Curs 8: Simularea distribuţiilor de probabilitate discrete şi continue

1.1 Histograma observaţiilor asupra unei variabile aleatoare

Pentru a evidenţia aspectele practice legat de variabilele aleatoare discrete/continue, considerăm graficul unei densităţi de probabilitate şi divizăm domeniul său de definiţie prin puncte echidistante x_i, cu pasul h fixat. Probabilitatea ca variabila aleatoare X să ia valori în intervalul $[x_i, x_{i+1})$ este aria trapezului curbiliniu de baze segmentul $[x_i, x_{i+1})$ şi arcul de grafic deasupra acestui segment. Aria trapezului o putem aproxima cu aria dreptunghiului de bază $[x_i, x_{i+1})$ şi înălţime $f((x_i + x_{i+1})/2)$ (Fig.1, sus), fie cu aria dreptunghiului de bază $[x_i, x_{i+1})$ şi înălţime $f(x_i)$ (Fig.1, jos). Deci $P(x_i \leq X < x_{i+1}) = \int_{x_i}^{x_{i+1}} f(x)\,dx$=aria trapezului curbiliniu, este aproximativ aria dreptunghiului menţionat. Reuniunea tuturor dreptunghiurilor astfel construite se numeşte histogramă asociată densităţii de probabilitate. Histograma evidenţiază aproximativ distribuţia valorilor variabilei aleatoare în intervalele $[x_i, x_{i+1})$.

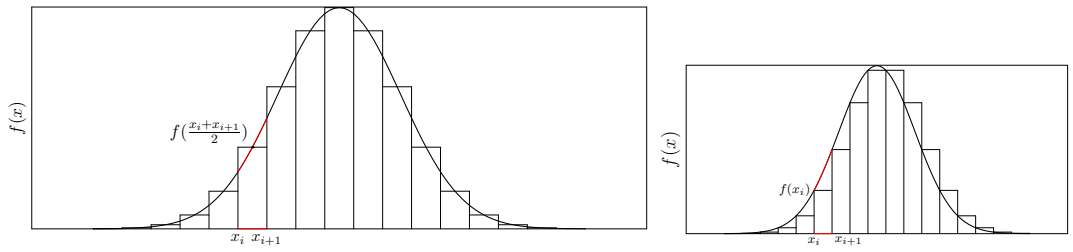


Fig.1: Histograma asociată unei densităţi de probabilitate

Precizarea distribuţiei de probabilitate a unei variabile aleatoare continue, se bazează de obicei pe istoricul observaţiilor asupra valorilor sale. În experimentele de laborator sau experimentele virtuale, de simulare, se înregistrează valorile observate, măsurate sau generate, ale unei variabile aleatoare, adică o listă de numere reale $x_1, x_2, \ldots, x_N$. Cel mai adesea nu se cunoaşte distribuţia de probabilitate a variabilei studiate. Informaţia primară se obţine asociind seriei de date înregistrate o histogramă cu m bare, m fixat apriori. Şi anume:

- se determină valoarea minimă, $xmin$, şi valoarea maximă, $xmax$, a seriei de date.

• Se divide segmentul $[xmin, xmax]$ prin puncte echidistante cu pasul, $h = \dfrac{xmax - xmin}{m}$. Notăm cu y_j punctele de diviziune, $y_j = xmin + j * h$, $j = \overline{0, m}$.

• Se calculează numărul de valori, n_j, ale seriei de date, care aparţin intervalului

$$I_j = [y_j, y_{j+1}),\ j = \overline{0, m-2},$$
$$\text{respectiv intervalului } I_{m-1} = [y_{m-1}, xmax]$$

• Deoarece din cele N valori ale seriei de date, n_j cad în intervalul I_j, rezultă că am obţinut informaţia că probabilitatea ca variabila observată să ia valori în intervalul I_j este aproximativ:

$$P(y_j \leq X < y_{j+1}) \approx \frac{n_j}{N}$$

Dar această probabilitate este comparând cu cazul teoretic de mai sus aria dreptunghiului (a barei) ce are baza segmentul $[y_j, y_{j+1})$ • Rezultă astfel că deasupra intervalului I_j desenăm un dreptunghi de arie $A = n_j/N$. Dar aria este baza ori înalţimea dreptunghiului, H_j, şi deci din $A = Baza \times H_j$, rezultă că înălţimea dreptunghiului (a barei) este $H_j = A/B$. Cunoscând lungimea bazei ca fiind $h = y_{j+1} - y_j$, rezultă că înalţimea dreptunghiului este $H_j = \dfrac{n_j}{h\,N}$, $j = \overline{0, m-1}$.

