Documentaţie Proiect

Probabilitate si Statistica

Membri:
Gheorghe Bogdan-Alexandru
Andrei Cristian-David
Sincari Sebastian-George
Cosor Mihail

Ianuarie 2025

Contents

Cerinta I

Descrierea Problemei

Se consideră o activitate care presupune parcurgerea secvențială a n etape. Timpul necesar finalizării etapei i de către o persoană A este o variabilă aleatoare $T_i \sim \text{Exp}(\lambda_i)$.

După finalizarea etapei i, A va trece în etapa $i+1$ cu probabilitatea α_i sau va opri lucrul cu probabilitatea $1-\alpha_i$.

Fie T timpul total petrecut de persoana A în realizarea activității respective.

```r
set.seed(123)
n <- 100 # Nr etape
lambda <- runif(n, 0.5, 2) # Rata exp pt fiecare etapa
alpha <- runif(n-1, 0.8, 1) # Prob trecerii la urmatoarea etapa
alpha <- c(1, alpha) # Prob ca o persoana sa participe la prima etapa este 100%
nrSimulari <- 1000000 # Nr simulari

simulator <- function() {
  Ti <- rexp(n, rate = lambda)  # Timpul pentru fiecare etapa

  stop_point <- which(runif(n - 1) > alpha[-1])[1]  # Determinam momentul opririi

  # Timpul total va fi cel pana la oprire/finalul vectorului daca nu se opreste
  if (!is.na(stop_point)) {
    return(sum(Ti[1:stop_point]))
  } else {
    return(sum(Ti))
  }
}

# Simulam 10^6
valoriT <- replicate(nrSimulari, simulator())
```

1. Construiți un algoritm în R care simulează 10^6 valori pentru v.a. T și în baza acestora aproximați $E(T)$. Reprezentați grafic într-o manieră adecvată valorile obținute pentru T. Ce puteți spune despre repartiția lui T?

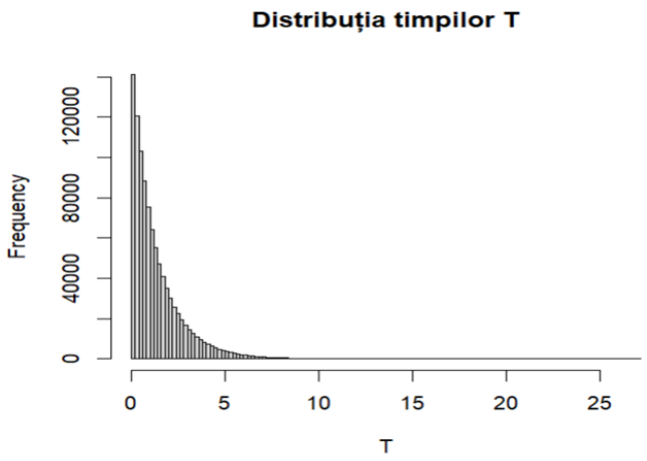
```r
approx_E_T <- mean(valoriT)

hist(valoriT, breaks = 100, main = "Distributia timpilor T", xlab = "T")
```

Explicații:

Pentru calculul aproximativ al mediei am făcut pur și simplu media valorilor rezultate în urma simulării.

Afișând în histograma valorile obținute în `valoriT` rezultă distribuția lui T. Se poate observa că reprezentarea grafică a timpiilor T urmează o distribuție exponențială.



Rezultat:

```
approx_E_T = 9.51113170988666
```

2. Calculați valoarea exactă a lui $E[T]$ și comparați cu valoarea obținută prin simulare.

```
exact_E_T <- sum(1 / \lambda * cumprod(alpha))
```

Explicații:

Valoarea exactă a lui $E[T]$ se calculează după următoarea formulă:

$$E[T] = E[T_1] + \alpha_1 E[T_2] + \alpha_1\alpha_2 E[T_3] + \ldots + \alpha_1\alpha_2 \ldots \alpha_{n-1} E[T_n]$$

Am folosit funcția `cumprod`, care returnează un vector cu produsul cumulativ. α are 99 de valori deoarece există $n-1$ pași pentru trecerea de la o etapă la alta. Am atașat acestui vector la început valoarea 1, deoarece probabilitatea ca persoana să participe la prima etapă este 100. Apoi am înmulțit produsul cumulativ cu $1/\lambda$, deoarece $X \sim \text{Exp}(\lambda) \Rightarrow E[X] = 1/\lambda$. În final, am făcut suma tuturor mediilor pe fiecare etapă, rezultând $E[T]$.

