

# Proiect

# Statistica 2025

Lider:Fîrtală Maria-Doina-grupa  
311

Scurtu Jasmin-Ana-Maria-grupa  
311

Tăranu Daria Cristiana-grupa 321

Lungu Anne-Marie-grupa 321

**Profesor coordonator: Simona Cojocea**

# Cuprins

<b>PROBLEMA 1:</b>	<b>3</b>
<b>PROBLEMA 3</b>	<b>14</b>
<b>PROBLEMA 2:</b>	<b>35</b>

## PROBLEMA 1:

1. Construiți funcții în R care să implementeze algoritmi de simulare pentru următoarele situații descrise:

a)  $X \sim \text{Cauchy}(x_0, \gamma)$

$$F(x) = \frac{1}{2} + \frac{1}{\pi} \arctg\left(\frac{x - x_0}{\gamma}\right) \Rightarrow$$

$$F(x) = u$$

$$\Rightarrow \frac{1}{2} + \frac{1}{\pi} \arctg\left(\frac{x - x_0}{\gamma}\right) = u \Leftrightarrow$$

$$\arctg\left(\frac{x - x_0}{\gamma}\right) = \pi(u - \frac{1}{2})$$

$$\frac{x - x_0}{\gamma} = \operatorname{tg}\left(u - \frac{1}{2}\right)$$

$$x - x_0 = \gamma \operatorname{tg}\left(u - \frac{1}{2}\right)$$

$$x = x_0 + \gamma \operatorname{tg}\left(u - \frac{1}{2}\right)$$

$$\Rightarrow X = x_0 + \gamma \operatorname{tg}\left(u - \frac{1}{2}\right)$$

Mai sus am folosit metoda inversei pentru a afla  $f$ -ul pe care il vom folosi în cod:

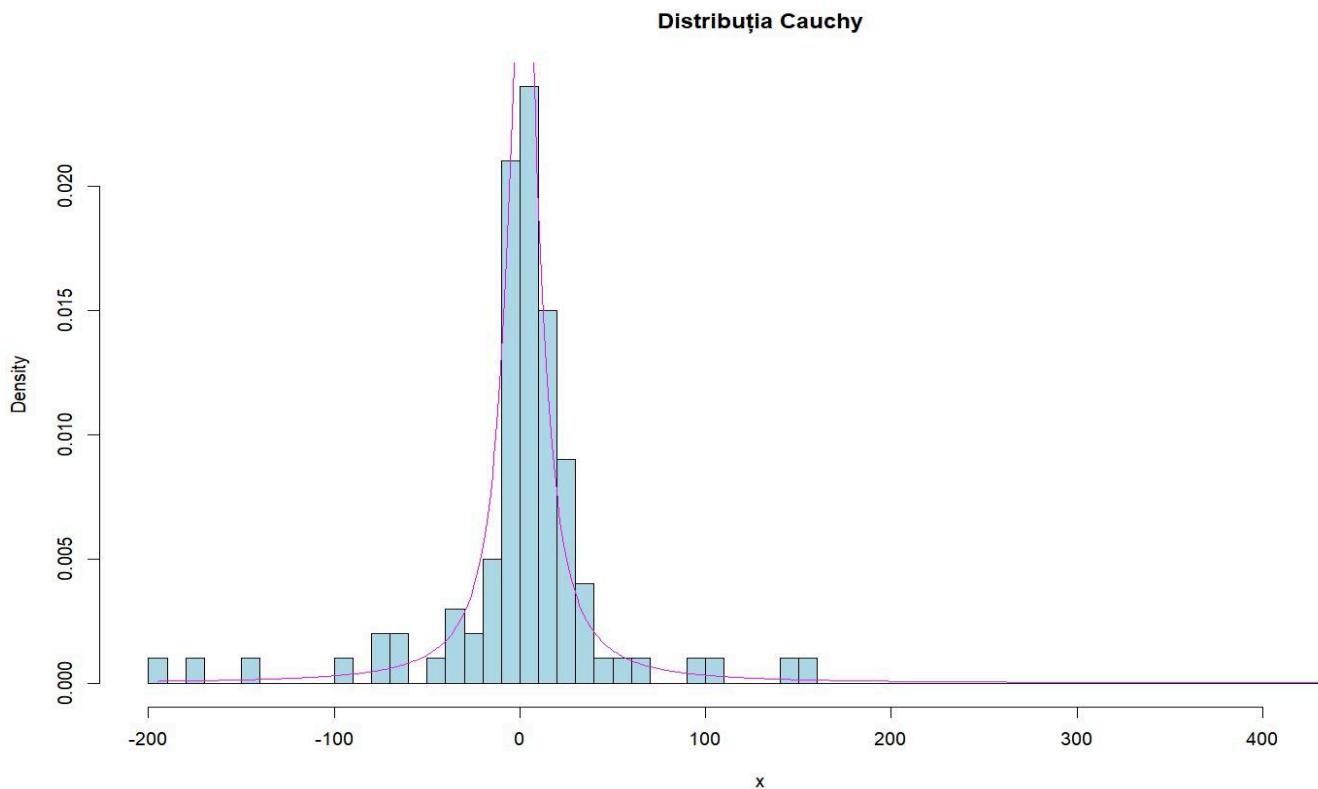
```
# Ex 1
# a) -----
set.seed(123)
n <- 10
x0 <- 2
gamma <- 1

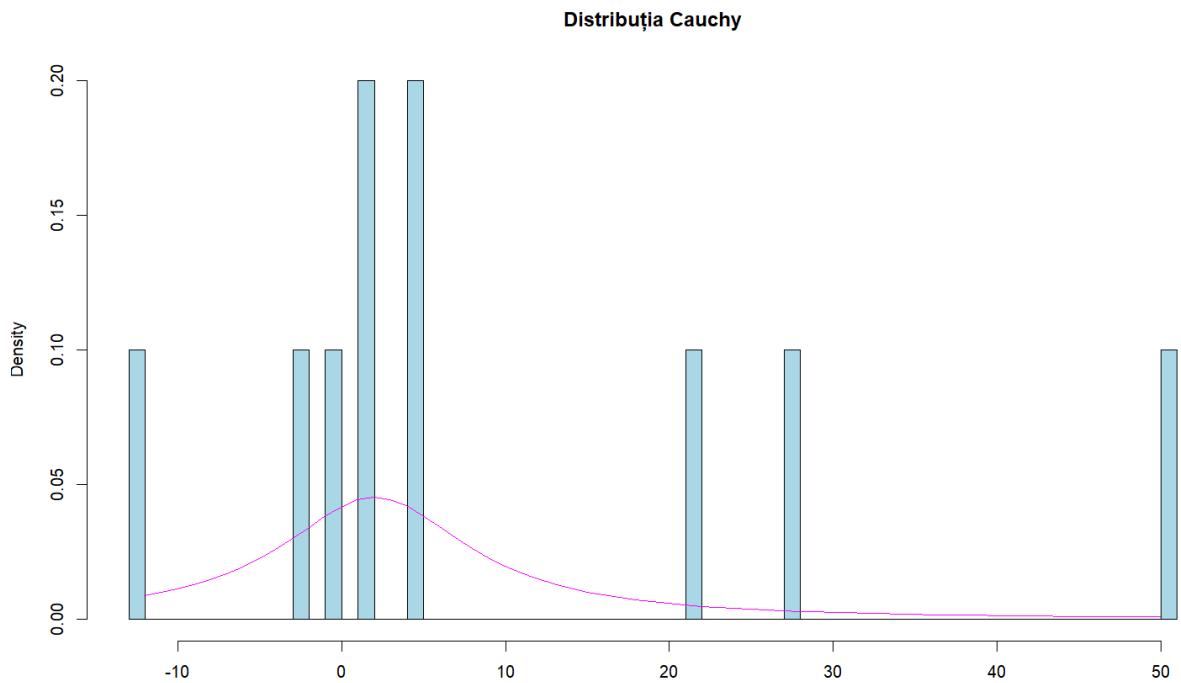
#Folosim functia de densitate pentru a genera functia Cauchy centrata in x0,
#avand abaterea medie absoluta data de gamma
f_Cauchy <- function(x,x0, gamma){
  1/(pi*gamma) * 1/(1+((x-x0)/gamma)^2)
}

#F_Cauchy am generat-o folosind metoda inversei
F_Cauchy <- function(n, x0, gamma){
  U <- runif(n)
  gamma*tan(pi*(U - 1/2)) + x0
}

#X ia valorile generate de functia cuantila
x <- F_Cauchy(n, x0, gamma)

#generam histograma
hist(x, breaks = 50, probability = TRUE, main = "Distribuția Cauchy", col = "lightblue")
t <- seq(min(x),max(x),1)
#creem graficul functiei f care urmeaza repartitia Cauchy
lines(t,f(t, x0, gamma),col="magenta",xlim=c(-10,10))
```





Observam ca atunci cand Gamma creste valorile departate de centru( $x_0$ ) vor avea o probabilitate mai mare de aparitie. (pentru un numar mai mic de simulari!)

b)  $X \sim \begin{pmatrix} 1 & 2 & 3 \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{6} \end{pmatrix}$

$$X \sim \begin{pmatrix} 1 & 2 & 3 \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{6} \end{pmatrix}$$

Functie de masă:  $f(x) = \begin{cases} \frac{1}{2}, & x=1 \\ \frac{1}{3}, & x=2 \\ \frac{1}{6}, & x=3 \\ 0, & \text{altfel} \end{cases}$

Functie de repartitie:  $F(x) = \begin{cases} 0, & x \in (-\infty, 1) \\ \frac{1}{2}, & x \in [1, 2) \\ \frac{1}{2} + \frac{1}{3} = \frac{5}{6}, & x \in [2, 3) \\ 1, & x \in [3, \infty) \end{cases}$

```

# b) ----

set.seed(123)
n <- 1000

# Generam n valori uniforme intre 0 si 1
U <- runif(n)

# Aplicam metoda transformarii inverse
X_discret <- ifelse(U < 1/2, 1, ifelse(U < 5/6, 2, 3))

# Creare histograma pentru distributia discreta
hist(X_discret, breaks = seq(0.5, 3.5, 1),
      main = "Distributia discreta a lui X",
      col = "lightblue", border = "black")

```

c)  $X \sim \begin{pmatrix} x_1 & x_2 \\ 1-p & p \end{pmatrix}$  unde  $x_1, x_2, p$  sunt transmiși funcției ca parametru

Observam ca  $X \sim \text{Bern}(p)$

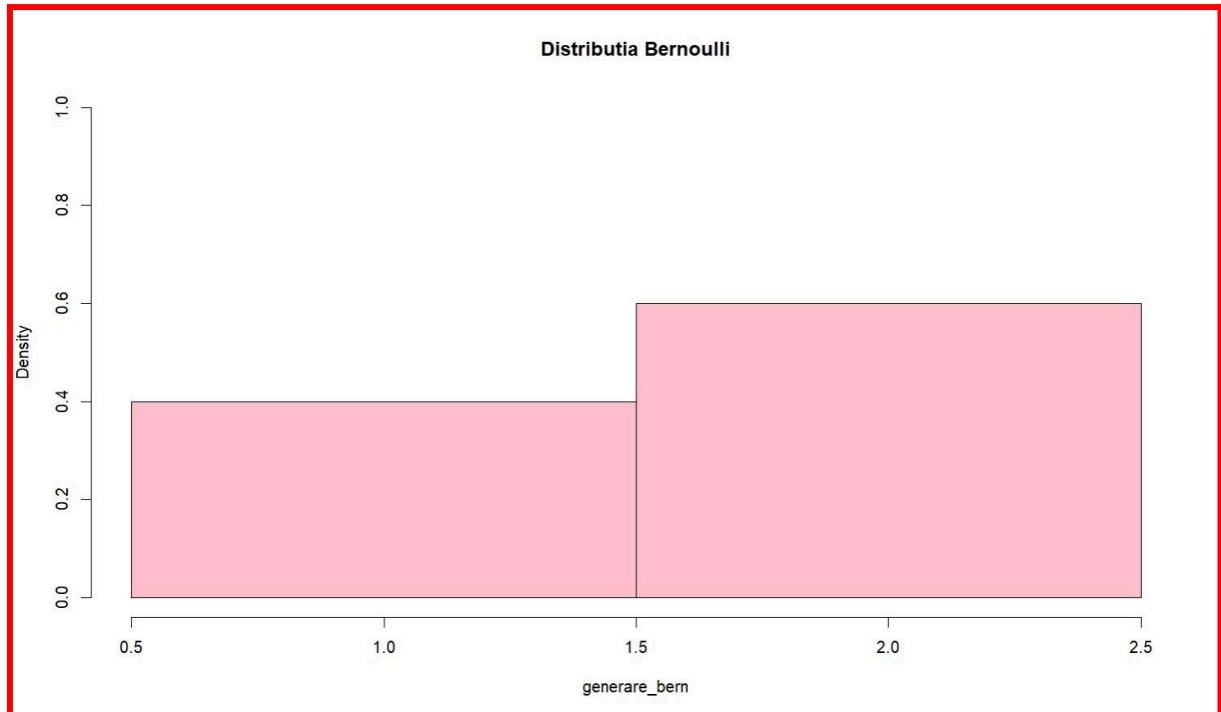
```

# c) ----
n <- 10
# Functie Bernoulli
bernoulli <- function(x1, x2, p, n) {
  generare_bern <- ifelse(runif(n) < p, x2, x1)
  # Creare histograma pentru distributia Bernoulli
  hist(generare_bern, breaks = seq(min(x1, x2) - 0.5, max(x1, x2) + 0.5, 1),
        probability = TRUE, main = "Distributia Bernoulli", col = "pink",
        border = "black", ylim = c(0, 1))
  return(generare_bern)
}

# Citirea valorilor pentru Bernoulli
test_x1 <- 1
test_x2 <- 2
# p-ul ia valori intre 0 si 1
test_p <- 0.7

# Generare si vizualizare Bernoulli
bernoulli(test_x1, test_x2, test_p, n)

```



d) extragerea unui număr întreg cuprins între a și b într-o manieră ***perturbată aleator*** față de repartiția uniformă(*de exemplu*, se extrage ora curentă a sistemului, se face o prelucrare asupra acesteia pentru a obține un număr natural x care va reprezenta poziția din sirul numerelor naturale cuprinse între a și b căreia i se va asocia o probabilitate mai mare de apariție)

În continuare, veți aplica acești algoritmi de simulare pentru a rezolva următoarea problemă:

Despre ***linia 501 STB*** se cunosc:

-*în medie* călătoresc y călători pe zi (medie raportată la datele dintr-o luna pentru un singur tramvai), unde y se obține apelând funcția de la d)

-*numărul minim* înregistrat în perioada studiată este de **x<sub>min</sub>** iar *numărul maxim* este **x<sub>max</sub>** călători pe zi (valori ce se obțin folosind funcția de la d) )

-zilele dintr-o lună sunt clasificate ca fiind **lejere**(mai puțin de **350 de călători pe zi**), **normale**(între **351 și 670 călători pe zi**), **aglomerate**(peste **671 călători pe zi**). Folosiți funcția de la b) pentru a simula câte zile lejere, normale și respectiv aglomerate are fiecare lună în parte.