Obiectul grafic rezultat din desenarea celor m dreptunghiuri (bare) se numeşte *histograma seriei de date* sau *histograma distribuţiei de frecvenţe*, deoarece ea ilustrează modul în care datele sunt distribuite în intervalele $[y_j, y_{j+1})$.

În Fig.2 sunt ilustrate histogramele a două serii de date.

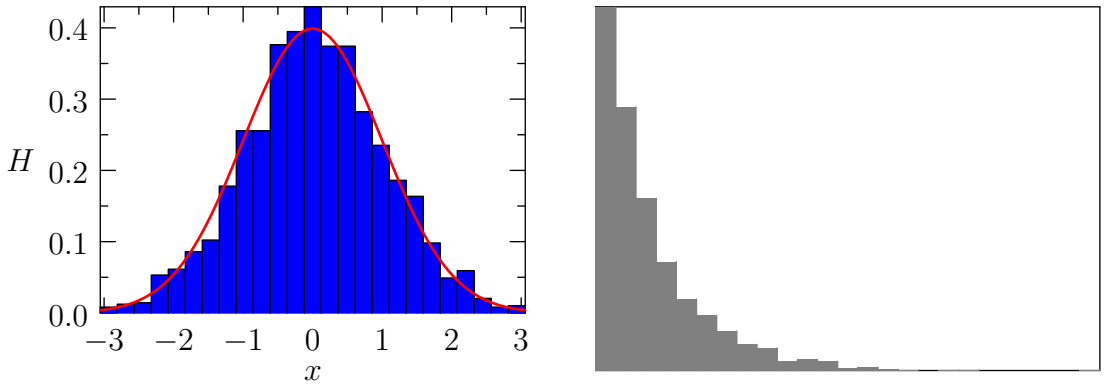


Fig.2: Histograme asociate la două serii de date numerice, rezultate din simularea unor variabile aleatoare ce au densitatea ilustrată în roşu.

1.2 Simularea variabilelor aleatoare variabilelor aleatoare

A simula o variabilă aleatoare discretă X, presupune a genera independent, conform unui algoritm, un şir de numere $y_1, y_2, \ldots, y_N$, care să aibă particularităţile unor valori de observaţie asupra variabilei. Mai precis dacă variabila X are distribuţia de probabilitate:

$$X = \begin{pmatrix} x_1 & x_2 & \dots & x_n \\ p_1 & p_2 & \dots & p_n \end{pmatrix}$$

atunci distribuţia experimentală a datelor generate trebuie să fie foarte apropiată de distribuţia teoretică a variabilei X. Si anume, dacă notăm cu nr_k numărul valorilor generate, egale cu x_k, $k = \overline{1,n}$, atunci $\frac{nr_k}{N}$ trebuie să fie apropiat de p_k.

În continuare prezentăm metode de simulare a variabilelor aleatoare ce au diverse distribuţii de probabilitate. Pentru fiecare distribuţie de probabilitate dăm pseudocodul pentru o funcţie ce returnează o singură valoare de observaţie simulată. O astfel de valoare se obţine printr-o transformare sau mai multe, aplicată unei valori `u` presupusă a fi returnată de un generator de numere pseudo–aleatoare uniform distribuite pe $[0, 1)$.

Apelul acestui generator îl simbolizăm în continuare prin `urand()`. ATENŢIE, NU există nici un limbaj de programare care să pună la dispoziţie o funcţie, numită `urand()`.Acesta este un nume generic, dat de noi, pt simulatorul unei variabile aleatoare $U \sim$ Unif$[0, 1)$.