În urma analizării rezultatului exact observăm că este foarte apropiat de rezultatul aproximativ obținut prin simulare.

Rezultat:

```
    exact_E_T = 9.51299920302413
```

3. În baza simulărilor de la 1. aproximați probabilitatea ca persoana A să finalizeze activitatea.

```
    prob_finalizare <- mean(valoriT >= sum(1 / lambda))
```

Explicatii:

Am calculat probabilitatea ca o persoana sa finalizeze fiecare etapa luand ca exemplu in simularea noastra un numar de 100 de etape si o probabilitate de a trece de la o etapa la alta ¿=80

In scrierea codului am calculat valoarea teoretica a timpului total pe care o persoana trebuie sa il petreaca in cele 100 de etape pentru a spune ca a finalizat. Astfel am calculat media unui vector de valori booleene care valoarea true corespunde rezultatului unei simulari¿= valoarea timpului adica persoanele care au finalizat respectiv false pentru persoanele care nu au reusit sa finalizeze. Rezultatul este mic datorita numarului de etape, acesta ar creste o data cu micsorarea numarului de etape/ ar scadea o data cu cresterea numarului de etape. Totodata, rezultatul va scadea odata cu micsorarea marginii inferioare a valorilor probabilitatilor din alpha si invers.

Rezultat:

```
prob_finalizare = 0.00004
```

4. În baza simulărilor de la 1. aproximaţi probabilitatea ca persoana A să finalizeze activitatea într-un timp mai mic sau egal cu σ .

```
sigma <-94
prob_sigma<- mean(valoriT <= sigma & valoriT >= sum(1 / lambda))
```

Explicatii:

Ne-am folosit de modalitatea de rezolvare a exercitiului anterior, astfel am folosit un vector de valori booleene care verifica 2 conditii ¡= sigma si ¿= valoarea timpului total, apoi am facut media si ne-a rezultat valoarea probabilitatii ca o persoana sa finalizeze intr-un timp mai mic decat sigma. Este normal ca valoarea acestei probabilitati sa fie mai mica sau egala cu valoarea probabilitatii de a finaliza.

Rezultat:

```
prob_sigma = 0.000007
```

5. În baza simulărilor de la 1. determinaţi timpul minim și respectiv timpul maxim în care persoana A finalizează activitatea și reprezentaţi grafic timpii de finalizare a activităţii din fiecare simulare. Ce puteți spune despre repartiția acestor timpi de finalizare a activităţii?

```
timpMin <- min(valoriT[valoriT >= sum(1 / lambda)] )
timpMax <- max(valoriT[valoriT >= sum(1 / lambda)] )
cat("Timpul minim:", timpMin, "Timpul maxim:", timpMax, "\n")

hist(valoriT[valoriT >= sum(1 / lambda)], breaks=50,
             main="Distributia timpilor de finalizare",
             xlab="Timpul de finalizare (T)", ylab="Frecventa",
             col="lightblue", border="black")
```

Explicatii:

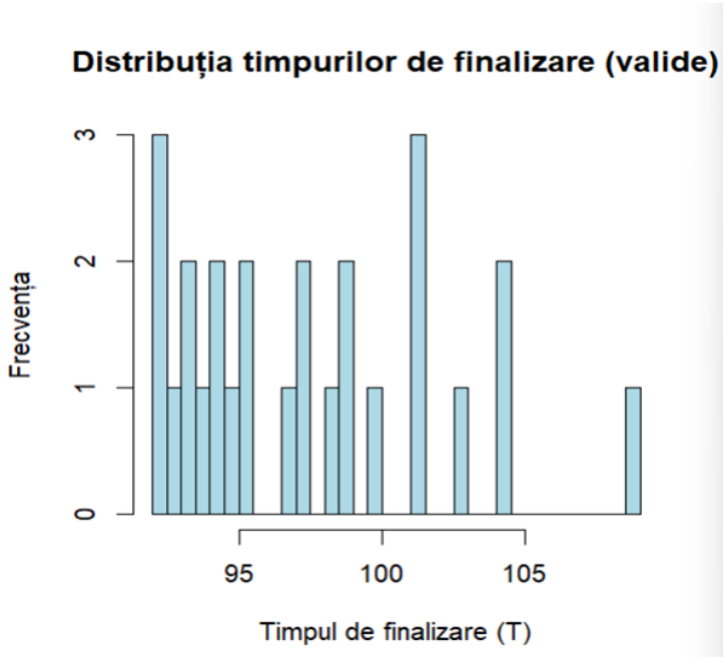
Am considerat variabilele `timpMin` și `timpMax` pentru valorile minime și maxime ale timpilor persoanelor care au finalizat evenimentul. Am folosit funcția `min` pentru minim și, ca parametru, am dat lista de timpi simulați, filtrată de valorile mai mari decât media timpului total. Am folosit din nou, pentru reprezentarea grafică, o histogramă, deoarece aceasta ajută la identificarea repartiției. Două observații sunt faptul că numărul de persoane care au reușit să finalizeze evenimentul este foarte mic în comparație cu numărul de simulări și că se observă că repartiția este uniformă, deoarece pentru alegerea valorilor random am folosit `runif`.

