatoare unidimensionale și respectiv bidimensionale.

Cum procedam?

Vom face o clasa folosind sistemul S4 ca mai apoi sa ii facem si un constructor prin care sa initializam fieldurile.Ea va avea doua fielduri globale una fiind functia de repartitie si daca este bidimensionala sau nu.

```
#clasa va tine minte functia de densitate si daca e bidimensionala sau nu
setClass("CRV", representation (
  densitate="function",
  is2Dimensional="logical"
))

#constructorul
CRV <- function(densitate,is2Dimensional= FALSE)
{
#verificare daca a fost pasata o functie buna
 if(!is2Dimensional)
    {j <- integrate(densitate,-Inf,Inf)
    if(j$value-j$abs.error>1 ||j$value+j$abs.error<1  )
  {
    print(integrate(densitate,-Inf,Inf)$value)
    stop("Functia data nu e densitate de probabilitate")
    }
  }

  obj <- new("CRV", densitate = densitate,is2Dimensional=is2Dimensional)
  return (obj)
}
```

5 Cerinta 4: Reprezentarea grafica

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

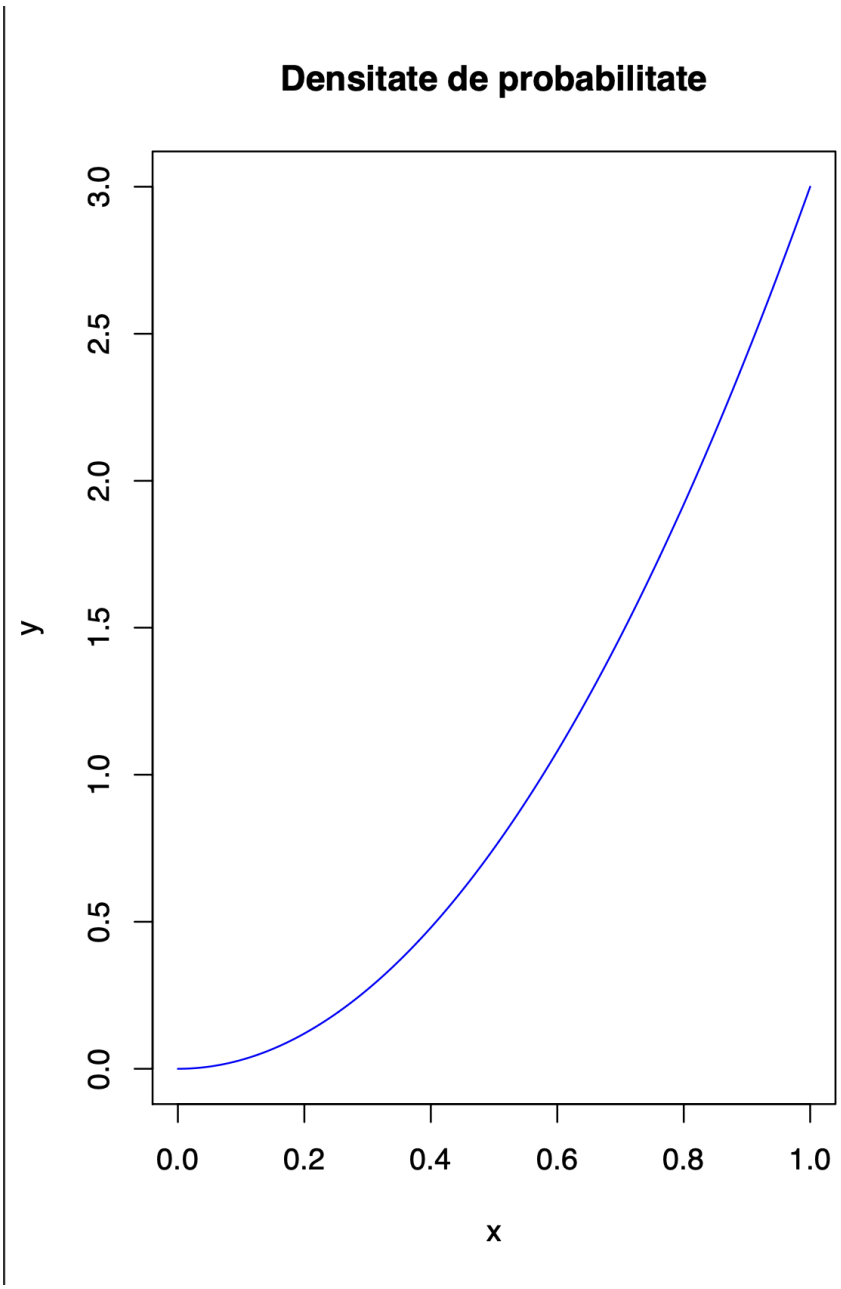
Definitie: Se numeste functie de repartitie a variabilei aleatoare X, functia reala $F : R \to [0, 1]$, $F(x) = P\{X \leq x\} = P(\{e \in E | X(e) \leq x\})$ pentru orice $x \in R$.