-*prețul* unui bilet este **3 lei** și, *în medie*, **x%** din pasagerii care nu au abonament plătesc biletul la utilizarea tramvaiului, unde x este un număr între 1 și 99 generat astfel: cu algoritmul de la a) se generează un număr v căruia i se extrage partea fracționară și i se atribuie parametrului p de la c), iar x<sub>1</sub> și respectiv x<sub>2</sub> sunt extrase uniform din mulțimea numerelor naturale de la 1 la 99. Valoarea medie va fi rezultatul simulării de la c) în condițiile descrise.

-*prețul* unui abonament este **70 lei pe lună** și, *în medie*, **x%** din pasagerii care călătoresc cu tramvaiul îl achiziționează, unde x este un număr între 1 și 99 generat cu algoritmul de la d) folosind ca element perturbator un număr z obținut ca modulul părții întregi a unui număr obținut prin apelul funcției de la a) cu parametrii 5 și 2.

# d) -----

```
n <- 10
# Functie pentru generarea unui numar intreg intre a si b
perturbare_aleat <- function(a, b) {
  secunda <- as.numeric(format(sys.time(), "%S")) # Extrage secunda curentă
  perturbare <- sin(secunda) * (b - a)/15 # vrem unif(0,1)
  # Generam un numar aleator intre a si b si aplicam perturbarea (i.e esantionul)
  val_aleat <- sample(a:b, 1)
  val_perturbata <- round(perturbare + val_aleat)
  # ne asiguram rezultatul ramane in intervalul [a, b]
  return(max(min(val_perturbata, b), a))
}
```

- a) Generați, prin simulare, valori care să reprezinte numărul de călători dintr-o zi, pentru fiecare zi a lunii decembrie 2024, respectând restricțiile de mai sus și stocați valorile obținute într-un vector. Construiți histograma acestor valori.

Avem funcția `simulare_pasageri` care se folosește de funcția creată anterior `perturbare_aleat` pentru a genera un număr oarecare de călători. Astfel, se reduce variația extrema rezultatele fiind mai puțin disperse.

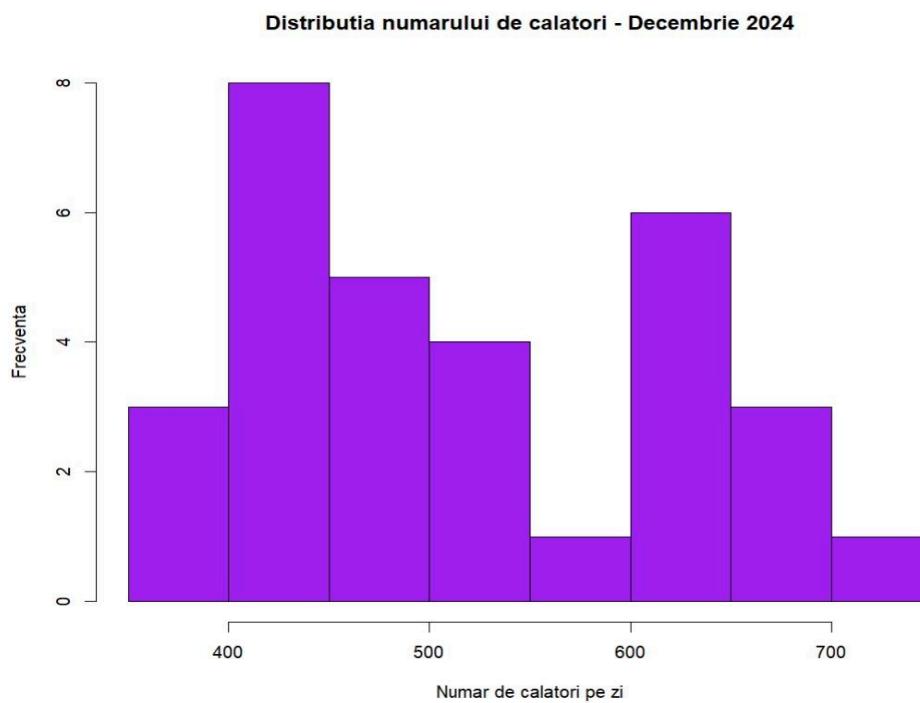
```
#a)
# Functie pentru simularea numarului de calatori
simulare_pasageri <- function(xmin, xmax, y) {
  pasageri <- perturbare_aleat(xmin, xmax)
  # Creste probabilitatea de a alege un numar apropiat de medie
  pasageri <- round((pasageri + y) / 2) # Ajustare spre medie
  return(pasageri)
}

# Setam parametrii pentru linia 501 STB
xmin <- perturbare_aleat(150, 300) # Numarul minim
xmax <- perturbare_aleat(700, 1000) # Numarul maxim
y <- perturbare_aleat(300, 700) # Valoarea medie aproximativa

# Generam numarul de calatori pentru fiecare zi din decembrie 2024
decembrie_2024 <- sapply(1:31, function(x) simulare_pasageri(xmin, xmax, y))

# Cream histograma folosind hist()
hist(decembrie_2024,
  breaks = 10,
  col = "purple",
  border = "black",
  main = "Distributia numarului de calatori - Decembrie 2024",
  xlab = "Numar de calatori pe zi",
  ylab = "Frecventa")
```

In `decembrie_2024` am stocat valorile obținute.



- b) Repetați procedul de la a) pentru fiecare luna a unui 2024 și centralizați rezultatele empirice într-un dataframe care să conțină, pentru fiecare lună valorile medii, minime și maxime de călători, precum și procentul de zile lejere, normale și respectiv aglomerate înregistrate.

Simuleaza\_luna este o funcție prin care generăm numărul călătorilor pe fiecare zi din luna, folosindu-ne de funcția simulare\_pasageri care va alege un număr aleatoriu între minimul și maximul de călători, dar cu o ușoară perturbare bazată pe secundele curente ale ceasului.

**Apoi ajustează spre medie:** round((pasageri + y) / 2), ceea ce înseamnă că ia valoarea generată și o "trage" spre y, care este valoarea medie aproximativă.

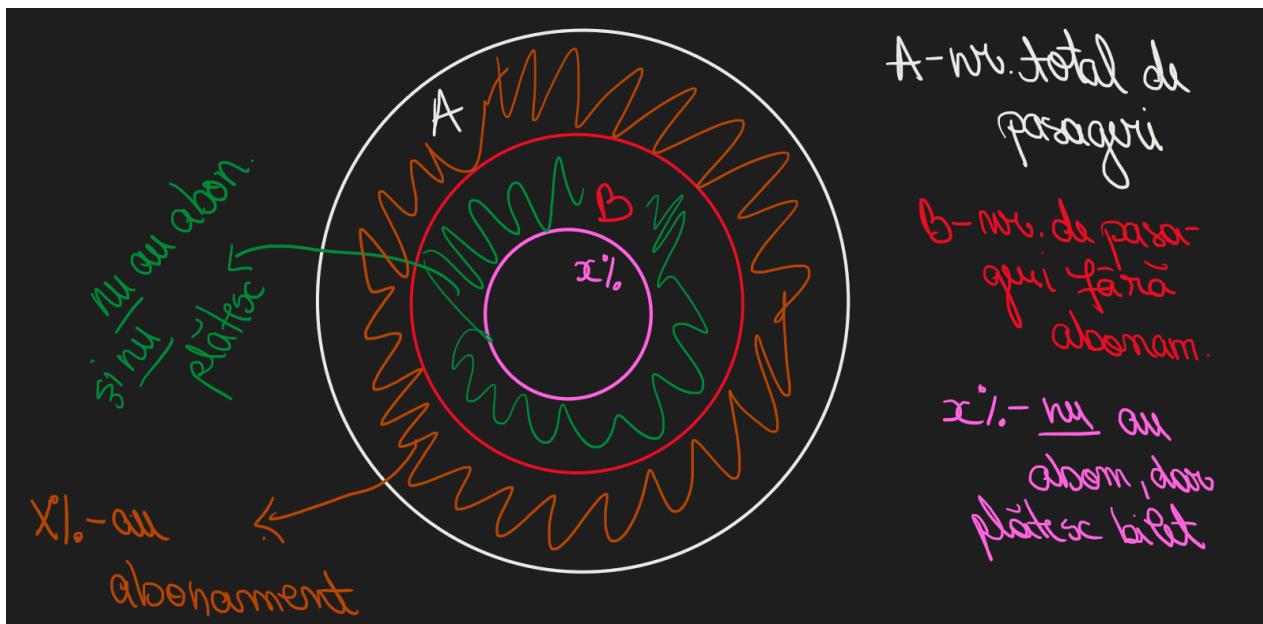
Aflăm câte zile usoare, normale și aglomerate avem și mai cream dataframe-ul

```
# b) -----
# Functie pentru simularea fiecarii lunii din anul 2024
simuleaza_luna <- function(luna, zile) {
  min_calatori <- perturbare_aleat(150, 300) # Numarul minim
  max_calatori <- perturbare_aleat(700, 1000) # Numarul maxim
  medie_calatori <- perturbare_aleat(300, 700) # Valoarea medie aproximativa
  calatori_zi <- numeric(zile) # Initializam vectorul
  for (i in 1:zile) {
    calatori_zi[i] <- simulare_pasageri(min_calatori, max_calatori, medie_calatori) # Generam datele zilnice
  }
  zile_usoare <- sum(calatori_zi < 350)
  zile_normale <- sum(calatori_zi >= 351 & calatori_zi <= 670)
  zile_aglomerate <- sum(calatori_zi > 671)
  return(data.frame(
    Luna = luna,
    Medie_Calatori = mean(calatori_zi),
    Min_Calatori = min(calatori_zi),
    Max_Calatori = max(calatori_zi),
    Proc_Zile_Usoare = zile_usoare / zile * 100,
    Proc_Zile_Normale = zile_normale / zile * 100,
    Proc_Zile_Aglomerate = zile_aglomerate / zile * 100)))
}

# Definim numarul de zile pentru fiecare luna
luni_2024 <- list(
  "Ianuarie" = 31, "Februarie" = 29, "Martie" = 31, "Aprilie" = 30,
  "Mai" = 31, "Iunie" = 30, "Iulie" = 31, "August" = 31,
  "Septembrie" = 30, "Octombrie" = 31, "Noiembrie" = 30, "Decembrie" = 31
)
# Simulam datele pentru fiecare luna
rezultate_2024 <- data.frame() # Initializam un dataframe gol
for (luna in names(luni_2024)) {
  rezultate_2024 <- rbind(rezultate_2024, simuleaza_luna(luna, luni_2024[[luna]]))
}
# Afisam rezultatele
print(rezultate_2024)

  Luna Medie_Calatori Min_Calatori Max_Calatori Proc_Zile_Usoare
 1  Ianuarie        486.5161      182        750     25.80645
 2  Februarie       418.8621      220        603     34.48276
 3   Martie         519.7097      194        895     29.03226
 4   Aprilie        496.0667      290        711     16.66667
 5     Mai          437.0645      184        785     38.70968
 6   Iunie          451.1000      239        687     33.33333
 7   Iulie          496.2903      275        659     16.12903
 8   August          463.1290      237        740     32.25806
 9 Septembrie        514.8000      272        745     20.00000
10 Octombrie        488.3548      287        772     22.58065
11 Noiembrie        475.1667      274        774     30.00000
12 Decembrie        434.4516      150        741     41.93548
  Proc_Zile_Normale Proc_Zile_Aglomerate
 1           48.38710          25.806452
 2           65.51724          0.000000
 3           38.70968          29.032258
 4           70.00000          13.333333
 5           48.38710          12.903226
 6           60.00000          6.666667
 7           83.87097          0.000000
 8           54.83871          12.903226
 9           53.33333          26.666667
10          58.06452          19.354839
11          50.00000          20.000000
12          38.70968          16.129032
```

- c) Completăți dataframe-ul de la b) cu simularea nr de pasageri cu abonament, nr de pasageri care platesc bilet și respectiv număr de pasageri care nu plătesc bilet. Determinați pentru fiecare lună în parte veniturile provenite din bilete și abonamente și respectiv, veniturile nerealizate prin neplata biletului de unii dintre pasageri. Organizați informația într-o manieră ușor de vizualizat.