In prezentarea şi argumentarea algoritmilor ce urmează exploatăm următoarea proprietate a unei variabile aleatoare $U \sim Unif[0, 1)$ (vezi Cursul Sap 8):

> Dacă $U \sim$ Unif$[0, 1)$, atunci pentru orice interval $(a, b] \subset [0, 1)$, probabilitatea ca U să ia valori în acest interval este egală cu lungimea intervalului: $P(U \in (a, b]) = b-a$.

Dacă A este un eveniment de probabilitate $P(A) = 0.65$ şi $B = \complement A$, de probabilitate $P(B) = 0.35$, atunci putem simula producerea unuia din cele două evenimente astfel:

• apelăm generatorul `urand()`:

```
u=urand();
```

• Dacă $u < 0.65$, spunem că s-a produs evenimentul $(U < 0.65)$, care are probabilitatea $P(U < 0.65) = P(U \in [0, 0.65)) = 0.65$, adică lungimea intervalului. Cum şi A are aceeaşi probabilitate putem considera că s-a produs evenimentul A.

• Dacă $u \in [0.65, 1)$, atunci s-a produs evenimentul $(U \in [0.65, 1))$ a cărui probabilitate este $P(U \in [0.65, 1)) = 1 - 0.65 = 0.35$. Deci în acest caz putem considera că s-a produs evenimentul B.

Această modalitate de a genera evenimente se foloseşte foarte mult în algoritmii randomizaţi `http://en.wikipedia.org/wiki/Randomized_algorithm`.

1.3 Simularea distribuţiei uniforme discrete

Propoziţia 1.3.1 *Dacă $U \sim$Unif[0,1) este o variabilă aleatoare uniform distribuită pe $[0, 1)$ şi n este număr întreg, $n > 1$, atunci variabila aleatoare $X = [nU]$ ($[x]$ notează partea întreagă a lui x) este o variabilă aleatoare discretă ce are distribuţia uniformă pe*

mulţimea $\{0, 1, 2, \ldots, n-1\}$*, adică:*

$$X = \begin{pmatrix} 0 & 1 & \ldots & n-1 \\ \frac{1}{n} & \frac{1}{n} & \ldots & \frac{1}{n} \end{pmatrix}$$

Demonstraţie: Deoarece variabila aleatoare U ia valori în $[0, 1)$, variabila nU ia valori în $[0, n)$ şi $[nU] \in \{0, 1, 2, \ldots, n-1\}$. Deci variabila $X = [nU]$ are ca mulţime de valori $\{0, 1, 2, \ldots, n-1\}$.

Să arătăm că probabilitatea ca X să ia valoarea k, $k = \overline{1, n}$, este $P(X = k) = \frac{1}{n}$:

$$\begin{aligned} P(X = k) &= P([nU] = k) = P([nU] = k) = P(k \leq nU < k+1) \\ &= P\left(\frac{k}{n} \leq U < \frac{k+1}{n}\right) = P\left(U \in \left[\frac{k}{n}, \frac{k+1}{n}\right)\right) = \text{lungimea interv} = \\ &= \frac{k+1}{n} - \frac{k}{n} = \frac{1}{n} \end{aligned}$$

Deci $X = [nU]$ este o variabilă aleatoare uniform distribuită pe mulţimea $\{0, 1, 2, 3, \ldots, n-1\}$ □

Exploatând această relaţie dintre variabila aleatoare discretă X şi variabila aleatoare U avem următorul algoritm ce returnează o valoare de observaţie asupra variabilei X pe mulţimea $\{0, 1, 2, \ldots, n-1\}$:

```
1: function SimDiscretU(n)
2:     u=urand();
3:     k=int(n*u);
4:     return k;
5: end function
```

Inlocuind `return k;` cu return x_k; se returnează o valoare de observaţie asupra unei variabile aleatoare discrete, uniform distribuită pe o mulţime $\{x_0, x_1, \ldots, x_{n-1}\}$, adică asupra variabilei aleatoare:

$$X = \begin{pmatrix} x_0 & x_1 & \ldots & x_{n-1} \\ \frac{1}{n} & \frac{1}{n} & \ldots & \frac{1}{n} \end{pmatrix}$$

Simularea acestei variabile aleatoare este echivalentă cu simularea experimentului de extragere la întâmplare a unui element (obiect) din lista $x_0, x_1, \ldots, x_{n-1}$ şi returnarea lui în "recipientul" din care a fost scos. La întâmplare înseamnă că fiecare element are aceeaşi şansă de a fi extras, adică probabilitatea $1/n$.