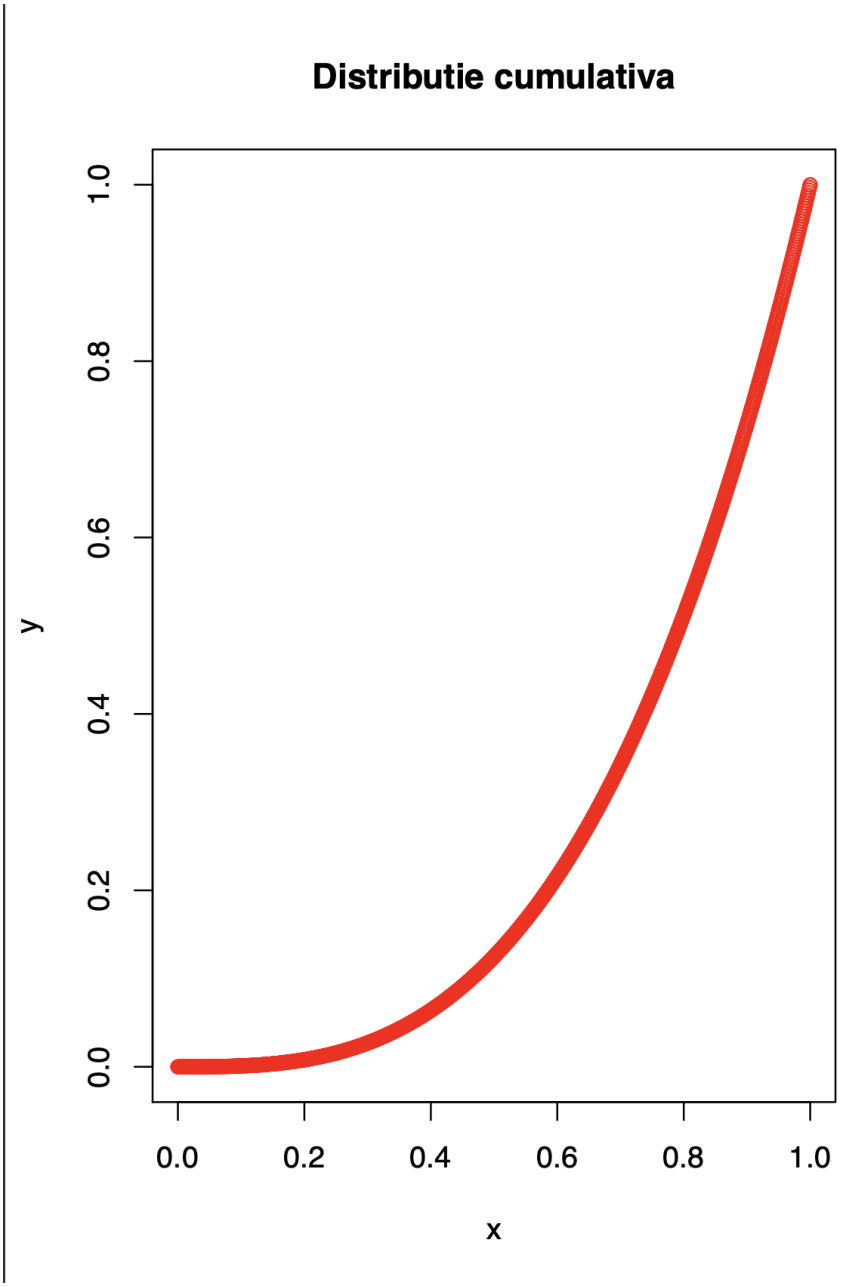
Ne-am folosit de functia **plot** pentru a reprezenta grafic PDF si CDF. Am definit functiile $plotPDF$ si $plotCDF$ ce au ca parametri de intrare functia pe care o primim de la utilizator $func$ pentru care vom genera graficul.