Cream functia simuleaza\_pasageri\_detaliat in care adaugam in plus pretul biletelor si al abonamentelor. Construim vectorii necesari pentru a stoca datele esentiale. Populam totalul de calatori si generam procentul de oameni care au abonament si ii aflam pe cei ramasi. Din cei ramasi aflam oamenii care cumpara bilet, iar restul sunt cei care nu au nici abonament, nu cumpara nici bilet.

```
# c)
# Adaugam simularea pentru pasagerii cu abonament, cei care platesc bilet si cei care nu platesc
simuleaza_pasageri_detaliat <- function(luna, zile) {
  min_calatori <- perturbare_aleat(150, 300)
  max_calatori <- perturbare_aleat(700, 1000)
  medie_calatori <- perturbare_aleat(300, 700)
  pret_bilet <- 3 # Preț bilet
  pret_abonament <- 70 # Preț abonament

  total_calatori <- numeric(zile)
  pasageri_abonament <- numeric(zile)
  pasageri_bilet <- numeric(zile)
  pasageri_fara_plata <- numeric(zile)
  venit_bilete <- numeric(zile)
  venit_abonamente <- numeric(zile)
  pierderi_neplata <- numeric(zile)

  for (i in 1:zile) {
    total_calatori[i] <- simulare_pasageri(min_calatori, max_calatori, medie_calatori)

    # Procent pasageri cu abonament intre 1% si 99%
    x_abonament <- perturbare_aleat(1, 99) / 100
    pasageri_abonament[i] <- round(total_calatori[i] * x_abonament)

    # Pasagerii ramasi
    ramasi <- total_calatori[i] - pasageri_abonament[i]

    # Procent pasageri care platesc bilet intre 1% si 99%
    x_bilet <- perturbare_aleat(1, 99) / 100
    pasageri_bilet[i] <- round(ramasi * x_bilet)
    pasageri_fara_plata[i] <- ramasi - pasageri_bilet[i]
  }
}
```

Calculam veniturile/pierderile si ne cream dataframe ul.

```
# Calcul venituri
venit_bilete[i] <- pasageri_bilet[i] * pret_bilet
venit_abonamente[i] <- pasageri_abonament[i] * pret_abonament
pierderi_neplata[i] <- pasageri_fara_plata[i] * pret_bilet
}

zile_usoare <- sum(total_calatori < 350)
zile_normale <- sum(total_calatori >= 351 & total_calatori <= 670)
zile_aglomerate <- sum(total_calatori > 671)

return(data.frame(
  Luna = luna,
  Medie_Calatori = mean(total_calatori),
  Min_Calatori = min(total_calatori),
  Max_Calatori = max(total_calatori),
  Proc_Zile_Usoare = zile_usoare / zile * 100,
  Proc_Zile_Normale = zile_normale / zile * 100,
  Proc_Zile_Aglomerate = zile_aglomerate / zile * 100,
  Pasageri_Abonamente = mean(pasageri_abonament), # facem media pt ca vrem pe luna
  Pasageri_Bilete = mean(pasageri_bilet),
  Pasageri_Fara_Plata = mean(pasageri_fara_plata),
  Venit_Bilete = sum(venit_bilete),
  Venit_Abonamente = sum(venit_abonamente),
  Pierderi_Neplata = sum(pierderi_neplata)))
}
```

Simulam valorile pentru fiecare luna si afisam

```
# Simulam pentru fiecare luna
rezultate_2024_detaliat <- data.frame()
for (luna in names(luni_2024)) {
  rezultate_2024_detaliat <- rbind(rezultate_2024_detaliat, simuleaza_pasageri_detaliat(luna, luni_2024[[luna]]))

# Afisam rezultatele
print(rezultate_2024_detaliat)
```

Vor fi afisate urmatoarele:

	Proc_Zile_Normale	Proc_Zile_Aglomerate	Pasageri_Abonamente	Pasageri_Bilete
0	54.83871	0.000000	193.5484	65.09677
1	37.93103	20.689655	186.9310	129.31034
2	90.32258	0.000000	191.0645	115.83871
3	66.66667	0.000000	195.0000	133.73333
4	45.16129	6.451613	155.3871	123.03226
5	36.66667	36.666667	241.1333	83.03333
6	64.51613	0.000000	161.0968	110.03226
7	41.93548	25.806452	231.7097	78.25806
8	56.66667	13.333333	196.0333	117.00000
9	38.70968	45.161290	256.8065	161.25806
10	50.00000	20.000000	239.5000	121.76667
11	54.83871	16.129032	218.6129	115.54839
	Pasageri_Fara_Plata	Venit_Bilete	Venit_Abonamente	Pierderi_Neplata
0	133.3548	6054	420000	12402
1	128.8966	11250	379470	11214
2	147.0645	10773	414610	13677
3	113.7333	12036	409500	10236
4	126.4839	11442	337190	11763
5	202.7333	7473	506380	18246
6	130.9677	10233	349580	12180
7	162.0645	7278	502810	15072
8	154.2667	10530	411670	13884
9	201.4839	14997	557270	18738
10	127.1667	10959	502950	11445
11	158.5161	10746	474390	14742

3. **Inegalitatea Berry-Essen** este un rezultat celebru cu ajutorul căruia putem determina acuratețea aproximării pe care o realizează TLC. Cerând în plus față de condițiile din TLC ca  $E|X_i|^3 < \infty$ , avem că:

$$\sup_x |P(Z_n \leq x) - \Phi(x)| \leq \frac{33}{4} \frac{E|X_1 - \mu|^3}{\sqrt{n}\sigma^3},$$

### PROBLEMA 3

unde  $X_1, X_2 \dots X_n$  i.i.d.,  $\Phi(x)$  este funcția de repartiție a normalei standard,  $\mu = E(X_1)$

$$\sigma = \sqrt{\text{Var}(X_1)} \text{ și } Z_n = \frac{\sqrt{n}(\bar{X}_n - \mu)}{\sigma}.$$

Cerinte:

- 1) Calculați  $P(Z_n \leq x)$  pentru repartițiile:

binomială ( $n, p$ )

$n \rightarrow$  nr. de încercări

$p \rightarrow$  probabilitatea de succes

$$\mu = n \cdot p$$

$$\sigma^2 = np(1-p)$$

Pentru o v.a. binomială, putem defini variabila standardizată  $Z_n$

$$Z_n = \frac{\sqrt{n}(\bar{X}_n - \mu)}{\sigma}$$

$$\begin{aligned} P(Z_n \leq x) &= P\left(\frac{\sqrt{n}(\bar{X}_n - \mu)}{\sigma} \leq x\right) = \\ &= P\left(\sqrt{n}(\bar{X}_n - \mu) \leq \sigma \cdot x\right) = \\ &= P\left(\bar{X}_n \leq \frac{\sigma \cdot x}{\sqrt{n}} + \mu\right) \end{aligned}$$

$$\bar{X}_n = \frac{1}{n} \sum_{i=1}^n X_i$$

$$\begin{aligned} X_1, X_2, \dots, X_n &\text{ iid } \sim \text{Bin}(n, p) \\ X = \sum_{i=1}^n X_i &\sim \text{Bin}(n, p) \end{aligned}$$

L

→

```

#Problema 3
# Exercitiul 3
#a)-----

#Binomiala

p_binomial <- function(n, p, x) {
  miu <- n * p #se calculeaza media
  sigma <- sqrt(n * p * (1 - p)) #se calc deviatia standard

  z <- (x - miu) / sigma #vrem sa aflam z_n prin standardizare
  # Calculam P(Z_n <= x) folosind distributia normala standard
  print(paste("Media sumei:", miu))
  print(paste("Deviatia standard a sumei:", sigma))
  print(paste("Valoarea lui z:", z))
  pnorm(z)
}

n <- 1000 # nr de incercari
p <- 0.3 # probabilitatea de succes
x <- 20 # valoarea pentru care calculam probabilitatea

#Calculam P(Z_n<=x)
prob <- p_binomial(n,p,x)
print(prob)

```

### Output:

```

> #Calculam P(Z_n<=x)
> prob <- p_binomial(n,p,x)
[1] "Media sumei: 300"
[1] "Deviatia standard a sumei: 14.4913767461894"
[1] "Valoarea lui z: -19.3218356615859"
> print(prob)
[1] 1.759555e-83
>

```

## Geometrica

```
#-----
#Geometrica

p_geometrica <- function(n, p, x) {
  # Calculam media si deviatia standard
  miu <- 1/ p
  sigma <- sqrt( (1 - p) / p^2)
  miu_sum <- n * miu
  sigma_sum <- sqrt(n * sigma^2)
  z <- (x - miu_sum) / sigma_sum # Transformam x in variabila standardizata z_n
  print(paste("Media sumei:", miu))
  print(paste("Deviatia standard a sumei:", sigma))
  print(paste("Valoarea lui z:", z))
  return(pnorm(z))
}

n <- 1000 # Numarul de variabile aleatoare geometrice
p <- 0.2 # Probabilitatea de succes
x <- 50 # Valoarea pentru care calculam probabilitatea

# Calculam probabilitatea si o afisam
prob <- p_geometrica(n, p, x)
print(prob)
```

## Output:

```
> # Calculam probabilitatea si o afisam
> prob <- p_geometrica(n, p, x)
[1] "Media sumei: 5"
[1] "Deviatia standard a sumei: 4.47213595499958"
[1] "Valoarea lui z: -35.0017856687341"
> print(prob)
[1] 1.056702e-268
> |
```

## Poisson:

```
#-----  
#Poisson  
  
p_poisson <- function(n, lambda,x) {  
  # Calculam media si deviataia standard  
  miu <- n*lambda  
  sigma <- sqrt(n*lambda)  
  z <- (x - miu) / sigma # Transformam x in variabila standardizata z_n  
  print(paste("Media sumei:", miu))  
  print(paste("Deviatia standard a sumei:", sigma))  
  print(paste("Valoarea lui z:", z))  
  return(pnorm(z))  
}  
  
n <- 10 # Numarul de variabile aleatoare geometrice  
lambda <- 50 # Valoarea pentru care calculam probabilitatea  
x <- 10 #avem nevoie de un x suficient de mare altfel pnorm pentru nr negative ne da 0  
# Calculam probabilitatea si o afisam  
prob <- p_poisson(n,lambda,x)  
print(prob)
```

## Output

```
> # Calculam probabilitatea si o afisam  
> prob <- p_poisson(n,lambda,x)  
[1] "Media sumei: 500"  
[1] "Deviatia standard a sumei: 22.3606797749979"  
[1] "Valoarea lui z: -21.9134661794979"  
> print(prob)  
[1] 9.664828e-107  
> |
```

## Uniforma pe caz discret:

Uniforma pe caz discret,

$$\text{Stim } E[X] = \frac{a+b}{2} ; X \in \{a, a+1, \dots, b\}$$

$$\text{Var}(X) = E(X^2) - [E(X)]^2 = \frac{(b-a+1)^2 - 1}{12}$$

$$(E[X])^2 = \left(\frac{n-1}{2}\right)^2 = \frac{n^2 - 2n + 1}{4} ; \text{ unde } n = b - a + 1$$

$$E[X^2] = \sum_{i=0}^{n-1} i^2 \cdot \frac{1}{n} = \frac{1}{n} \sum_{i=0}^n i^2 = \frac{1}{n} \cdot \frac{n(n-1)(2n-1)}{6} =$$

$$= \frac{(n-1)(2n-1)}{6} = \frac{2n^2 - 3n + 1}{6}$$

$$\sigma^2(X) = E[X^2] - E[X]^2 = \frac{2n^2 - 3n + 1}{6} - \frac{n^2 - 2n + 1}{4} =$$

$$= \frac{4n^2 - 6n + 2 - 3n^2 + 6n - 3}{12} = \frac{n^2 - 1}{12}$$

#-----

#Uniforma de caz discret

```
p_uniforma_discreta <- function(a, b, n, x) {
  miu_suma <- n * (a + b) / 2
  sigma_suma <- sqrt(n * ((b - a + 1)^2 - 1) / 12)
  z <- (x - miu_suma) / sigma_suma
  print(paste("Media sumei:", miu_suma))
  print(paste("Deviatia standard a sumei:", sigma_suma))
  print(paste("Valoarea lui z:", z))
  pnorm(z)
}
```

```
a <- 1 # limita inferioara a intervalului
b <- 6 # limita superioara a intervalului
n <- 100 # nr de variabile aleatoare
x <- 350 # Valoarea pentru care calculam probabilitatea
```

```
# Calculam P(Z_n ≤ x)
prob <- p_uniforma_discreta(a, b, n, x)
print(prob)
```

## Output:

```
> # Calculam P(Z_n ≤ x)
> prob <- p_uniforma_discreta(a, b, n, x)
[1] "Media sumei: 350"
[1] "Deviatia standard a sumei: 17.0782512765993"
[1] "Valoarea lui z: 0"
> print(prob)
[1] 0.5
> |
```

