Savić Jovana 2020/3423

Elektrotehnički fakultet u Beogradu
Matematička statistika 13M081MAST
Prof. dr Milan Merkle
Domaći zadatak 4

Zadatak 1. Data je slučajna promenljiva X sa funkcijom raspodele

$$F(x) = \frac{1 - e^{-x/2}}{1 + e^{-x/2}}, \quad x > 0$$

a) Naći inverznu funkciju od F (izvesti formulu). Zatim objasniti kako biste simulirali slučajnu promenljivu X koristeći inverznu funkciju i uniformnu slučajnu promenljivu U.

b) Simulirajte $n = 1000$ vrednosti za X i prikažite empirijsku funkciju raspodele (nazovimo je F_n) zajedno sa funkcijom F na istoj slici.

c) Primenom testa Kolmogorova-Smirnova testirati hipotezu da je $F_n = F$ (možete koristeći softver po izboru).

Rešenje.

a) Datu funkciju možemo da napišemo kao

$$F(x) = \frac{1 - e^{-x/2}}{1 + e^{-x/2}} = \frac{e^{x/2} - 1}{e^{x/2} + 1} = \tanh \frac{x}{4}$$

Odavde lako nalazimo

$$x = 4 \tanh^{-1} F(x)$$

Prema tome, inverzna funkcija je:

$$F^{-1}(x) = 4 \tanh^{-1}(x) = 2 \log \frac{1 + x}{1 - x}, \quad x \in [0, 1)$$

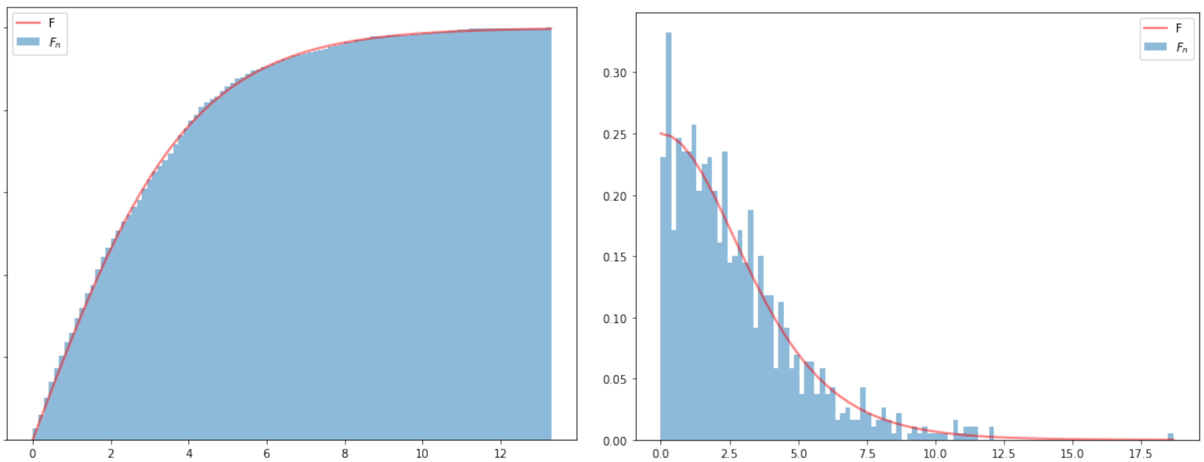
gde deo domena $x \in (-1, 0)$ koji bismo imali za inverzni hiperbolički tanges ne koristimo jer on on odgovara domenu $x < 0$, koji ne spada u domen naše originalne funkcije.

Genersanje slučajne promenljive čija je funkcija raspodele F monotono rastuća i neprekidna, onda kada je na raspolaganju uniformna slučajna promenljiva U na $[0, 1]$, je dato u primeru 78 koji navodimo u nastavku. Neka je $Y = F^{-1}(U)$. Tada je

$$F_Y(y) = P(Y \leq y) = P(F^{-1}(U) \leq y) = P(U \leq F(y)) = F(y)$$

jer je $P(U \leq t) = t$ za $t \in [0, 1]$. Prema tome, ukoliko na generisanu slučajnu promenljivu U primenimo F^{-1} dobijamo slučajnu promenljivu sa željenom raspodelom F.

b) Simulacija je urađena u programskom jeziku Python. Na slici 1 je prikazana emprijska i stvarna funkcija raspodele. Iako je naš domen $x > 0$ retko se dešava da dobijemo vrednosti koje su veće od 20. Razlog je jasan ako primetimo da je, na primer, $F^{-1}(0.9998) = 18.47$.