Cum procedam?
Definim vectorul *pace* pe care il generam cu ajutorul functiei *seq*. *seq* primeste ca parametrii: capatul stang al intervalului, capatul drept al intervalului, cat si pasul din cat in cat generam valori.

Pentru functia *plot* putem primi ca parametrii si urmatoarele: *main* etse folosit pentru a afisa titlul, *xlab* pentru a afisa axa-x, *ylab* pentru a afisa axa-y, *typel* pentru a afisa a rezultaul sub forma de linie, *colblue* pentru culoarea albastra.

Am tinut cont ca intotdeauna cand incecam sa integram, sa aruncam eroare in cazul in care integrarea nu functioneaza. Pentru functia $plotCDF$ am folosit un vector *aux* -¿ vector de puncte. In vectorul de puncte *aux* este adaugat de fiecare data punctulcu coordonatele i si integrala de la 0 la i de f.


Densitate de probabilitate
y
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0.0
0.2
0.4
0.6
0.8
1.0
x



Functia noastra este urmatoarea:

```
# Realizez graficul pentru o functie densitate de probabilitate
```

```
# func este functia PDF pe care o primim ca parametru si o vom plota
# left este capatul stang
# right este capatul drept
plotPDF <- function (func, left, right)
{

  # pace va fi vectorul cu ajutorul caruia facem plot
  # intre left si right (din cat in cat afisam punctele)
  pace <- seq (left, right, 0.001)

  #main etse folosit pentru a afisa titlul
  #xlab pentru a afisa axa-x
  #ylab pentru a afisa axa-y
  #type l pentru a afisa a rezulttaul sub forma de linie
  #col blue pentru culoarea albastra
  plot (pace, func (pace), main = 'Densitate de probabilitate',
  xlab = 'x', ylab = 'y', type="l", col="blue")
}

# exemplu de plot pentru PDF
plotPDF(function(x) {3*x*x}, 0, 1)


# Realizez graficul pentru care ploteaza CDF-ul
# unei variabile aleatoare discrete
# func este functia PDF pe care o primim ca parametru si o vom plota
# left este capatul stang
# right este capatul drept

plotCDF <- function (func, left, right)
{
  # aux este vectorul de puncte din grafic
  aux <- c()
  # pace va fi vectorul cu ajutorul
  # caruia facem plot intre left si right
  # (din cat in cat afisam punctele)
  pace <- seq(left, right, 0.001)
```

```
  # parcurgem vectorul de pasi
  for (i in pace)
  {

    # tryCatch-ul este pentru a verifica daca functia este integrabila
    # in caz contrar, este returnata o eroare
    tryCatch (
      {
        # in vectorul de puncte aux este adaugat
        # punctul cu coordonatele i si integrala de la 0 la i de f
        # $value - valoarea finala a integralei
        # integrate returneaza un obiect de tipul "integrate"
        # al carui element "value" contine valoarea estimata
        # a integralei
        aux = append (aux, integrate (func, 0, i)$value)
      },
      error = function (e)
      {
        # afisam eroarea
        message (paste("Eroare! ", e))
      }
    )
  }

  #main etse folosit pentru a afisa titlul
  #xlab pentru a afisa axa-x
  #ylab pentru a afisa axa-y
  #col red pentru culoarea albastra
  plot (pace, aux, main = 'Distributie cumulativa',
  xlab = 'x', ylab = 'y', col = "red")
}

# exemplu de plt cu CDF
plotCDF(function(x) {3*x*x}, 0, 1)
```

6 Cerinta 5: Calcularea mediei, dispersiei si momentelor

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

Cum procedam?