## Uniform pe caz continuu:

Unif. pe caz continuu

$$\mathbb{E}[X] = \frac{a+b}{2}$$
$$\mathbb{E}(x^2) = \int_a^b x^2 f(x) dx = \frac{1}{b-a} \int_a^b x^2 dx = \frac{1}{b-a} \cdot x^3 / 3 =$$
$$= \frac{1}{b-a} \left( \frac{b^3}{3} - \frac{a^3}{3} \right) = \frac{b^3 - a^3}{3(b-a)} = \frac{(b-a)(b^2 + ab + a^2)}{3(b-a)} = \frac{b^2 + ab + a^2}{3}$$
$$\Rightarrow \text{Var}(x) = \frac{b^2 + ab + a^2}{3} - \left( \frac{a+b}{2} \right)^2 = \frac{4(b^2 + ab + a^2) / 3 - (a^2 - 2ab - b^2)}{12} =$$
$$= \frac{4b^2 + 4ab + 4a^2 - 3a^2 - 6ab - 3b^2}{12} = \frac{(b-a)^2}{12}$$
$$\Rightarrow \text{Var}(x) = \frac{(b-a)^2}{12}$$

```
#Uniforma de caz continuu
```

```
p_uniforma_continuu <- function(a, b, n, x) {  
    miu_suma <- n * (a + b) / 2  
    sigma_suma <- sqrt(n * ((b - a)^2 / 12))  
    z <- (x - miu_suma) / sigma_suma  
    print(paste("Media sumei:", miu_suma))  
    print(paste("Deviatia standard a sumei:", sigma_suma))  
    print(paste("Valoarea lui z:", z))  
    pnorm(z)  
}  
  
a <- 1      # limita inferioara a intervalului  
b <- 6      # limita superioara a intervalului  
n <- 100    # nr de variabile aleatoare  
x <- 350    # Valoarea pentru care calculam probabilitatea  
  
# Calculam P(Z_n ≤ x)  
prob <- p_uniforma_continuu(a, b, n, x)  
print(prob)
```

### Output:

```
> # Calculam P(Z_n ≤ x)  
> prob <- p_uniforma_continuu(a, b, n, x)  
[1] "Media sumei: 350"  
[1] "Deviatia standard a sumei: 14.4337567297406"  
[1] "Valoarea lui z: 0"  
> print(prob)  
[1] 0.5  
> |
```

## Exponentiala:

```
# Exponentiala

p_exponentiala <- function(lambda, n, x) {
  miu_suma <- n / lambda
  sigma_suma <- sqrt(n) / lambda

  z <- (x - miu_suma) / sigma_suma
  print(paste("Media sumei:", miu_suma))
  print(paste("Deviatia standard a sumei:", sigma_suma))
  print(paste("Valoarea lui z:", z))
  pnorm(z)
}

lambda <- 1      # Rata evenimentelor
n <- 100         # Numarul de variabile aleatoare
x <- 120

# Calculăm P(Z_n ≤ x)
prob <- p_exponentiala(lambda, n, x)
print(prob)
```

## Output:

```
> # Calculăm P(Z_n ≤ x)
> prob <- p_exponentiala(lambda, n, x)
[1] "Media sumei: 100"
[1] "Deviatia standard a sumei: 10"
[1] "Valoarea lui z: 2"
> print(prob)
[1] 0.9772499
> |
```

## Gamma:

```
#Gamma

p_gamma <- function(k, theta, n, x) {
  miu_suma <- n * k * theta
  sigma_suma <- sqrt(n * k) * theta
  z <- (x - miu_suma) / sigma_suma
  print(paste("Media sumei:", miu_suma))
  print(paste("Deviatia standard a sumei:", sigma_suma))
  print(paste("Valoarea lui z:", z))
  pnorm(z)
}

k <- 2          # Parametrul de forma
theta <- 30      # Parametrul de scala
n <- 50          # Numarul de variabile aleatoare
x <- 300

# Calculam P(Z_n ≤ x)
prob <- p_gamma(k, theta, n, x)
print(prob)
```

## Output:

```
> # Calculăm P(Z_n ≤ x)
> prob <- p_gamma(k, theta, n, x)
[1] "Media sumei: 3000"
[1] "Deviatia standard a sumei: 300"
[1] "Valoarea lui z: -9"
> print(prob)
[1] 1.128588e-19
> |
```

## Beta:

```
#Beta

p_beta <- function(alpha, beta, n, x) {
  miu_suma <- n * (alpha / (alpha + beta))
  sigma_suma <- sqrt(n * (alpha * beta) / ((alpha + beta)^2 * (alpha + beta + 1)))
  z <- (x - miu_suma) / sigma_suma

  print(paste("Media sumei:", miu_suma))
  print(paste("Deviatia standard a sumei:", sigma_suma))
  print(paste("Valoarea lui z:", z))
  pnorm(z)

}

alpha <- 2
beta <- 30
n <- 50
x <- 3.5

prob <- p_beta(alpha, beta, n, x)
print(prob)
```

## Output:

```
[1] "Media sumei: 3.125"
[1] "Deviatia standard a sumei: 0.297957059139248"
[1] "Valoarea lui z: 1.25857061780418"
> print(prob)
[1] 0.8959073
> |
```

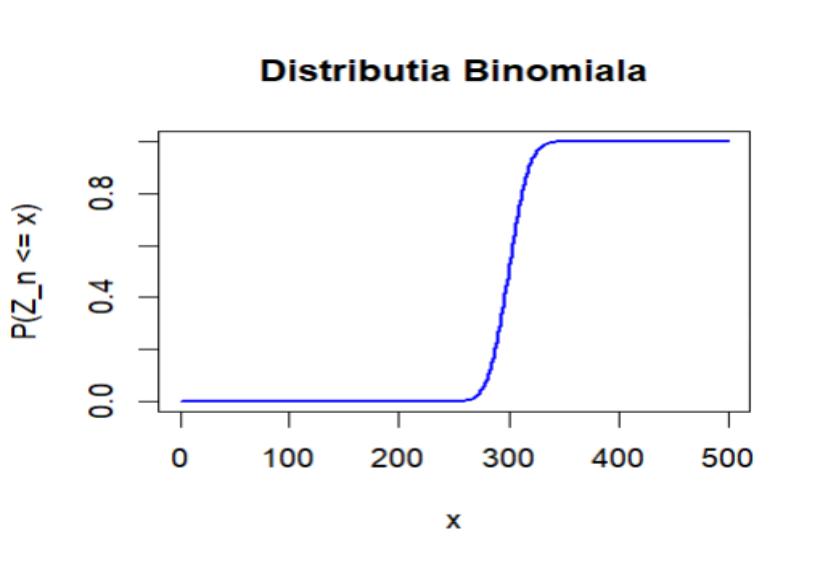
## Subpunctul 2:

2) Reprezentați grafic funcțiile obținute la 1).

### Binomiala:

```
#Binomiala
# Cream un vector de valori x pentru care calculam probabilitatea
x_vals <- seq(0, 500, by = 1) # De la 0 la 500 cu pasul 1
y_vals <- sapply(x_vals, function(x) p_binomial(n, p, x))

# Generam graficul
plot(x_vals, y_vals, type = "s", col = "blue", lwd = 2,
      main = "Distributia Binomiala",
      xlab = "x", ylab = "P(Z_n <= x)")
```



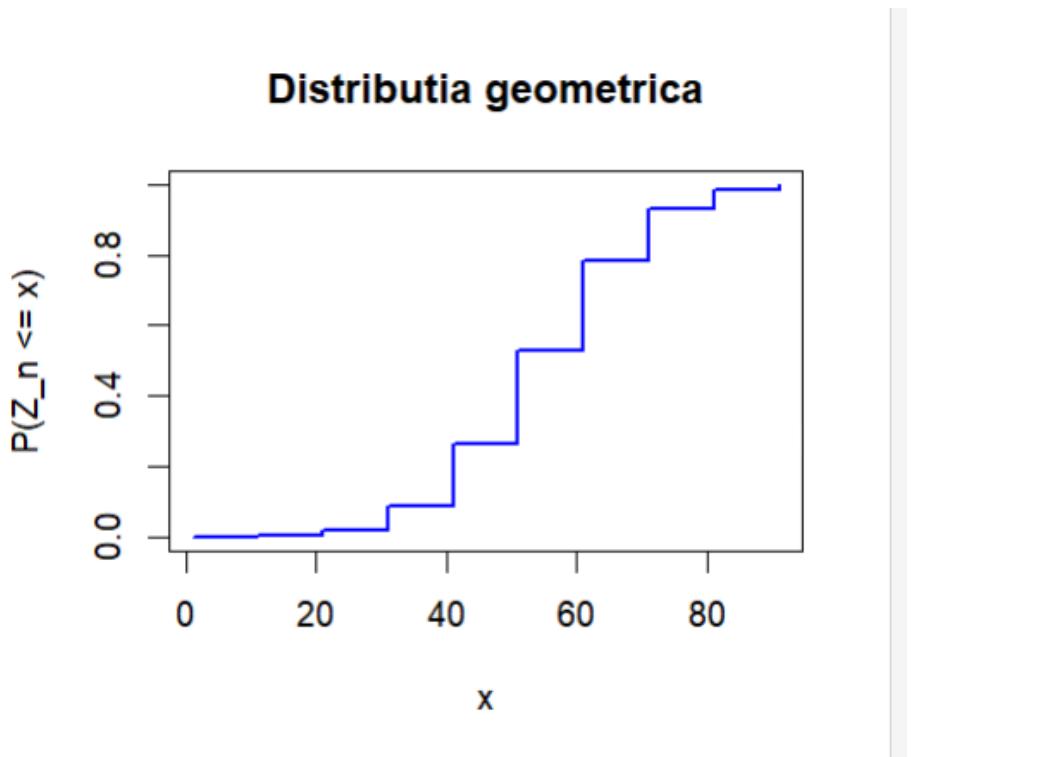
## Geometrica:

```
n <- 10 # Numarul de variabile aleatoare geometrice
p <- 0.2 # Probabilitatea de succes
x <- 5 # Valoarea pentru care calculam probabilitatea

prob <- p_geometrica(n, p, x)
print(prob)

#Graficul Geom.
x_values <- seq(1, 100, by = 10)
prob_values <- sapply(x_values, function(x) p_geometrica(n, p, x))

# Reprezentam grafic functia
plot(x_values, prob_values, type = "s", col = "blue", lwd = 2,
      xlab = "x", ylab = "P(Z_n <= x)", main = "Distributia geometrica")
```

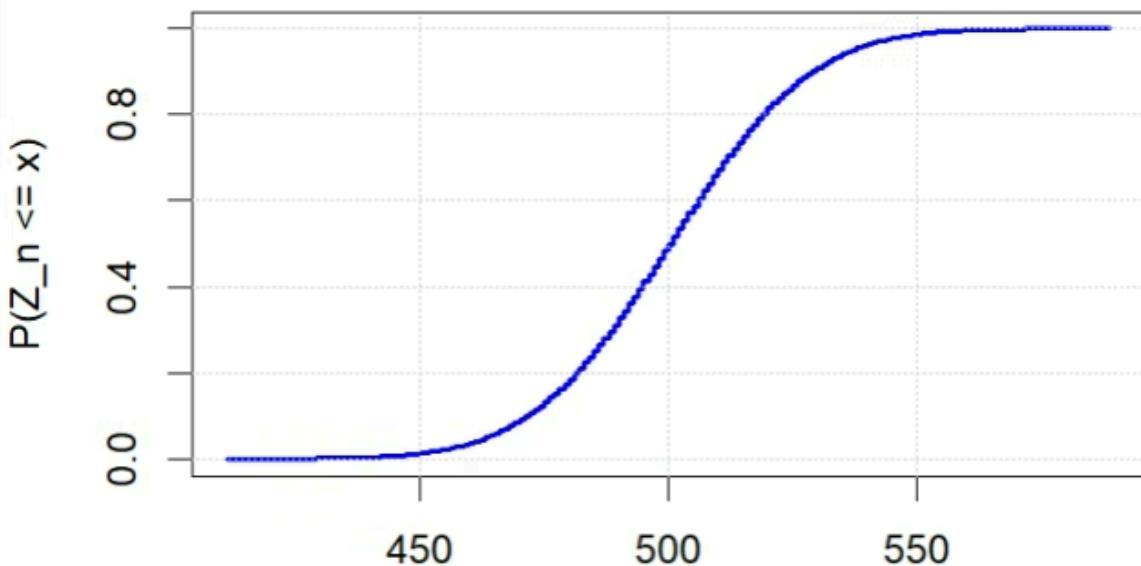


### Poisson:

```
n <- 100
lambda <- 5
# Generam valori pentru x
x_values <- seq(411, 589, by = 1) # avem nevoie de intervalul [miu+4*tetha;miu-4*tetha]
prob_values <- sapply(x_values, function(x) p_poisson(n, lambda, x))

# Reprezentam grafic functia
plot(x_values, prob_values, type = "s", col = "blue", lwd = 2,
      xlab = "x", ylab = "P(Z_n <= x)", main = "Functia de repartitie Poisson")
```