Rezultat:

```
timpMax = 108.54443085275
timpMin = 92.0171680316328
```

6. În baza simulărilor de la 1. aproximaţi probabilitatea ca persoana A să se oprească din lucru înainte de etapa k , unde 1<k≤n . Reprezentaţi grafic probabilităţile obţinute într-o manieră corespunzătoare. Ce puteţi spune despre repartiţia probabilităţilor obţinute?

```
k <- 5
PStopK <- mean(sapply(valoriT, function(x) {
if(x<=sum(1/lambda[1:k-1]))
    return (TRUE)
```




```
    else
        return(FALSE)
    }))
```

Explicatii:

Am creat o variabilă numită `PstopK` care primește probabilitatea ca persoana A să se oprească din lucru înainte de etapa k. Pentru această probabilitate am aplicat ca parametru unei funcții fiecare valoare simulată. Funcția returnează TRUE dacă timpul este mai mic decât media timpului total până la pasul k-1 (adică poate face maxim k-1 pași), și FALSE altfel. Variabila primește media valorilor din lista booleană.

Rezultat:

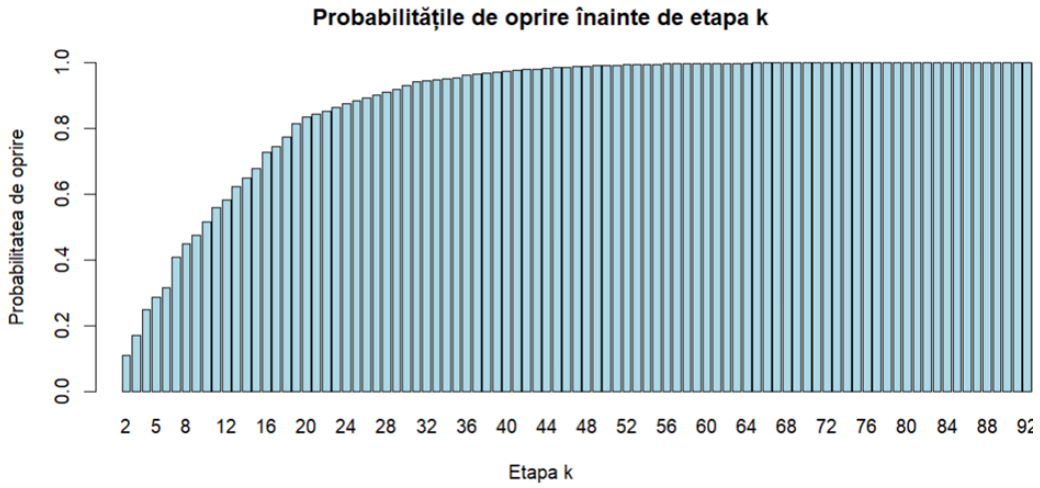
```
PstopK = 0.284293  pentru k=5
PstopK = 0.51482 pentru k=10
PstopK = 0.991332 pentru k=50
```

```
calculate_PStopBeforeK <- function(valoriT, lambda, alpha, n) {
# Calculam timpii cumulativi pentru fiecare etapa
cumulative_times <- cumsum(1 / lambda)

# Calculam probabilitatile pentru fiecare k
PStopBeforeK <- sapply(2:n, function(k) {
    mean(valoriT <= cumulative_times[k - 1])
            # Probabilitatea de oprire inainte de etapa k
})

return(PStopBeforeK)
}
PStopBeforeK <- calculate_PStopBeforeK(valoriT, lambda, alpha, n)

barplot(PStopBeforeK, names.arg = 2:n, col = "lightblue",
        main = "Probabilitatile de oprire inainte de etapa k",
        xlab = "Etapa k", ylab = "Probabilitatea de oprire",
        xlim = c(1, n), ylim = c(0, 1))
```