Vom avea cate o functie pentru fiecare moment: Prima functie, pentru medie(momentul initial), primeste ca parametru o variabila aleatoare si calculeaza, folosind formula mediei din teoria probabilitatilor, valoarea ceruta. A doua functie, pentru dispersie(momentul 2), primeste ca parametru o variabila aleatoare si returneaza valoarea integralei din formula pentru calcularea dispersiei din teoria probabilitatilor. A treia functie calculeaza cel de-al treilea moment, este similara cu functia pentru momentul 2, avand schimbata doar puterea la care se ridica functia pentru medie(E). Cea de-a patra functie calculeaza momentul 4 si este similiara cu cele 2 dinaintea ei, ridicand media la o alta putere.

```
    #Formula pentru medie este integrala (inf to inf) xf(x)
E <- function(VA){
  return(integrate(Vectorize(function(x)VA@densitate(x)*x),
  -Inf, Inf)$value)
}
#Formula pentru varianta este integrala (inf to inf) (x-medie)^2f(x)
Var <- function(VA)
{
  return(integrate(Vectorize(function(x)VA@densitate(x)*(x-E(VA))^2),
  -Inf, Inf)$value )
}
#se schimba doar puterea din formula variantei
M3 <- function(VA)
{
  return(integrate(Vectorize(function(x)VA@densitate(x)*(x-E(VA))^3),
  -Inf, Inf)$value )
}
```

```
#se schimba doar puterea din formula variantei
M4 <- function(VA)
{
  return(integrate(Vectorize(function(x)VA@densitate(x)*(x-E(VA))^4),
  -Inf, Inf)$value )
}
```

7 Cerinta 6: Media si Dispersia

Cerinta: 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.

Definitie: Fie X o variabila aleatoare continua, avand densitatea continua $f(x)$ cu domeniul $[a, b]$. Atunci media lui $g(X)$ este:

$$E(g(X)) = \int_a^b g(x)f(x)dx$$

Definitie: Fie X o variabila aleatoare continua cu media E(X). Atunci dispersia lui X este:

$$Var(X) = \int_a^b (g(x) - E(g(X)))^2 f(x)dx$$

Formula: O alta formula pentru calcul dispersiei este

$$Var(X) = E(X^2) - E(X)^2$$

Pentru rezolvarea cerintei am implementat o functie in care se apeleaza mai multe functii auxiliare, una pentru a calcula media variabilei X, iar alta pentru a calcula dispersia lui X, fiecare dintre cele 2 functii avand implementate formulele din definitiile de mai sus. Dispersia am calculat-o in doua moduri folosind ambele formule.

Functia va fi:

```
# aceleasi ca precedentele doar ca avem in loc
# de functia identica functia g
EApplyFunc<- function(VA,g)
{
```

```
  return(integrate(Vectorize(function(x)VA@densitate(x)*g(x)),
  -Inf, Inf)$value)
}
VarApplyFunc<- function(VA,g)
{
  return(integrate(Vectorize(function(x)VA@densitate(x)*
  (x-EApplyFunc(VA))^2, -Inf, Inf)$value )
}
```

8 Cerinta 7: Functia P

Cerinta: Crearea unei functii P care permite calculul diferitelor tipuri de probabilitati asociate unei variabile aleatoare continue(similar functiei P din pachetul discreteRV)