Functia de repartitie Poisson



## Uniforma pe caz discret:

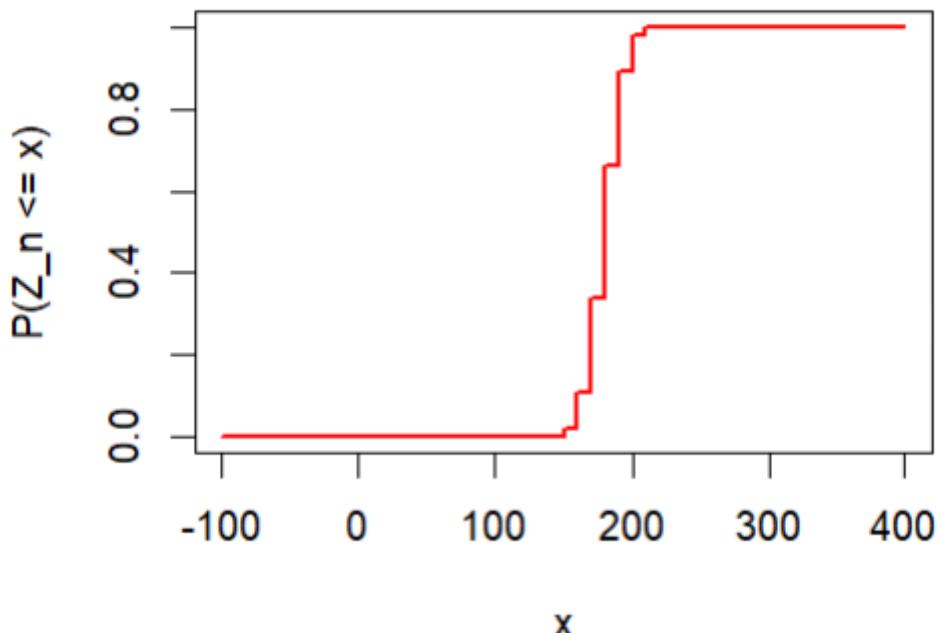
```
a <- 1      # limita inferioara a intervalului
b <- 6      # limita superioara a intervalului
n <- 50 # nr de variabile aleatoare
x <- 350 # Valoarea pentru care calculam probabilitatea

# Calculam P(Z_n ≤ x)
prob <- p_uniforma_discreta(a, b, n, x)
print(prob)

# 2) -----
#Graficul uniforma caz discret
x_values <- seq(-100, 400, by = 10)
prob_values <- sapply(x_values, function(x) p_uniforma_discreta(a, b, n, x))

# Reprezentam grafic functia
title <- "Distributia Uniforma Discreta"
plot(x_values, prob_values, type = "s", col = "red", lwd = 2,
     xlab = "x", ylab = "P(Z_n ≤ x)", main = title)
```

Distributia Uniforma Discreta



## Exponentiala:

```
lambda <- 1      # Rata evenimentelor
n <- 100         # Numărul de variabile aleatoare
x <- 120          # Valoarea pentru care calculăm probabilitatea

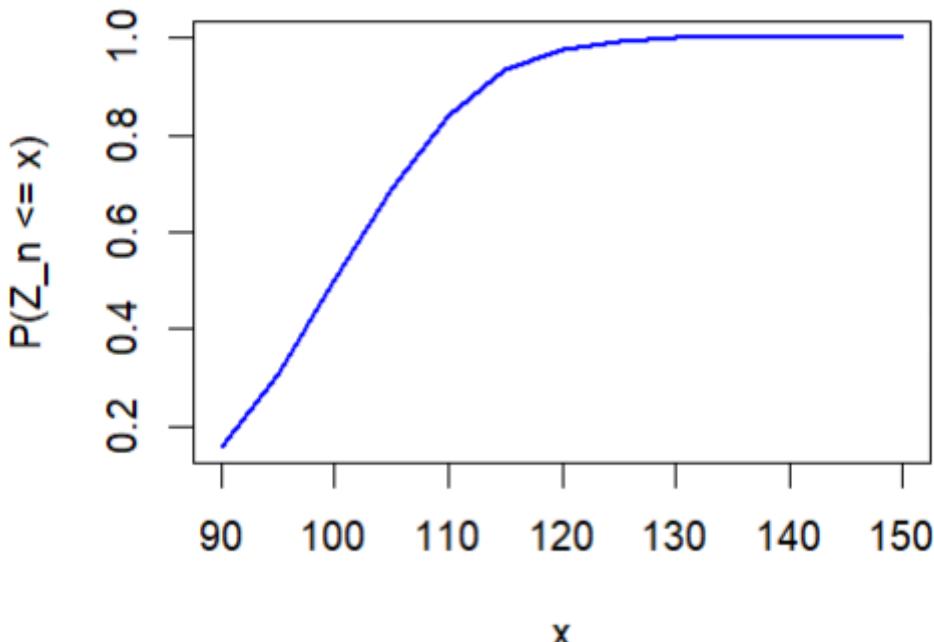
# Calculăm  $P(Z_n \leq x)$ 
prob <- p_exponentiala(lambda, n, x)
print(prob)

# 2) -----
# Generăm valori pentru x
x_values_exp <- seq(90, 150, by = 5)
prob_values_exp <- sapply(x_values_exp, function(x) p_exponentiala(lambda, n, x))

# Reprezentăm grafic funcția exponentială
title_exp <- "Distributia Exponentiala"
plot(x_values_exp, prob_values_exp, type = "l", col = "blue", lwd = 2,
     xlab = "x", ylab = "P(Z_n <= x)", main = title_exp)
```

"

**Distributia Exponentiala**



### Subpunctul 3:

- 3) Folosind funcția *optimize* aproximați pentru repartițiile de mai sus:

$$\sup_x |P(Z_n \leq x) - \Phi(x)|$$

### Binomiala

#### COD:

```
# Functia pentru supremum folosind p_binomial
sup_binomial <- function(n, p) {
  miu <- n * p
  sigma <- sqrt(n * p * (1 - p))

  # Definim functia care calculeaza diferența dintre p_binomial si normala
  diff_function <- function(x) {
    abs(p_binomial(n, p, x) - pnorm((x - miu) / sigma))
  }

  # Calculam limitele suportului binomial
  lower_bound <- 0
  upper_bound <- n

  # Gasim maximul diferenței folosind optimize pe suportul binomialei
  result <- optimize(diff_function, c(lower_bound, upper_bound), maximum = TRUE)
  print(result)
  return(result$objective)
}
```

```
$maximum
[1] 308

$objective
[1] 0.01270705

> print(sup_value)
[1] 0.01270705
>
```

## Geometrica:

### COD:

```
sup_geom <- function(n, p) {  
  miu <- 1/p  
  sigma <- sqrt((1 - p) / p^2)  
  miu_sum <- n * miu  
  sigma_sum <- sqrt(n * sigma^2)  
  # Definim functia care calculeaza diferența dintre p_geometrica si normala  
  diff_function <- function(x) {  
    abs(p_geometrica(n, p, x) - pgeom(floor((x - miu) / sigma), p))  
  }  
  # Calculam limitele suportului geometric  
  lower_bound <- 0  
  upper_bound <- n  
  # Gasim maximul diferenței folosind optimize pe suportul geometric  
  result <- optimize(diff_function, c(lower_bound, upper_bound), maximum = TRUE)  
  print(result)  
  return(result$objective)  
}
```

```
$maximum  
[1] 999.9999  
  
$objective  
[1] 0  
  
> print(sup_value)  
[1] 0  
>
```

## Beta

### COD:

```
sup_beta <- function(alpha, beta, n) {  
  miu <- n * (alpha / (alpha + beta))  
  sigma <- sqrt(n * (alpha * beta) / ((alpha + beta)^2 * (alpha + beta + 1)))  
  # Definim functia care calculeaza diferența dintre p_binomial si normala  
  diff_function <- function(x) {  
    abs(p_beta(alpha, beta, n, x) - pnorm((x - miu) / sigma))  
  }  
  # Calculam limitele suportului binomial  
  lower_bound <- 0  
  upper_bound <- n  
  # Gasim maximul diferenței folosind optimize pe suportul binomialei  
  result <- optimize(diff_function, c(lower_bound, upper_bound), maximum = TRUE)  
  print(result)  
  return(result$objective)  
}
```

## Gamma

Cod:

```
# Functia pentru supremum folosind p_gamma
sup_gamma <- function(n, k, theta) {
  miu_suma <- n * k * theta
  sigma_suma <- sqrt(n * k) * theta
  # Definim functia care calculeaza diferența dintre p_gamma si normala
  diff_function <- function(x) {
    abs(p_gamma(k, theta, n, x) - pnorm((x - miu_suma) / sigma_suma))
  }
  # Calculam limitele suportului gamma
  lower_bound <- 0
  upper_bound <- n
  # Gasim maximul diferenței folosind optimize pe suportul gamma
  result <- optimize(diff_function, c(lower_bound, upper_bound), maximum = TRUE)
  print(result)
  return(result$objective)
}
```

## Poisson:

Cod:

```
#3) POISSON

lambda <- 3
n <- 100
p <- 0.5

# Functia pentru supremum folosind p_binomial
sup_poisson <- function(n, lambda) {
  miu <- n*lambda
  sigma <- sqrt(n*lambda)
  # Definim functia care calculeaza diferența dintre p_binomial si normala
  diff_function <- function(x) {
    abs(p_poisson(n, lambda, x) - pnorm((x - miu) / sigma))
  }
  # Calculam limitele suportului binomial
  lower_bound <- 0
  upper_bound <- n
  # Gasim maximul diferenței folosind optimize pe suportul binomialei
  result <- optimize(diff_function, c(lower_bound, upper_bound), maximum = TRUE)
  print(result)
  return(result$objective)
}

sup_value <- sup_poisson(n, lambda)
print(sup_value)
```

## Exponentiala:

### Cod:

```
# Functia pentru supremum folosind p_exponentiala
sup_exponentiala <- function(lambda, n) {
  miu_suma <- n / lambda
  sigma_suma <- sqrt(n) / lambda

  # Definim functia care calculeaza diferența dintre p_exponentiala si normala
  diff_function <- function(x) {
    abs(p_exponentiala(lambda, n, x) - pnorm((x - miu_suma) / sigma_suma))
  }

  # Calculam limitele suportului exponentialei
  lower_bound <- 0
  upper_bound <- n / lambda
}

# Gasim maximul diferenței folosind optimize pe suportul exponentialei
result <- optimize(diff_function, c(lower_bound, upper_bound), maximum = TRUE)
return(result$objective)
```

- 4) Construiți câte o funcție în R care să calculeze  $E(X)$  și respectiv  $Var(X)$ , unde tipul repartiției v.a.  $X$  este transmis fie printr-o denumire, fie prin funcția de masă/funcția densitate de probabilitate, în cazul discret și respectiv continuu.

#### Subpunctul 4:

```
# 4)
calc <- function(nume, param) {
  if (nume == "binomial") {
    n <- param$n
    p <- param$p
    medie <- n * p
    varianta <- n * p * (1 - p)
  } else if (nume == "uniform") {
    a <- param$a
    b <- param$b
    medie <- (a + b) / 2
    varianta <- (b - a)^2 / 12
  } else if (nume == "gamma") {
    a <- param$a
    b <- param$b
    medie <- a * b
    varianta <- a * b^2
  } else if (nume == "beta") {
    alpha <- param$alpha
    beta <- param$beta
    medie <- alpha / (alpha + beta)
    varianta <- (alpha * beta) / ((alpha + beta)^2 * (alpha + beta + 1))
  } else if (nume == "poisson") {
    lambda <- param$lambda
    medie <- lambda
    varianta <- lambda
  } else if (nume == "geometric") {
    p <- param$p
    medie <- 1 / p
    varianta <- (1 - p) / p^2
  } else if (nume == "exponential") {
    lambda <- param$lambda
    medie <- 1 / lambda
    varianta <- 1 / lambda^2
  } else {
    stop("Distributie necunoscuta")
  }
  return(list(medie = medie, varianta = varianta))
}

# Exemplu de utilizare:
param <- list(n = 10, p = 0.5) # Parametri pentru o distributie binomiala
calc("binomial", param)

param <- list(alpha=2, beta= 2)
calc("beta",param)
```

## OUTPUT:

```
> print("Binomiala")
[1] "Binomiala"
> param <- list(n = 10, p = 0.5) #
> calc("binomial", param)
$medie
[1] 5

$varianta
[1] 2.5

>
> print("Beta")
[1] "Beta"
> param <- list(alpha=2, beta= 2)
> calc("beta",param)
$medie
[1] 0.5

$varianta
[1] 0.05
```

## Subpunctul 5:

5) Construiți o funcție în R care să calculeze  $E|X_1 - \mu|^3$

```
# 5) -----
calculeaza_E_abs_X_miu <- function(f_dens_masa, miu, lower = -Inf, upper = Inf, discreta= FALSE) {
  if (discreta) {
    val <- lower:upper # x1,x2...,xn
    E_abs_X_miu <- sum(f_dens_masa(val)) * (abs(val - miu))^3
  } else {
    E_abs_X_miu <- integrate(function(x) f_dens_masa(x) * (abs(x - miu))^3, lower, upper)$value
  }
  return(E_abs_X_miu)
}

#pt Normala(0,1)
f_dens_normal <- function(x) dnorm(x, mean = 0, sd = 1)
miu_normal <- 0

# Calculam  $E|X_1 - \mu|^3$  pentru o distribuție normală standard
calculeaza_E_abs_X_miu(f_dens_normal, miu_normal, lower = -Inf, upper = Inf, discreta = FALSE)
```

## OUTPUT:

```
> # Calculam E
> calculeaza_E_
[1] 1.595769
>
```

## PROBLEMA 2:

**2. Def.** O variabilă aleatoare  $X$  face parte din *familia exponențială s-dimensională* dacă densitatea/funcția de masă poate fi scrisă sub forma:

$$p(x; \theta) = h(x) \cdot \exp\left(\sum_{i=1}^s \eta_i(\theta) \cdot T_i(x) - A(\theta)\right),$$

unde  $\eta_i$  și  $A$  sunt funcții reale de  $\theta = (\theta_1, \theta_2, \dots, \theta_s)$ ,  $T_i$  reprezintă statistici suficiente, iar  $h$  este o funcție pozitivă de  $x$ .

Funcția  $A(\theta)$  se numește *constantă de log-normalizare* (rolul ei este acela de a face ca  $p(x; \theta)$  să îndeplinească acele condiții necesare pentru a fi o funcție de densitate de probabilitate/funcție de masă, după caz).