Algoritm ce extrage un număr la întâmplare din mulţimea de numere întregi $\{m, m+1, \ldots, n\}$**,** $m < n$.

Mulţimea conţine $N = n - m + 1$ elemente. Un număr selectat la întâmplare din această mulţime este o valoare de observaţie asupra variabilei aleatoare:

$$X = \begin{pmatrix} m & m+1 & \dots & n \\ \dfrac{1}{n-m+1} & \dfrac{1}{n-m+1} & \dots & \dfrac{1}{n-m+1} \end{pmatrix}.$$

```
1: function randint(m,n)
2:     u=urand();
3:     k=int((n-m+1)*u));//k in {0,1,2,...,n-m}
4:     return k+m;
5: end function
```

Aplicaţie: Generarea unei permutari aleatoare

Reamintim că o permutare a mulţimii $\{1, 2, \dots, n\}$ este o bijecţie:

$$\pi : \{1, 2, \dots, n\} \to \{1, 2, \dots, n\}$$

Există $n!$ permutari ale mulţimii ordonate $(1, 2, \dots, n)$. Dacă le notăm $\pi_1, \pi_2, \dots, \pi_{n!}$ toate permutările posibile, atunci o permutare aleatoare este o observaţie asupra variabilei aleatoare:

$$X = \begin{pmatrix} \pi_1 & \pi_2 & \dots & \pi_{n!} \\ \dfrac{1}{n!} & \dfrac{1}{n!} & \dots & \dfrac{1}{n!} \end{pmatrix},$$

ce are distribuţia uniformă pe mulţimea tuturor pemutărilor.

Un algoritm de generare a unei permutări aleatoare a elementelor $a_0, a_1, \dots, a_{n-1}$, ale unui tablou a, generează oricare din cele $n!$ permutări posibile cu probabilitatea $\dfrac{1}{n!}$. În loc să calculăm toate permutarile şi să simulăm variabila discretă X (abordare ineficientă!), dăm un continuare un algoritm, numit *FisherYates* ce generează secîvenţial elementele din poziţiile $(1, 2, \dots, n)$ ale permutării aleatoare.

Algoritmul *FisherYates* este următorul:

```
1: function PermAleat(a, n);
2:     for k = 0 : n − 2
3:         j=randint(k, n-1);
4:         a_j ↔ a_k; //interschimba a_j cu a_k
5:     end for
6:     return a;
7: end function
```

Se spune că algoritmul *Fisher-Yates* este un algoritm *in place*, deoarece returnează permutarea realizată în array-ul iniţial.

Detaliind observăm că pentru:

• $k = 0$, $a_0 \leftrightarrow a_{j_0}$, unde j_0 este generat aleator şi uniform cu probabilitatea $1/n$, din mulţimea de indici $\{0, 1, 2 \dots, n-1\}$; după această etapă în locaţiile $a_1, a_2, \dots, a_{n-1}$ nu va mai ajunge a_{j_0} pentru ca nu se va mai acţiona asupra poziţiei 0 din array.

• $k = 1$, $a_1 \leftrightarrow a_{j_1}$, cu j_1 generat aleator şi uniform cu probabilitatea $1/(n-1)$, din mulţimea de indici $\{1, 2, \ldots, n-1\}$; Analog, după această etapă în poziţiile $a_2, a_3, \ldots, a_{n-1}$ nu va mai fi distribuit a_{j_1} pentru ca nu se va mai acţiona asupra poziţiei 1 din array.