Pentru a rezolva cerinta vom face o functie ce primeste ca parametru un CRV si 2 intervale, primul va fi cel pe care este facuta valoarea aleatoare iar cel de-al 2-lea cel la care conditionam.

```
#P(lowerbound <=VA<=upperbound|lowerCondition <=VA<=upperCondition)
P = function (VA,lowerbound,upperbound,lowerCondition,upperCondition)
{
  # Daca intervalul conditionalului si cel pe care vrem sa facem
  # densitatea sunt disjuncte atunci returnam 0
  if (upperbound < lowerCondition || lowerbound > upperCondition)
    return (0)

  # Gasim intersectia intervalelor
  lb <- max(lowerbound, lowerCondition)
  ub <- min(upperbound, upperCondition)


  # Gasim probabilitatea avand in vedere conditiile puse
  i1 <- integrate (VA@densitate, lb, ub)$value

  # Calculam posibilitatea sa cadem in conditiile puse
```

```
  i2 <- integrate (VA@densitate, lowerCondition, upperCondition)$value

  if (i2 != 0)
    return (i1 / i2)

  # Daca nu avem cazuri in care se intampla conditia returnam 0
  return(0)
}
```

9 Cerinta 8: Afisarea unei "fise de sinteza"

Enunt: Afisarea unei "fise de sinteza" care sa conțină 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.

Functia noastra este urmatoarea:

```
hipergeometrica <- c("Definitie:Variabila aleatoare X urmeaza legea
hipergeometrica cu parametrii a, b si n (a,b,n apartin lui N*, n <=a+b )
daca poate lua orice valoare intreaga intre max(0,n - b)si min(n,a) si
P(X=k)=C(k,a)*C(n-k,b)/C(n,a+b)
oricare ar fi k de la acel max la acel min.
(C e notatia pentru combinari luate de la pana la)",
             "Utilizare: Testul provenit din distributia
             hipergeometrica poate fi folosit pt
             determinarea daca o populatie e slab reprezentata",
             "Notatie:",
             "Media: E(X)=ap unde p=a/(a+b),q=1-p",
             "Variatia: Var(X) = npq*(a+b-n)/(a+b-1)",
             "Sursa: http://dep2.mathem.pub.ro/pdf/didactice/
             Probabilitati%20si%20statistica.pdf")
```

```
uniforma <- c("Definitie: O v.a. U este repartizata uniform
pe intervalul (a,b)
daca admite densitatea
de repartitie f(x)=1/(b-1), x in (a,b).",
              "Utilizare: Se foloseste atunci
              cand numarul cazurilor posibile tinde la infinit",
              "Notatie: U ~ U((a,b)),
              unde (a,b) este intervalul pe care este
              repartizata variabila aleatoare",
              "Media: E[U] = integrala
              de la -Inf la Inf din x*f(x) dx = (a+b)/2",
              "Variatia: Var(U) = E[U^2]-E[U]^2 = ((b-a)^2)/12",
              "Sursa: http://cs.unitbv.ro/
              ~pascu/stat/Distributii%20continue%20clasice.pdf")

exponentiala <- c("Definitie: O v.a. X este repartizata
exponential de parametru lambda > 0 daca
densitatea de repartitie a lui X este de
forma f(x) = lambda * e^(-lambda*x), x > 0",
              "Utilizare: Modeleaza
              timpul de asteptare pana la aparitia unui eveniment de interes.",
              "Notatie: X ~ Exp(lambda)",
              "Media: E[X] = integrala
              de la -Inf la Inf din x*f(x) dx = 1/lambda",
              "Variatia:
              Var(X) = E[X^2] - (E[x])^2 = 1/(lambda^2)",
              "Sursa: Curs 10,
              Probabilitati si Statistica, Prof. Alexandru Amarioarei")

normala <- c("Definitie: Spunem ca o v.a.
X este repartizata normal de parametri miu si sigma^2
daca admite densitatea
f(x) = (1/(sqrt(2pi)*sigma))*e^(-(x-miu)^2/2*sigma^2),
unde x in R)",
             "Notatie: X ~ N(miu,sigma^2)",
             "Media: E[X] = miu",
             "Variatia: Var(U) = sigma^2",
```

```
            "Sursa: Curs 10,
            Probabilitati si Statistica, Prof. Alexandru Amarioarei")


# Functie ajutatoare care afiseaza chestiile cerute din
repartitia pe care e chemata
sinteza <- function(rep){
  for (i in rep)
  {
    print(i)
  }

}
```

10 Cerinta 9: Generare

Enunt: Generarea a n valori(unde n este precizat de utilizator!) dintr-o repartiție de variabile aleatoare continue.