**Ex.** Fie  $X \sim Norm(\mu, \sigma^2)$ .

Vom demonstra că aceasta face parte din familia exponențială 2-dimensională:

$$\begin{aligned} f(x; \mu, \sigma^2) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\mu}{\sigma^2}x - \frac{1}{2\sigma^2}x^2 - \frac{\mu^2}{2\sigma^2}\right) \\ &= \frac{1}{\sqrt{2\pi}} \exp\left(\frac{\mu}{\sigma^2}x - \frac{1}{2\sigma^2}x^2 - \frac{\mu^2}{2\sigma^2} - \ln(\sigma)\right) \end{aligned}$$

Identificăm următoarele relații:

$$h(x) = \frac{1}{\sqrt{2\pi}}$$

$$\eta_1(\theta_1, \theta_2) = \frac{\mu}{\sigma^2}, \quad \eta_2(\theta_1, \theta_2) = -\frac{1}{2\sigma^2}, \quad \theta_1 = \mu, \quad \theta_2 = \sigma^2$$

$$A(\theta) = \frac{\mu^2}{2\sigma^2} + \ln(\sigma)$$

$$T_1(x) = x, \quad T_2(x) = x^2$$

Așadar, repartiția normală (cu parametrii media  $\mu$  și dispersia  $\sigma^2$ ) face parte din familia exponențială 2-dimensională.

- 1) Verificați dacă următoarele repartiții fac parte din familia exponențială:
- Binom(3,p)
  - Binom(n,p)
  - Geom(p)
  - Pois( $\lambda$ )
  - Gamma( $\alpha, \beta$ )
  - Beta( $\alpha, \beta$ )
  - $\chi^2(\nu)$

a)

① a)

$\text{Bin}(3, p)$ :

$$f(x) = C_3^x p^x (1-p)^{3-x}$$
$$= C_3^x e^{x \ln p + (3-x) \ln(1-p)}$$

$$\pi(x) = C_3^x$$

$$T_1(x) = x$$

$$T_2(x) = 3-x$$

$$\eta_1(p) = \ln(p)$$

$$\eta_2(p) = \ln(1-p)$$

b)

b)  $\text{Bin}(n, p)$ :

$$f(x) = e^{\ln(C_n^x) + \ln(p^x) + \ln((1-p)^{n-x})}$$
$$= C_n^x e^{\ln(p^x) + \ln((1-p)^{n-x})} = C_n^x e^{x \ln(p) + (n-x) \ln(1-p)}$$

$\Rightarrow f(x) = C_n^x$  nu se depinde de un parametru din cei doi.

c)

c)  $\text{Geom}(\theta)$ :

$$P_\theta(X=x) = (1-\theta)^{x-1} \cdot \theta = e^{\ln((1-\theta)^{x-1} \cdot \theta)} = e^{(x-1) \ln(1-\theta) + \ln(\theta)}$$
$$= e^{T_1(x) \eta_1(\theta) + T_2(x) \eta_2(\theta)}$$
$$\Rightarrow T_1(x) = (x-1), T_2(x) = 1$$
$$\eta_1(\theta) = \ln(1-\theta), \eta_2(\theta) = \ln(\theta)$$

d)

$$d) \text{ Poisson}(\lambda): P_\lambda(X=x) = e^{-\lambda} \cdot \frac{\lambda^x}{x!} = \frac{1}{x!} e^{-\lambda + \ln(\lambda^x)} = \frac{1}{x!} e^{-\lambda + x \ln(\lambda)}$$

$$\lambda(x) = \frac{1}{x!} \quad T_1(x) = x$$

$$A(x) = x \quad \eta_1(x) = \ln(\lambda)$$

e)

d)  $\Pi(\alpha, \beta)$ :

$$f(x) = \frac{\beta^\alpha x^{\alpha-1}}{\Gamma(\alpha)} e^{-\beta x} = e^{-\beta x + (\alpha-1)\ln(x) + \ln\left(\frac{\beta^\alpha}{\Gamma(\alpha)}\right)}$$

$$T_1(x) = -x \quad \eta_1(\alpha, \beta) = -\beta \quad A(\alpha, \beta) = \ln\left(\frac{\beta^\alpha}{\Gamma(\alpha)}\right)$$

$$T_2(x) = \ln(x) \quad \eta_2(\alpha, \beta) = \alpha-1$$

f)

$$f) \text{ Beta}(\alpha, b): f(x) = \frac{1}{\text{Beta}(\alpha, b)} x^{\alpha-1} (1-x)^{b-1} = e^{(\alpha-1)\ln(x) + (b-1)\ln(1-x) - \ln\beta(\alpha, b)}$$

$$\Rightarrow \eta_1(\alpha, b) = \alpha-1; \quad \eta_2(\alpha, b) = b-1 \quad A(\alpha, b) = -\ln\beta(\alpha, b)$$

$$T_1(x) = \ln(x) \quad T_2(x) = \ln(1-x)$$

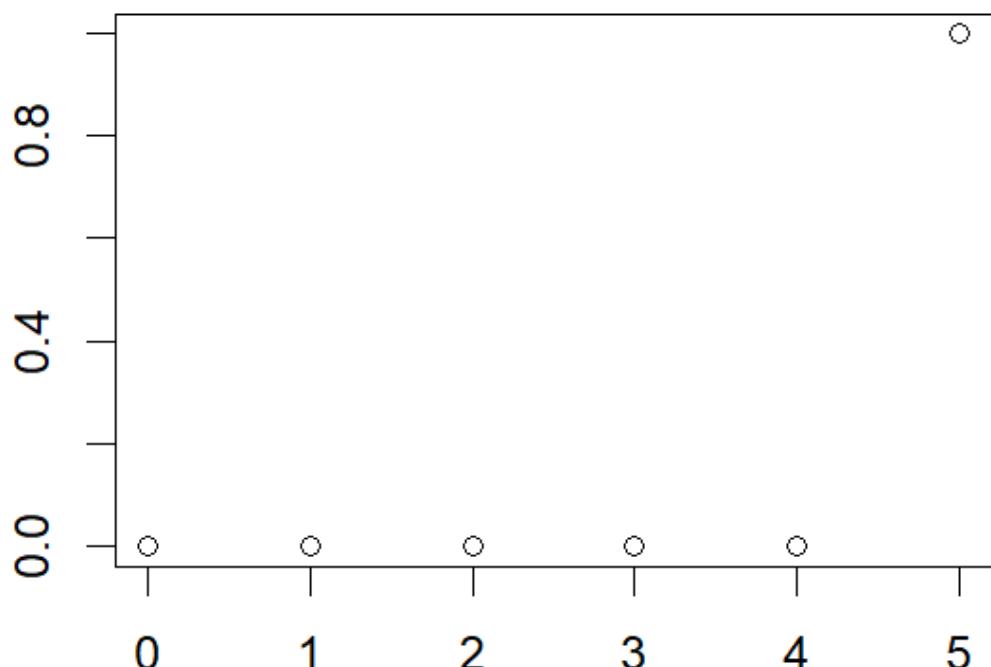
- 2) Ilustrați grafic în R densitățile/funcțiile de masă(după caz) ale repartițiilor de mai sus pentru 4 parametri particulari, la alegere, în cadrul aceluiași sistem de axe ortogonale(fiecare repartie va avea însă reprezentări grafice distincte de celelalte repartii).

## BINOMIALA:

COD:

```
#GRAFIC BINOMIALA- DISCRETA
div_space_bin = seq(0,5,1)
dens_bin = function(x) (dbinom(x,5,1))
plot(div_space_bin,sapply(div_space_bin,dens_bin))
```

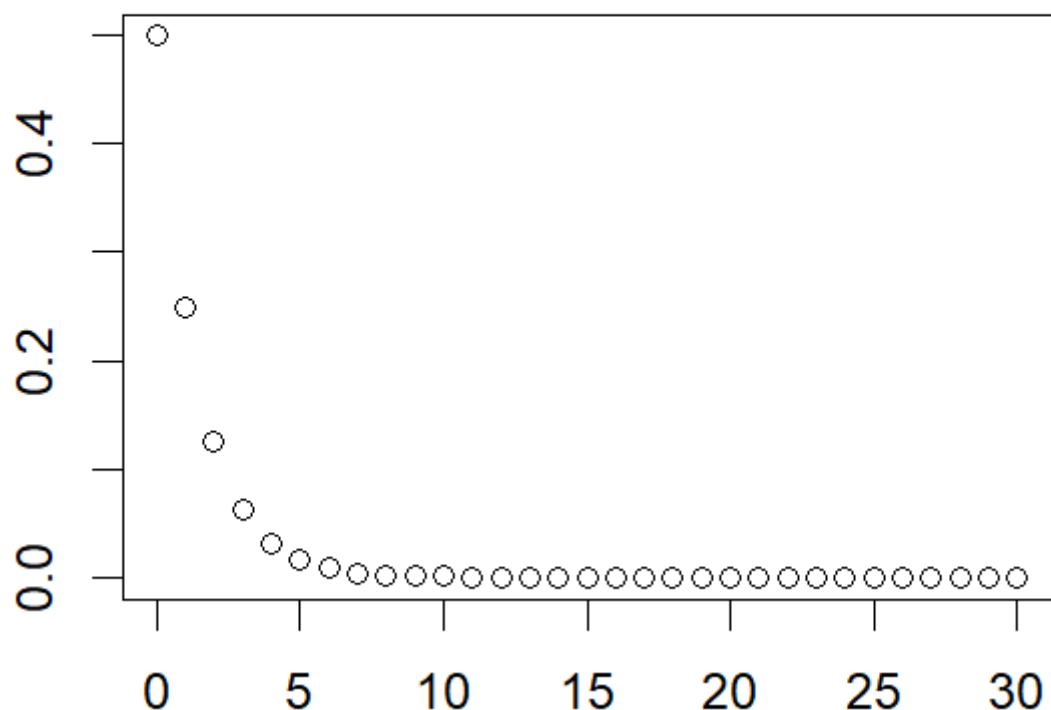
### Graficul binomialei



## GEOMETRICA:

```
#GRAFIC GEOMETRICA- DISCRETA
div_space_geom = seq(0,30,1)
dens_geom = function (x) (dgeom(x,1/2))
plot(div_space_geom,sapply(div_space_geom,dens_geom))
```

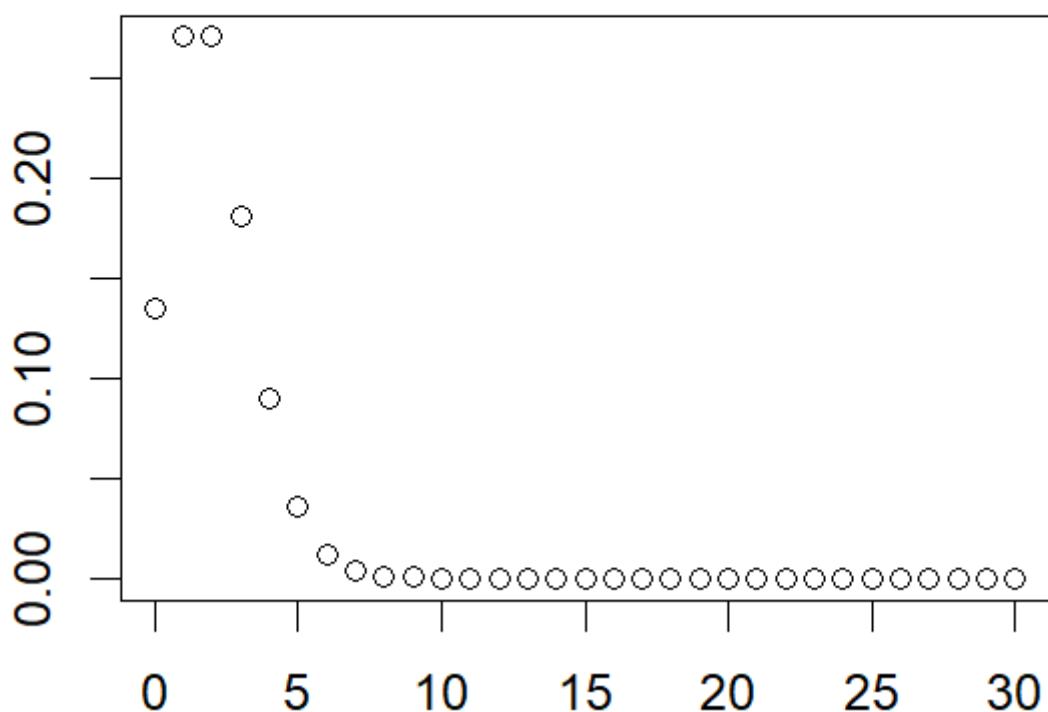
## Graficul:



**Poisson:**

```
#GRAFIC POISSON-DISCRETA
div_space_poiss = seq(0,30,1)
dens_poiss = function(x) (dpois(x,2))
plot(div_space_poiss,sapply(div_space_poiss,dens_poiss))
```