Observații:

Se poate observa că o persoană, cu cât ajunge mai departe în etape, cu atât are șanse mai mari să finalizeze evenimentul, iar în etapele inițiale șansele sunt foarte mici. Aceste observații sunt date datorită pantei abrupte în faza inițială, respectiv panta lină de la final.

Cerinta III

Descrierea Problemei

Scopul acestui proiect este de a construi o aplicaţie web interactivă folosind framework-ul **Shiny** din **R**, pentru a reprezenta grafic funcţiile de repartiţie (CDF) ale unor variabile aleatoare definite conform cerinţei din enunţ. Aplicaţia permite vizualizarea CDF-urilor pentru distribuţii normale, exponenţiale, binomiale, Poisson, precum şi combinaţii ale acestora.

Aspecte Teoretice

Pentru dezvoltarea aplicaţiei, am utilizat următoarele aspecte teoretice importante:

- **Distribuţia Normală**: Este definită ca o variabilă aleatoare continuă $X \sim N(\mu, \sigma^2)$, unde μ reprezintă media, iar σ^2 varianţa.
- **Distribuţia Exponenţială**: O variabilă aleatoare $X \sim \text{Exp}(\lambda)$ este folosită pentru modelarea timpilor de aşteptare între evenimente.
- **Distribuţia Poisson**: Reprezintă numărul de evenimente într-un interval fix de timp şi este notată $X \sim \text{Poisson}(\lambda)$.
- **Distribuţia Binomială**: Este definită de parametrii n şi p şi descrie numărul de succese în n experimente independente.

Reprezentări Grafice

Aplicaţia oferă reprezentări grafice ale funcţiilor de repartiţie pentru fiecare dintre variabilele aleatoare definite în cerinţe. În cele ce urmează, vom ilustra câteva capturi de ecran din aplicaţie.

Pachete Software Folosite

Am folosit următoarele pachete software pentru implementarea proiectului:

- **shiny**: Crearea aplicaţiei interactive.
- **bslib**: Personalizarea interfeţei grafice a aplicaţiei folosind teme Bootstrap.
- **graphics**: Generarea reprezentărilor grafice ale funcţiilor de repartiţie.

Codul Aplicaţiei

Mai jos este prezentat codul sursă al aplicaţiei, împreună cu comentarii relevante.

app.R

```r
library(shiny)
library(bslib)

source("utils.R")
source("ui.R")
source("server/server.R")

shinyApp(ui = ui, server = server)
```

ui.R

```r
# Generarea interfe\c{t}ei aplica\c{t}iei.
create_tab <- function(tab_title, title, distribution) {
  tabPanel(
    tab_title,
    div(
      class = "container",
      h1(title),
      div(
        class = "row",
        div(
          class = "col-4",
          tags$h3("Input:"),
          switch(
            distribution,
            BINOMIALA = create_binom_slider(),
            NORMALA_STANDARD = create_std_normal_slider(),
            NORMALA = create_normal_slider(),
            EXPONENTIALA = create_exponential_slider(),
            POISSON = create_pois_slider()
          )
        ),
        div(
          class = "col-8",
          h4("Reprezentare Grafic\u{a}"),
          get_output_distribution(distribution)
        )
      )
    )
  )
}
```

Dificultăţi Întâmpinate

- Ajustarea automată a parametrilor pentru fiecare distribuție.
- Reprezentarea grafică a unor transformări mai complexe.

Probleme Deschise

- Posibilitatea extinderii aplicației pentru a suporta și alte distribuții.
- Implementarea testelor statistice pentru variabilele simulate.

Concluzii

Aplicația dezvoltata oferă o platformă interactivă utilă pentru vizualizarea funcțiilor de repartiție ale variabilelor aleatoare și poate fi extinsă cu ușurință pentru a adăuga noi funcționalități și distribuții.