Cum procedam?

Vom avea o functie care primeste ca parametri n(numarul cerut de valori) si F(o functie de repartitie). Functia de generare calculeaza aceste valori dupa algoritmul de generare din curs. Se calculeaza F_1 functia inversa a lui F, dupa care, pana sunt n valori in lista pe care o returnam, generam o variabila uniforma intre 0 si 1 pe care calculam F_1. Punem rezultatul intr-o lista si o returnam. Functia trebuie sa primeasca neaparat o functie explicita care se poate calcula.

```
# Generarea a n valori(unde n este precizat de utilizator!)
# dintr-o repartiție de variabile aleatoare continue

generation_n_numbers <- function(n, VA)
{
  #creem o lista vida
  result <- list()
```

```
  #F_1 functie inversa
  F_1 <- inverse(VA@densitate, -Inf, Inf)

  #cat timp nu avem n numere
  while(length(result) != n)
  {
    #se genereaza U variabila uniforma pe [0,1] - si 1-U e in [0,1]
    U <- runif(1, 0, 1)
    #se calculeaza functia inversa in punctul U
    X <- F_1(U)

    #adaugam pe U la rezultat
    result <- append(result, X)
  }

  #intoarcem rezultatul
  return(result)
}
```

11 Cerinta 10: Covarianta si Coeficientul de corelatie

Cerinta: Calculul covarianței și coeficientului de corelație pentru două variabile aleatoare continue(Atenție:Trebuie să folosiți densitatea comună a celor două variabile aleatoare!)

Teorema: Daca X si Y au pdf comuna $f(x, y)$ pe domeniul $[a, b][c, d]$, atunci:

$$Cov(X, Y) = \int_c^d \int_a^b (x - \mu_x)(y - \mu_y) f(x, y) dx dy$$

Definitie: Daca X si Y au pdf comuna $f(x, y)$ pe domeniul $[a, b][c, d]$, atunci coeficientul de corelatie este:

$$Cor(X, Y) = \frac{Cov(X, Y)}{\sqrt{Var(X)Var(Y)}}$$

Pentru rezolvarea cerintelor primul pas este acela de a extrage functiile densitate de probabilitate marginale, apoi cu ajutorul acestora se calculeaza mediile celor doua variabile si se integreaza dublu ca si in teorema de mai sus. Pentru calculul coeficientului de corelatie vom folosi una din definitiile din cerinta 6, pentru calculul variantei celor doua variabile continue, iar apoi se aplica formula prezentata mai sus.

antetul functiilor este urmatorul: Covariance ¡- function (VA) Corelation ¡- function (VA)

Functia va fi:

```
#fct auxiliara pt integrarea dubla
dubluintegrat <- function (f) {
  return (
    integrate(Vectorize(function (y) {
      integrate(Vectorize(function (x) { f(x, y) }), -Inf,Inf)$value
    }), -Inf,Inf)$value
  )
}


Covariance <- function(VA) {
  if (!VA@is2Dimensional) {
    stop("Variabila nu este bidimensionala")
  }
  else {
    EX <- E(CRV(Vectorize(margineX(VA))))
    EY <- E(CRV(Vectorize(margineY(VA))))
    #functia de integrat din care rezulta covarianta
    funcint <- function(x, y) {
    return ((x - EX) * (y - EY) * VA@densitate(x, y)) }
    return(dubluIntegrat(funcint))
  }
}

Corelation <- function(VA)
{
```

```
  if (!VA@is2Dimensional) {
    stop("Variabila nu este bidimensionala")

  }
  else {
    #Formula pentru covarianta
    X <- CRV(Vectorize(margineX(VA)))
    Y <- CRV(Vectorize(margineY(VA)))
    return (Covariance(VA) / sqrt(Var(X) * Var(Y)))

  }
}
```

12 Cerinta 11: Densitatea marginala/conditionata

Cerinta: Pornind de la densitatea comună a două variabile aleatoare continue, sa se realizeze construirea densităților marginale și a densităților condiționate.