• etc

• pentru $k = n-2$, $a_{n-2} \leftrightarrow a_{j_{n-2}}$, cu j_{n-2} generat cu probabilitatea $1/2$, din mulţimea de indici $\{n-2, n-1\}$;

• in bucla `for` nu s-a considerat şi $k = n-1$, pentru ca alegerea uniformă dintr-o mulţime cu 1 element se face cu probabilitatea 1 şi deci $a_{n-1} \leftrightarrow a_{n-1}$.

1.4 Simularea variabilelor aleatoare discrete având o distribuţie generală, neuniformă

Fie X o variabilă discretă ce are distribuţia de probabilitate

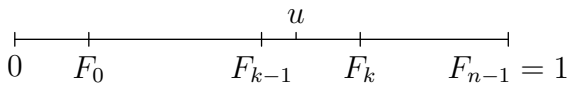
$$X = \begin{pmatrix} x_0 & x_1 & \ldots & x_{n-1} \\ p_0 & p_1 & \ldots & p_{n-1} \end{pmatrix}, \quad \sum_{k=0}^{n-1} p_k = 1$$

şi valorile sale sunt ordonate, adică $x_0 < x_1 < \cdots < x_{n-1}$.

Pentru simularea variabilei X, folosim din nou generatorul de numere pseudo–aleatoare uniform distribuite pe $[0, 1)$, şi faptul că dacă $U \sim$ Unif$[0, 1)$, atunci pentru orice interval $(a, b] \subset [0, 1)$, probabilitatea ca U să ia valori în acest interval este egală cu lungimea intervalului: $P(U \in (a, b]) = b - a$.

Ţinând seama că $\sum_{k=0}^{n-1} p_k = 1$, divizăm intervalul $[0, 1)$, prin punctele

$$F_0 = p_0, F_1 = p_0 + p_1, \ldots, F_k = p_0 + p_1 + \cdots + p_k, \ldots,$$
$$F_{n-1} = 1$$



Probabilitatea ca un număr u, valoare de observaţie asupra variabilei $U \sim$Unif[0,1), să cadă în intervalele de forma $(F_{k-1}, F_k]$, $k = 0, 1, 2, \ldots, n-1$, este $P(U \in (F_{k-1}, F_k]) = \underbrace{F_k - F_{k-1}}_{lung.interv.} = p_k = P(X = x_k)$ (am considerat $F_{-1} = 0$).

Rezultă că evenimentul ca X să ia valoarea x_k, $(X = x_k)$, se produce cu aceeaşi probabilitate ca şi evenimentul ca U să ia valori în intervalul $(F_{k-1}, F_k]$, $(U \in (F_{k-1}, F_k])$. Deci generând u în intervalul $(F_{k-1}, F_k]$, este echivalent cu producerea evenimentului $(X = x_k)$ şi în acest caz generatorullui X returnează elementul x_k.

Astfel algoritmul de generare a unui număr pseudo–aleator din legea de probabilitate a variabilei discrete X ce ia valorile x_k cu probabilităţile p_k, $k = 0, \ldots, n-1$ este:

```
1: function SimDiscret(x, p, n)//x, p sunt tablouri de n elemente
2:     k = 0;
3:     F = p_0;
4:     u=urand();
5:     while (u > F){
6:     k = k + 1;
7:     F = F + p_k;
8:     }
9:     return x_k;
10: end function
```

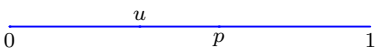
Această metodă de căutare secvenţială a intervalului în care cade numărul u este recomandabilă doar pentru variabilele aleatoare care au un număr redus de valori $x_0, x_1, \ldots, x_{n-1}$.

Pentru variabilele aleatoare discrete, X, cu număr mare de valori se recomandă calcularea prealabilă a sumelor $F_k = p_0 + p_1 + \cdots + p_k$, $k = \overline{0, n-1}$ şi căutarea binară a intervalului $(F_{k-1}, F_k]$ în care cade numărul $u \in [0, 1)$, returnat de `urand()`.

Un caz particular de distribuţie discretă este variabila Bernoulli:

$$X = \begin{pmatrix} 1 & 0 \\ p & 1-p \end{pmatrix}$$

Ea se simulează când avem de făcut o alegere dintre două alternative, codificate cu 1 şi 0.