(a) Funkcija raspodele (b) Funkcija gustine raspodele

Slika 1: Empirijske i stvarne funkcije raspodele i funkcije gustine raspodele

Naša generisana funkcija raspodele će imati manji domen nego stvarna funkcija raspodele jer je i broj decimala promenljive U ograničen. Za broj decimala korišćen u ovoj implementaciji, maksimalna vrednost koju slučajna promenljiva X može da dosigne je 75.

Iako nam način koji je opisan omogućava da simuliramo raspodelu F na osnovu uniformne slučajne promenljive, činjenica da na računaru radimo sa diskretnim vrednostima uvodi neka ograničenja. Iako veće vrednosti slučajne promenljive X imaju jako male verovatnoće, one su ipak veće od nule, dok su u našoj simulaciji to nemogući događaji. Ipak, u praksi ovo ne predstavlja problem.

c) *(Test Kolmogorova-Smirnova)* Test hipoteze $H_0 \; : \; F = F_0$ protiv $H_1 \; : \; F \neq F_0$ sa nivoom značajnosti α. Ako je

$$\lambda = \sqrt{n} \sup_{x \in R} |F_n(x) - F_0(x)| > \varepsilon_{1-\alpha}$$

hipotezu H_0 odbacujemo.

Maksimalna razlika koja se dobije je 0.019, a njoj odgovara

$$\lambda = 0.593$$

P-vrednost je $P(K > 0.593)$. Ova vrednost nije data u tabeli 7, ali vidimo da će ova verovatnoća biti veća od $P(K > 1) = 0.27$, odakle zaključujemo da hipotezu ne odbacujemo. □

```
def F(x):
    return np.tanh(x / 4)

def F_inv(x):
    return 4 * np.arctanh(x)

def f(x):
    return 1 / (4 * np.cosh(x / 4) * np.cosh(x / 4))

U = np.random.uniform(0, 1, (1000, 1))
X = F_inv(U)

plt.figure(figsize = (9,7))
myHist = plt.hist(X, 100, cumulative=True,
                  density=True, alpha=0.5,
                  label="$F_n$")
x = np.linspace(0,np.max(X))
h = plt.plot(x, F(x), lw=2, color = "red",
             alpha=0.5, label="F")
legend=plt.legend(loc="best")

F_n = myHist[0]
x = myHist[1]
F_o = F(x)

diff = np.max(np.abs(F_o[1:] - F_n))
print("Maksimalna razlika = ", diff)
lambd =  np.sqrt(1000) * diff
print("Lambda = ", lambd)
```

Program 1: Program kojim je urađena simulacija u zadatku 1

Zadatak 2. Predmet ovog zadatka je Beta raspodela $B(\alpha, \beta)$, sa gustinom

$$f(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}, \quad x \in (0,1)$$

sa $\alpha > 1$, $\beta > 1$. U primeru 115 možete naći definiciju i osobine Gama funkcije. U primeru 213 u udžbeniku je pokazano da se ova raspodela može simulirati metodom odbacivanja.

a) Za α i β izaberite prirodne brojeve (radi jednostavnijeg izračunavanja Gama funkcije). Simulirati $n = 1000$ tačaka primenom metoda odbacivanja, pri čemu za svaku od 1000 vrednosti treba sačuvati broj ponavljanja do prihvatanja.

b) Prema teoriji (zadatak 222), prosečni broj iteracija jednak je c, gde je c konstanta iz algoritma metoda odbacivanja. Koristeći se podacima sačuvanih pod a) proveriti da li prosečan broj iteracija odgovara teoriji i komentarisatiti.

Rešenje. **a)** *(Metod odbacivanja za generisanje raspodele sa gustinom f)* Neka je Y slučajna promenljiva sa gustinom g i neka je f funkcija raspodele koju treba generisati, takva da su f i g koncentrisane na istom skupu D i da je $f(y) \le cg(y)$ za svako $y \in D$ i za neku pozitivnu konstantu c. Metod odbacivanja se sastoji od dva koraka.