**Grafic:**

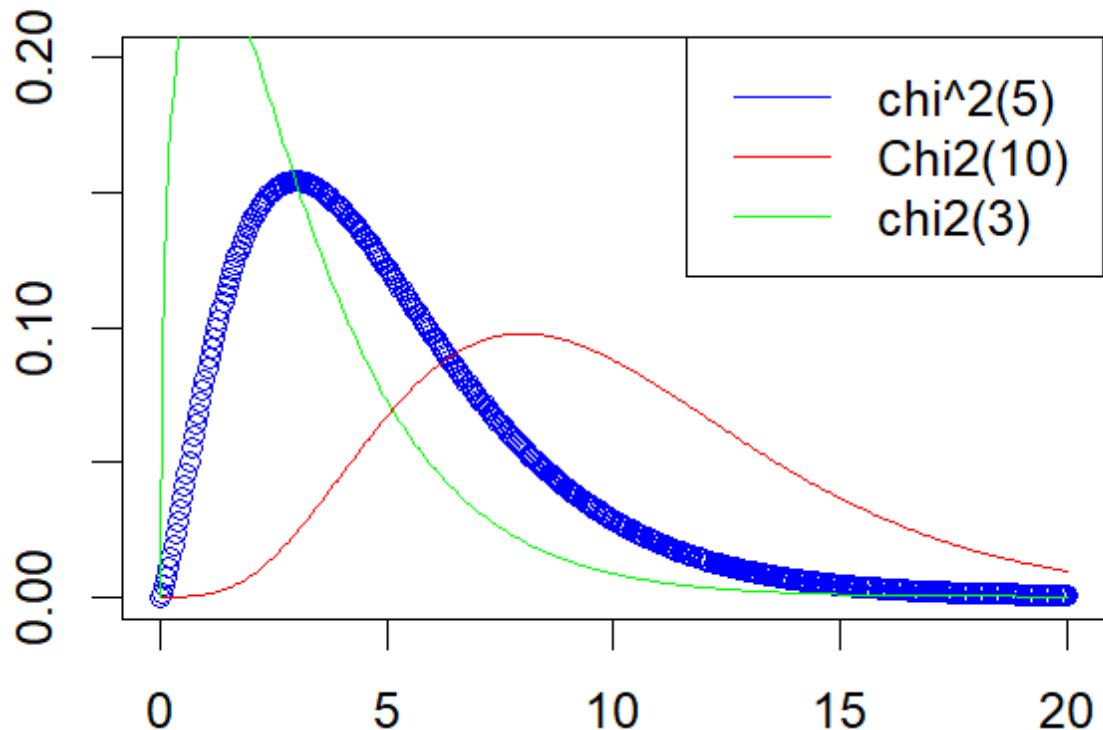


## Chi<sup>2</sup>

```
#GRAFIC CHI-PATRAT-UNIFORMA
div_space_chi = seq(0,20,0.05)
dens_chi2 = function (x,nu) (dchisq(x,nu))
plot(div_space_chi,sapply(div_space_chi,function(x) dchisq(x,5)),col = 'blue',ylim = c(0,0.2),main = 'Densitatea chi2')
lines(div_space_chi,sapply(div_space_chi,function(x) dchisq(x,10)),col = 'red')
lines(div_space_chi,sapply(div_space_chi,function(x) dchisq(x,3)),col = 'green')
legend('topright',legend = c('chi^2(5)', 'chi2(10)', 'chi2(3)'),col = c('blue','red','green'),lty = 1)
```

Grafic:

**Densitatea Chi2**

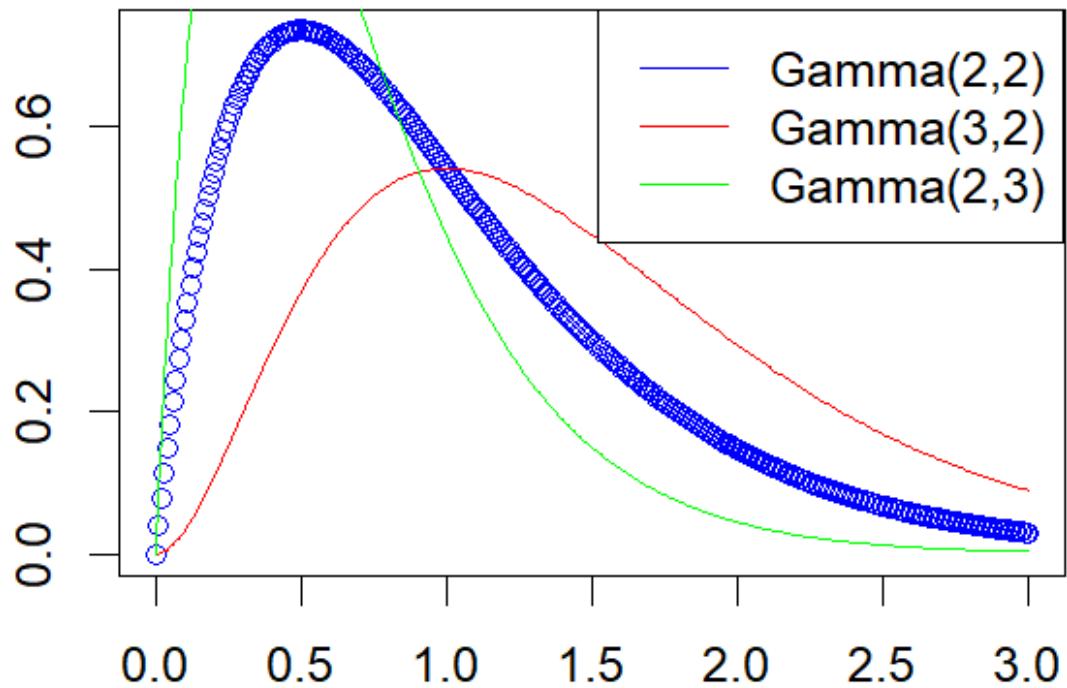


## Gamma:

```
#GRAFIC GAMMA-UNIFORMA
div_space_gamma = seq(0,3,0.01)
dens_gamma = function(x,alpha,lambda) (dgamma(x, alpha, shape = 1/lambda))
plot(div_space_gamma,sapply(div_space_gamma,function(x) dgamma(x,2,2)),col = 'blue',main = 'Densitatea Gamma')
lines(div_space_gamma,sapply(div_space_gamma,function(x) dgamma(x,3,2)),col = 'red')
lines(div_space_gamma,sapply(div_space_gamma,function(x) dgamma(x,2,3)),col = 'green')
legend('topright',legend = c('Gamma(2,2)', 'Gamma(3,2)', 'Gamma(2,3)'),col = c('blue', 'red', 'green'),lty = 1)
```

## Grafic:

**Densitatea Gamma**

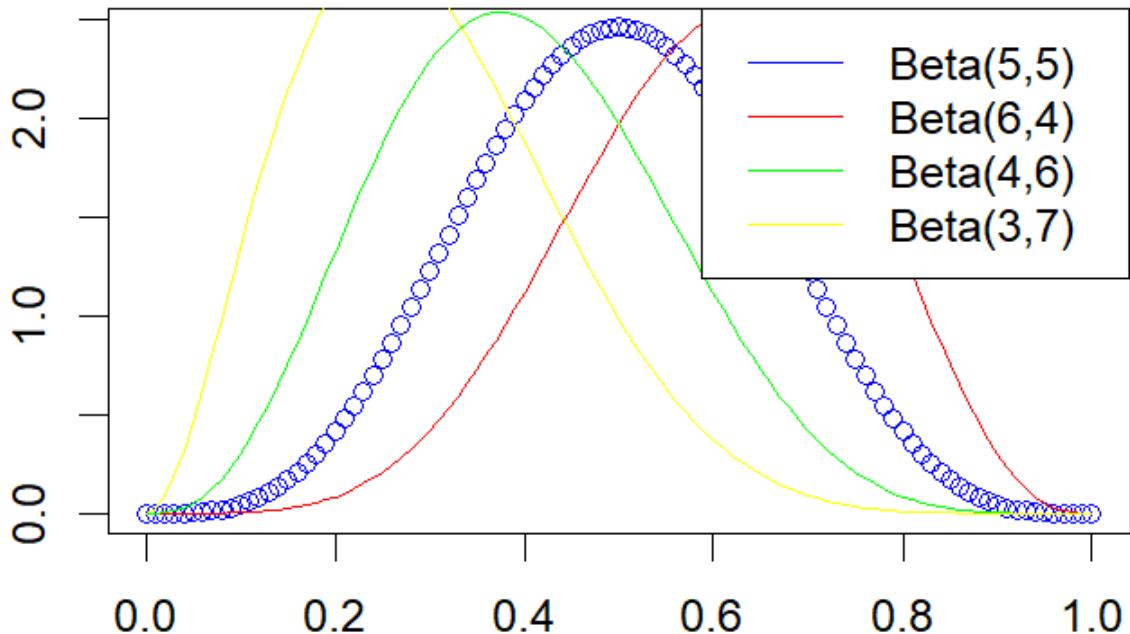


## Beta:

```
#GRAFIC BETA-UNIFORMA
div_space_beta = seq(0,1,0.01)
dens_beta = function (x,alpha,beta) (dbeta(x,alpha,beta))
plot(div_space_beta,sapply(div_space_beta,function(x) dbeta(x,5,5)),col = 'blue',main = 'Densitatea Beta')
lines(div_space_beta,sapply(div_space_beta,function(x) dbeta(x,6,4)),col = 'red')
lines(div_space_beta,sapply(div_space_beta,function(x) dbeta(x,4,6)),col = 'green')
lines(div_space_beta,sapply(div_space_beta,function(x) dbeta(x,3,7)),col = 'yellow')
legend('topright',legend = c('Beta(5,5)', 'Beta(6,4)', 'Beta(4,6)', 'Beta(3,7)'),col = c('blue', 'red', 'green', 'yellow'),lty = 1)
```

## Grafic:

**Densitatea Beta**



4) Particularizați forma funcției de log-verosimilitate de la 2) pentru repartițiile de la 1) ce fac parte din clasa exponențială cu un parametru. Reprezentați grafic(în  $\mathbf{R}$ ) aceste funcții și găsiți punctul lor de maxim(folosiți funcția **optimise**).

5) Particularizați forma funcției de log-verosimilitate de la 2) pentru repartițiile de la 1) ce fac parte din clasa exponențială cu doi parametri. Fixați, pe rând, unul din parametri(alegeti o valoare particulară după cum doriti), reprezentați grafic(în  $\mathbf{R}$ ) aceste funcții în raport cu celălalt parametru și găsiți punctul lor de maxim(folosiți funcția **optimise**).

**4)-5)**

**BINOMIALA:**

$$\begin{aligned}
 \text{Bin}(3, p): f(x) &= C_3^x e^{-\ln(p)} + (3-x) \ln(1-p) \\
 &= C_3^x e^{-\ln(p)} + 3 \ln(1-p) - x \ln(1-p) \\
 &= C_3^x e^{x(\ln(p) - \ln(1-p)) + 3 \ln(1-p)} \\
 &= C_3^x e^{x \ln\left(\frac{p}{1-p}\right)} \cdot e^{3 \ln(1-p)}
 \end{aligned}$$

$$L(x_1, \dots, x_n, p) = \prod_{i=1}^n C_3^{x_i} e^{x_i \ln\left(\frac{p}{1-p}\right)} e^{3 \ln(1-p)}$$

$$\ln L(x_1, \dots, x_n, p) = \sum_{i=1}^n \ln(C_3^{x_i}) + \sum_{i=1}^n x_i \ln\left(\frac{p}{1-p}\right) + \sum_{i=1}^n 3 \ln(1-p)$$

$$= \sum_{i=1}^n \ln(C_3^{x_i}) + \ln\left(\frac{p}{1-p}\right) \sum_{i=1}^n x_i + 3n \ln(1-p)$$

$$\begin{aligned}
 \frac{\partial \ln L}{\partial p} &= \ln\left(\frac{p}{1-p}\right)' \sum_{i=1}^n x_i + 3n \cdot (\ln(1-p))' = 0 \quad | \\
 \ln\left(\frac{p}{1-p}\right)' &= \frac{(\frac{p}{1-p})'}{\frac{p}{1-p}} = \frac{1}{(1-p)^2} \cdot \frac{1-p}{p} = \frac{1}{p(1-p)} = \frac{1}{p} + \frac{1}{1-p} \quad | \\
 \left(\frac{p}{1-p}\right)' &= \frac{1-p-p \cdot (-1)}{(1-p)^2} = \frac{1-p+p}{(1-p)^2} = \frac{1}{(1-p)^2}
 \end{aligned}$$

$$\textcircled{2} \frac{\partial L}{\partial p} = \left( \frac{1}{p} + \frac{1}{1-p} \right) \sum_{i=1}^n x_i + \frac{3m}{1-p} \cdot (1) = 0$$

$$\Leftrightarrow \left( \frac{1}{p} + \frac{1}{1-p} \right) \sum_{i=1}^n x_i = \frac{3m}{1-p}$$

$$\Leftrightarrow \sum_{i=1}^n x_i = \frac{3m}{1-p} \cdot \frac{p(1-p)}{1}$$

$$\Leftrightarrow \sum_{i=1}^n x_i = 3mp.$$

$$\frac{\sum_{i=1}^n x_i}{3m} = p \Rightarrow \hat{p} = \frac{\bar{x}}{3}$$

$$\frac{\partial^2 L}{\partial p^2} = \left( -\frac{1}{p^2} - \frac{1}{(1-p)^2} \right) \sum_{i=1}^n x_i - \frac{3m}{(1-p)^2} < 0 \Rightarrow \text{are punct de maxima}$$

COD:

```
#Binomiala
comb = function (n,k)
{
  return (factorial(n)/(factorial(k)*factorial(n-k)))
}

log_ver_bin = function (x,n,p)
{
  return (sum(sapply(x,function (k) log(comb(n,k))))+log(p/(1-p))*sum(x)+n*length(x)*log(1-p))
}
max_ver_bin = function(x,n)
{
  "Returneaza estimatia de verosimilitate maxima pentru repartitia chi2"
  log_ver_bin = function (p)
  {sum(sapply(x,function (k) log(comb(n,k))))+log(p/(1-p))*sum(x)+n*length(x)*log(1-p)}
  res_bin = optimize(function (x)-log_ver_bin(x),c(0,1))
  return (res_bin)
}
```