Şi anume se generează $u \in [0, 1)$, apelând `u = urand()`;

Dacă $u \leq p$, înseamnă că s-a produs evenimentul $(U \leq p)$ a cărui probabilitate este $P(U \leq p) = P(0 \leq U \leq p) = p - 0 = p$. Dar cum şi evenimentul $(X = 1)$ are probabilitatea p, putem presupune că s-a produs acesta şi deci facem alegerea codificată cu 1. Dacă însă $u > p$, atunci facel alegerea codificată de 0.

Avem deci algoritmul următor de simulare a unei variabile aleatoare Bernoulli:

```
1: function Bernoulli(p);
2:     u=urand();
3:     if(u < p) return 1;
4:     else return 0;
5: end function
```

Acest algoritm se mai poate aplica la generarea unui şir de biţi aleatori, la căutarea aleatoare într-un arbore binar, etc.

1.5 Simularea variabilelor aleatoare binomiale

Fie X o variabilă aleatoare binomială ce dă numărul de succese în n încercări Bernoulli şi probabilitatea succesului în orice încercare este p. Probabilitatea ca în cele n încercări să înregistrăm k succese, $k = 0, 1, 2, \ldots, n$ este:

$$Pr_k = P(X = k) = C_n^k p^k (1-p)^{n-k}$$

Variabila $X \sim Bin(n, p)$ este o variabilă aleatoare discretă neuniform distribuită:

$$X = \begin{pmatrix} 0 & 1 & \ldots & k & \ldots & n \\ Pr_0 & Pr_1 & \ldots & Pr_k & \ldots & Pr_n \end{pmatrix}$$

şi deci se simulează conform algoritmului prezentat mai sus. Singura problemă este calculul probabilităţilor Pr_k şi a sumelor $F_k = Pr_0 + Pr_1 + \cdots + Pr_k$, $k = \overline{0, n}$

Pentru a evita calculul combinărilor conform formulei învăţate în algebra de liceu, deducem o formulă recursivă şi anume:

$$C_n^{k+1} = \frac{n(n-1)\cdots(n-k+1)(n-k)}{1 \cdot 2 \cdots k \cdot (k+1)} = C_n^k \frac{n-k}{k+1}$$

Folosind această formulă recursivă deducem acum o formulă de calcul recursiv a probabilităţilor Pr_k:

$$Pr_{k+1} = P(X = k+1) = C_n^{k+1} p^{k+1} (1-p)^{n-k-1} = C_n^k \frac{n-k}{k+1} \frac{p}{1-p} p^k (1-p)^{n-k} = \frac{n-k}{k+1} \frac{p}{1-p} P(X = k) \tag{1}$$

Notând cu $c = \frac{p}{1-p}$ şi avem următoarea formulă recursivă de calcul a probabilităţilor valorilor distribuţiei binomiale:

$$Pr_{k+1} = \frac{n-k}{k+1} \, c \, Pr_k, \quad Pr_0 = (1-p)^n$$

Cu această precizare, simularea unei variabile aleatoare, X, ce are distribuţia binomială se realizează folosind algoritmul de simulare a unei variabile aleatoare discrete. Şi anume:

1: **function Bin**(n, p)
2: $k = 0$;
3: $\mathtt{c} = \mathtt{p}/(1 - \mathtt{p})$;
4: $pr = (1-p)^n$;
5: $F = pr$;
6: **u=urand();**
7: **while** $(u > F)\{$
8: $pr = (c * (n-k)/(k+1)) * pr$
9: $F = F + pr$;

10: $k = k + 1$**;**
11: }
12: **return** k**;**
13: **end function**

Dacă parametrul n al distribuţiei binomiale este foarte mare, atunci căutarea liniară nu este performantă. În acest caz se recomandă precalcularea probabilităţilor $Pr_k = P(X = k)$, $k = \overline{0, n}$, folosind formula de recurenţă dedusă, precum şi a sumelor $F_j = \sum_{j=0}^{k} pr_j$ şi folosirea căutării binare.