Teorema: Fie X si Y doua variabile aleatoare continue, avand densitatea comuna continua $f(x, y)$ cu domeniul $[a, b] \times [c, d]$. Pdf-urile marginale sunt:

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

Pentru a calcula densitatile conditionate, vom spune ca vrem sa calculam densitatea lui X conditionata de Y care trebuie sa fie in intervalul $[c, d] \subseteq [c, d]$; si densitatea lui Y conditionata de X care trebuie sa fie in intervalul $[a, b] \subseteq [a, b]$ (daca intervalul conditionarii nu este inclus in domeniu, putem lua intersectia dintre interval si intervalul domeniului, avand acelasi rezultat). Astfel, densitatile conditionate devin:

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

Pentru a rezolva cerinta, am implementat doua functii, prima pentru a calcula densitatea marginala/conditionata pentru variabila X (*densitate*

$MarginalaX$), prima componenta a functiei; iar cealalta pentru a calcula densitatea marginala/conditionata pentru variabila Y ($densitateMarginalaY$), a doua componenta a functiei.

antetul functiilor este urmatorul: margineX ¡- function(VA,lowerCond = -Inf, upperCond = Inf) margineY ¡- function(VA,lowerCond = -Inf, upperCond = Inf)

Functiile vor fi:

```
#functie care returneaza marginala la X pt o bidimensionala
margineX <- function(VA,lowerCond = -Inf, upperCond = Inf) {
  tryCatch (
    {
      #evident 0 :)))
      if (lowerCond > upperCond) 0
      #formula pentru gasirea marginii
      else {  function(x) (integrate(Vectorize(function(y)
      (VA@densitate(x, y))), lowerCond, upperCond)$value)}
    },
    error = function(e)
    {
      stop("Functia de densitate comuna nu este valida!")
    })
}

#functie care returneaza marginala la Y pt o bidimensionala
margineY <- function(VA,lowerCond = -Inf, upperCond = Inf) {
  tryCatch (
    {
      #evident 0 :)))
      if (lowerCond > upperCond) 0
      #formula pentru gasirea marginii
      else function(y) (integrate(Vectorize(function(x)
      (VA@densitate(x, y))), lowerCond, upperCond)$value)
    },
    error = function(e)
```

```
    {
      stop("Functia de densitate comuna nu este valida!")
    })
}
```

13 Cerinta 12: Convolutie

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

Cum procedam?

Vom avea o functie care primeste ca parametri 2 valori reale x si y(punctele in care se calculeaza suma si diferenta) si cele 2 functii de densitate. Calculam suma si diferenta cu ajutorul integralelor de mai jos luate din teorema de la curs. Functia intoarce o lista in care pe prima pozitie se afla valoarea sumei si pe a doua pozitie se afla valoarea diferentei.
Pentru suma:

$$f_Z(U) = \int_{-\infty}^{\infty} f_X(x) * f_Y(U - x)dx$$

Pentru diferenta:

$$f_W(U) = \int_{-\infty}^{\infty} f_X(x) * f_Y(x - U)dx$$

```
    # Construirea sumei și diferenței
    # a două variabile aleatoare continue independente

sum_dif <- function(x, y, VA1, VA2)
{

  f1 <- VA1@densitate
  f2 <- VA2@densitate
  #calculam valoarea in care trebuie calculata integrala
  U <- x + y

  #integrala suma
  suma <- integrate(function(U){
    f1(x)*f2(U-x)
  }, -Inf, Inf)$value
```

```
  #calculam valoarea in care trebuie calculata integrala x2
  U <- x - y

  #integrala diferenta
  diferenta <- integrate(function(U){
    f1(x)*f2(x-U)
  }, -Inf, Inf)$value

  #punem valorile intr-o lista pentru a le returna
  result = list(suma, diferenta)
  return(result)
}
```