**Punctul de maxim:**

```
> print(res_bin1)
```

```
$maximum
```

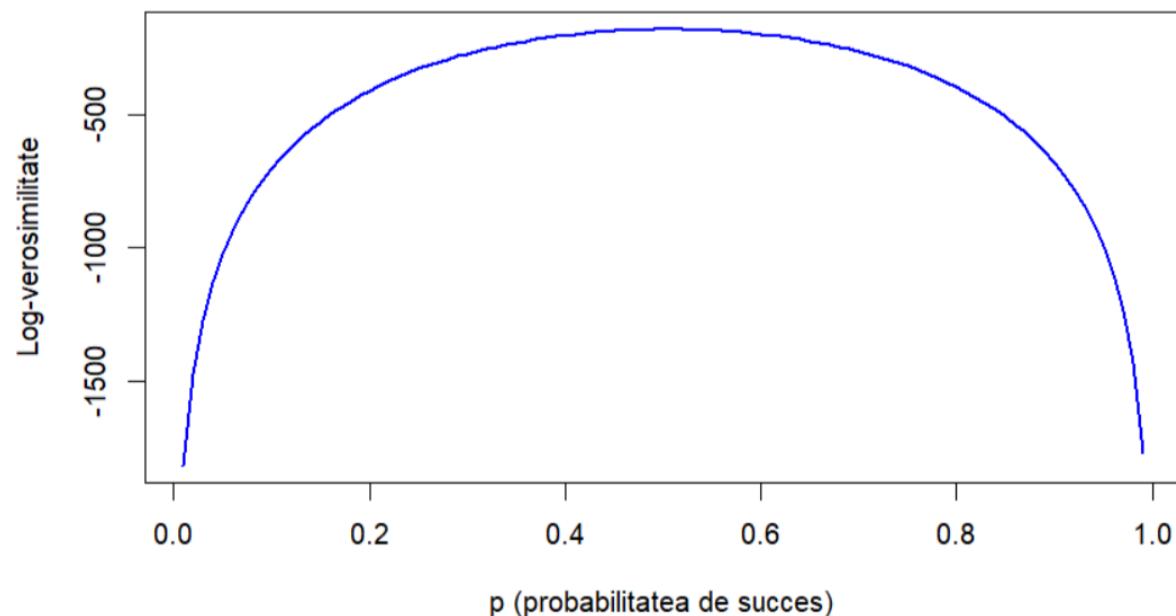
```
[1] 6.610696e-05
```

```
$objective
```

```
[1] 436.6919
```

**Grafic:**

**Funcția de log-verosimilitate pentru distribuția Binomială**



## GEOMETRICA:

Problema 2

Se dă la subiectul 1) avem:

$$f_{\text{geom}}(\theta) = P_{\theta}(X=x) = (1-\theta)^{x-1} \cdot \theta = e^{\ln((1-\theta)^{x-1} \cdot \theta)}$$

$$= e^{(x-1) \cdot \ln(1-\theta) + \ln \theta}$$

$$L(x_1, \dots, x_m; \theta) = \prod_{i=1}^m e^{(x_i) \cdot \ln(1-\theta) + \ln \theta}$$

$$\ln L(x_1, \dots, x_m; \theta) = \left(-m + \sum_{i=1}^m x_i\right) \cdot \ln(1-\theta) + m \ln \theta \quad \text{notă } h(\theta)$$

$$\frac{\partial}{\partial \theta} h(\theta) = \frac{-1}{1-\theta} \left(-m + \sum_{i=1}^m x_i\right) + \frac{m}{\theta}$$

$$h''(\theta) = \frac{-1}{(1-\theta)^2} \left(-m + \sum_{i=1}^m x_i\right) - \frac{m}{\theta^2} < 0 \Rightarrow \theta \text{ punct de maxim}$$

$\Rightarrow$  aflat estimatorul astfel:

$$\frac{\partial}{\partial \theta} h(\theta) = 0$$

$$\frac{m}{\theta} = \frac{1}{1-\theta} \left(-m + \sum_{i=1}^m x_i\right)$$

$$\theta \left(-m + \sum_{i=1}^m x_i\right) = m(1-\theta)$$

~~$$\theta(-m) + \theta \sum_{i=1}^m x_i = m - m\theta$$~~

$$\theta = \frac{m}{\sum_{i=1}^m x_i}$$

## COD:

```
# Geometrica
log_ver_geom = function(x,theta)
{
  return ((sum(x)-length(x))*log(1-p)+length(x)*log(p))
}
max_ver_geom = function(x)
{
  "Returneaza estimatia p(sau theta) care maximizeaza verosimilitatea"
  log_ver_geom = function (p) {(sum(x)-length(x))*log(1-p)+length(x)*log(p)}
  return (optimize(function (x) -log_ver_geom(x),c(0,1)))
}
```

## Punct de maxim:

```
> print(res_geom1)
```

```
$maximum
```

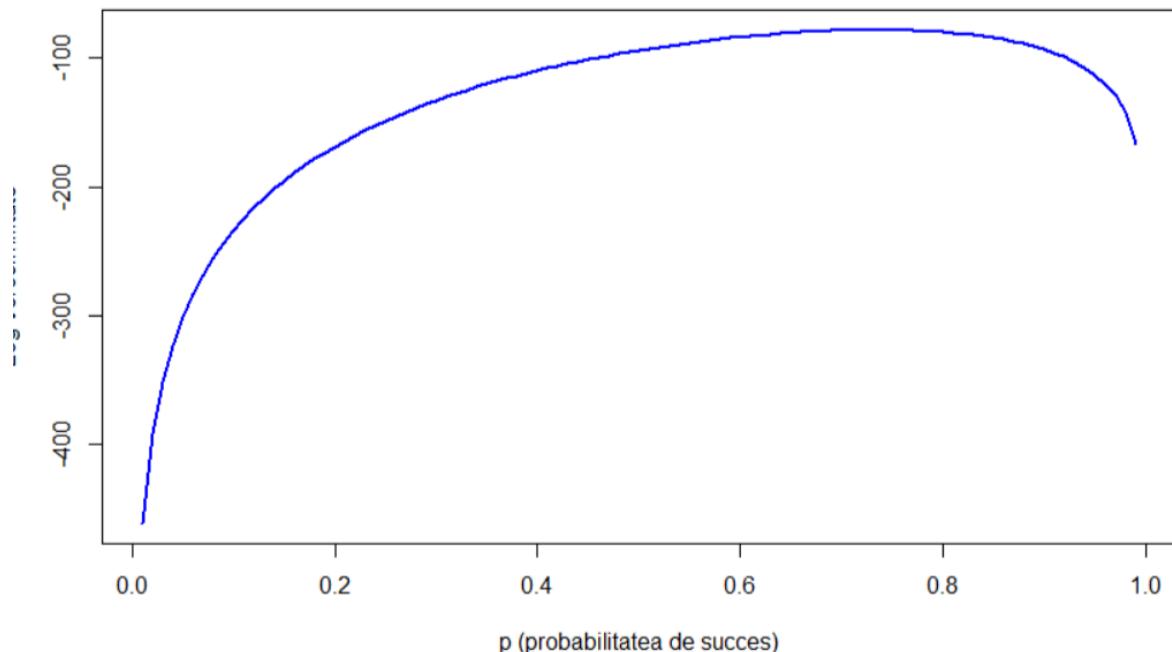
```
[1] 0.9999339
```

```
$objective
```

```
[1] 336.8491
```

## Grafic:

Functia de log-verosimilitate pentru distributia geometrica



## POISSON

$$\begin{aligned}
 \text{Poisson}(\lambda) : P_\lambda(x=x) &= e^{-\lambda} \cdot \frac{\lambda^x}{x!} = \frac{1}{x!} e^{-\lambda+x\ln(\lambda)} \\
 L_n(x_1, \dots, x_n; \lambda) &= \prod_{i=1}^n \frac{1}{x_i!} \cdot e^{-\lambda+x_i\ln(\lambda)} \\
 \Rightarrow \ln L_n(x_1, \dots, x_n; \lambda) &= \ln \left( \prod_{i=1}^n \frac{1}{x_i!} \cdot e^{-\lambda+x_i\ln(\lambda)} \right) \\
 &= \sum_{i=1}^n \ln \frac{1}{x_i!} + \sum_{i=1}^n -\lambda + x_i \ln(\lambda) = -n\lambda + \sum_{i=1}^n x_i \ln(\lambda) \\
 \frac{d \ln L}{d \lambda} &= -n + \frac{\sum_{i=1}^n x_i}{\lambda} = 0 \Rightarrow \frac{\sum_{i=1}^n x_i}{\lambda} = n \Rightarrow \\
 \Rightarrow \sum_{i=1}^n x_i &= n\lambda \Rightarrow \lambda = \frac{1}{n} \sum_{i=1}^n x_i \Rightarrow \boxed{\hat{\lambda} = \bar{x}} \\
 \text{Cuv. } \frac{d^2 \ln L}{d \lambda^2} &= -\frac{\sum_{i=1}^n x_i}{\lambda^2} < 0 \Rightarrow \text{areu pe de maxima.}
 \end{aligned}$$

## COD:

```
#Poisson
```

```

log_ver_poiss = function(x, lbd)
{
  n = length(x)
  return (-n*lbd + log(lbd)*sum(x))
}
max_ver_poiss = function(x, xmax)
{
  n = length(x)
  log_ver = function(lbd) -n*lbd + log(lbd)*sum(x)
  res = optimize(function (lbd) -log_ver(lbd), c(0, xmax), maximum = TRUE)
  return (res)
}
res_poiss1 = max_ver_poiss(x, 10)
print(res_poiss1)

```

**Punct de maxim:**

```
> print(res_poiss1)
```

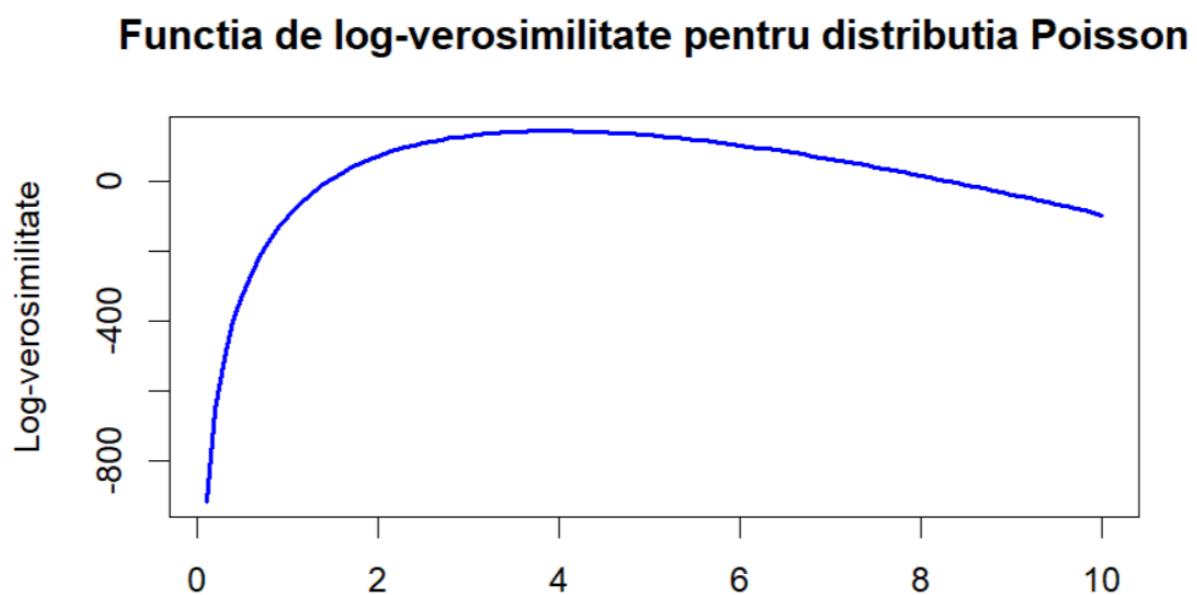
```
$maximum
```

```
[1] 9.999944
```

```
$objective
```

```
[1] 11.77809
```

**Grafic:**



Beta:

Pygmy  $\beta(a, b)$ :

$$f(x) = \frac{1}{\text{Beta}(a, b)} x^{a-1} (1-x)^{b-1}$$

$$= e^{(a-1)\ln(x) + (b-1)\ln(1-x) - \ln \text{Beta}(a, b)}$$

$$L(x_1, \dots, x_n, a, b) = \prod_{i=1}^n \frac{1}{\text{Beta}(a, b)} (x_i)^{a-1} (1-x_i)^{b-1}$$

$$\ln L(x_1, \dots, x_n, a, b) = -n \ln \text{Beta}(a, b) + (a-1) \sum_{i=1}^n \ln(x_i) + (b-1) \sum_{i=1}^n \ln(1-x_i)$$

$$\frac{\partial \ln L}{\partial a} = -\frac{n}{\text{Beta}(a, b)} \cdot \frac{\partial}{\partial a} (\text{Beta}(a, b)) + \sum_{i=1}^n \ln(x_i)$$

$$= -\frac{n}{\text{Beta}(a, b)} \cdot \frac{\partial}{\partial a} \left( \int_0^1 x^{a-1} (1-x)^{b-1} dx \right) + \sum_{i=1}^n \ln(x_i) = 0$$

$$\frac{\partial \ln L}{\partial b} = -\frac{n}{\text{Beta}(a, b)} \cdot \frac{\partial}{\partial b} \left( \int_0^1 x^{a-1} (1-x)^{b-1} dx \right) + \sum_{i=1}^n \ln(1-x_i) = 0$$