1. Generisati Y sa gustinom g i generisati slučajan broj U.

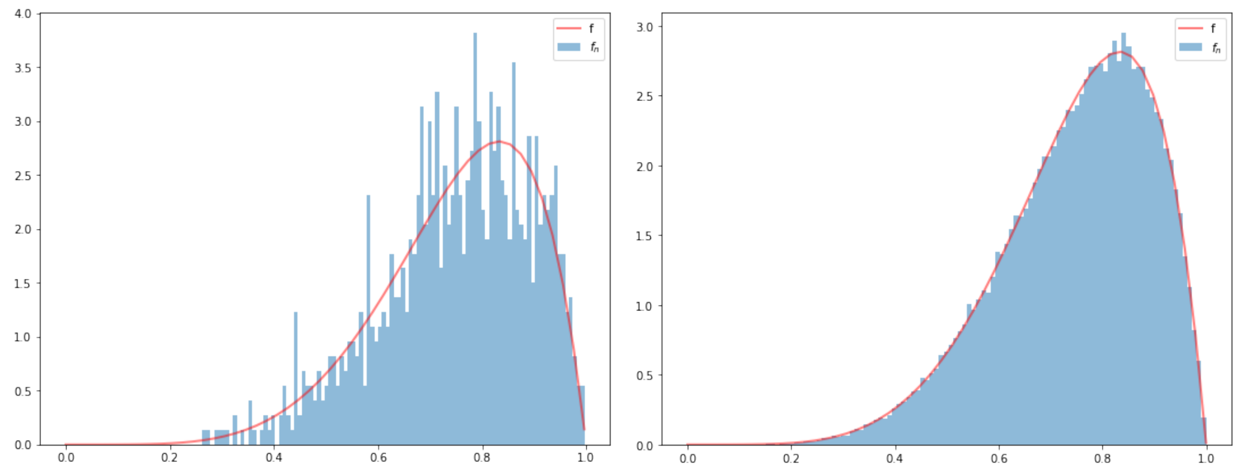
2. Ako je $U \leq f(Y)/cg(Y)$ staviti $X = Y$. U protivnom ponoviti korak 1.

Generisanje beta raspodele metodom odbacivanja, opisano u primeru 213, navodimo u nastavku. Ukoliko izaberemo $c = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}$, a g predstavlja gustinu uniformne slučajne promenljive na $(0,1)$ važiće $f(x) \leq cg(x)$. Prema tome, na osnovu metoda odbacivanja imamo sledeći algoritam:

1. Generisati nezavisne U_1 i U_2.

2. Ako je $U_2 < U_1^{\alpha-1}(1-U_1)^{\beta-1}$, staviti $X = U_1$. U protivnom, ponoviti korak 1.

Simulacija je urađena u programskom jeziku Python. Izabrane su vrednosti $\alpha = 6$ i $\beta = 2$ za koje se gama funkcija svodi na faktorijel. Na slici 2 su prikazane dobijene empirijske funkcije gustine verovatnoće za $n = 1000$ i $n = 100000$.



(a) $n = 1000$ (b) $n = 100000$

Slika 2: Empirijska i stvarna funkcija gustine verovatnoće beta raspodele sa parametrima $\alpha = 6$ i $\beta = 2$

b) Kada koristimo metod odbacivanja, verovatnoća da dodelmo vrednost Y slučajnoj promenljivoj X je

$$P\left(U \leq \frac{f(Y)}{cg(Y)}\right) = \int_D \int_0^{\frac{f(Y)}{cg(Y)}} g(y) \mathrm{d}u \mathrm{d}y = \int_D \frac{f(y)}{c} \mathrm{d}y = \frac{1}{c}$$

Generisanje slučajne promenljive se dešava sa verovatnoćom $1/c$. Broj iteracija koji je potreban ima geometrijsku raspodelu, jer se u ovom slučaju ponavljaju Bernulijevi eksperimenti sa verovatnoćom $1/c$ do prvog uspeha. Matematičko očekivanje je u ovom slučaju c, pa je ovo i prosečan broj iteracija koji je potreban za generisanje jednog odbirka.

U simulaciji su dobijeni rezultati koji su u skladu sa teorijom. Za $n = 1000$, prosečan broj iteracija je 42.956, a za $n = 100000$ iznosi 42.055. Za izabrane vrednosti $\alpha = 6$ i $\beta = 2$ imamo $c = 42$.

$\square$

```
def generate_sample(alpha, beta):
    c = 0
    while True:
        U1= np.random.uniform(0, 1)
        U2 = np.random.uniform(0,1)
        c += 1
        if U2 < (U1 ** (alpha-1)) * ((1-U1)**(beta-1)):
            return U1, c

import math
def f(x, alpha, beta):
    c = math.factorial(alpha+beta-1) /
           (math.factorial(alpha - 1) * math.factorial(beta -1))
    return c * np.power(x, alpha-1) * np.power(1-x, beta-1)

alpha, beta = 6, 2
n = 1000
X = np.zeros((n, 1))
c = np.zeros((n, 1))

for i in range(n):
    X[i], c[i] = generate_sample(alpha, beta)

plt.figure(figsize = (9,7))
myHist = plt.hist(X, 100, density=True,
                     alpha=0.5, label="$f_n$")
x = np.linspace(0,np.max(X))
h = plt.plot(x, f(x, alpha, beta), lw=2,
                color = "red", alpha=0.5, label="f")
legend=plt.legend(loc="best")

print("Prosecan broj iteracija = ", np.mean(c))
```

Program 2: Program kojim je urađena simulacija u zadatku 2