1.6 Simularea distribuţiei geometrice

Fie X o variabilă aleatoare ce are distribuţia geometrică de parametru $p \in (0, 1)$, $X \sim \text{Geom}(p)$. X dă numărul de încercări într-un experiment Bernoulli pâna la primul succes înregistrat, inclusiv.

Distribuţia de probabilitate este ilustrată în tabloul:

$$X = \begin{pmatrix} k \\ p(1-p)^{k-1} \end{pmatrix}, \quad k \in \mathbb{N} \setminus \{0\}$$

Metoda directă de simulare, exploatând faptul ca o variabilă geometrică este asociată unui experiment Bernoulli, este următoarea:

1: **function Geom1(**p**)**
2: **k=0; // k contorul pentru incercarile Bernoulli**
3: **do** {
4: **u=urand()**
5: **k=k+1;**
6: }**while(**$u > p$**);**
7: **return k**
8: **end function**

Se execută blocul de instrucţiuni din bucla `do-while` atâta timp cât încercările sunt un eşec. La primul succes returnează numărul încercării respective. Dacă probabilitatea succesului p este mică numărul mediu de încercări până la primul succes este mare pentru că $M(X) = 1/p$ şi în acest caz algoritmul `Geom1` este ineficient.

O modalitate mai rapidă de simulare a variabilei aleatoare X rezultă din următoarea propoziţie:

Propoziţia 1.6.1 *Dacă* $p \in (0, 1)$ *şi* $U \sim Unif[0, 1)$*, atunci variabila aleatoare:*

$$X = \left[\frac{\ln(1-U)}{\ln(1-p)} \right] + 1 \tag{2}$$

are distribuţia geometrică de parametru p ([ ] *notează funcţia parte întreagă*).

Demonstraţie: Notăm cu Y variabila aleatoare $\left[\frac{\ln(1-U)}{\ln(1-p)}\right]$. Fie u o valoare de observaţie asupra variabilei aleatoare U, adică $u \in [0,1)$ este generat de `urand()`. Dacă $u = 0$, atunci $Y = 0$. În caz contrar, Y este un întreg pozitiv. Deci Y este o variabilă aleatoare discretă ce ia valori în $\mathbb{N}$. În continuare arătăm că probabilitatea evenimentului $(Y = k)$ coincide cu probabiltatea unui eveniment E, asociat variabilei aleatoare U, uniform distribuite, pe care în plus îl putem şi simula:

$$P(Y = k) = P\left(\left[\frac{\ln(1-U)}{\ln(1-p)}\right] = k\right)$$

Dar cum partea întreagă $[x] = k$ implică $k \leq x < k+1$, avem că:

$$P(Y = k) = P(k \leq \frac{\ln(1-U)}{\ln(1-p)} < k+1)$$

Deoarece $\ln(1-p) < 0$ rezultă:

$$\begin{aligned} P(Y=k) &= P((k+1)\ln(1-p) < \ln(1-U) \leq k\ln(1-p)) \\ &= P(\ln(1-p)^{k+1} < \ln(1-U) \leq \ln(1-p)^k) \\ &= P((1-p)^{k+1} < 1-U \leq (1-p)^k) \\ &= P(-(1-p)^k \leq U-1 < -(1-p)^{k+1}) \\ &= P(1-(1-p)^k \leq U < 1-(1-p)^{k+1}) \end{aligned}$$

U fiind uniform distribuit pe $[0,1)$, avem că:

$$P(1-(1-p)^k \leq U < 1-(1-p)^{k+1}) = 1-(1-p)^{k+1}-(1-(1-p)^k) = (1-p)^k p$$

Variabila aleatoare $X = Y + 1$ şi:

$$P(X = k) = P(Y + 1 = k) = P(Y = k-1) = (1-p)^{k-1}p$$

Deci variabila aleatoare X are distribuţia geometrică. □

Ţinând seama că pentru variabila aleatoare $X \sim \text{Geom}(p)$, evenimentul $(X = k)$ are aceeaşi probabilitate ca evenimentul $\left(\left[\frac{\ln(1-U)}{\ln(1-p)}\right] + 1 = k\right)$, rezultă că se poate genera o valoare de observaţie x asupra lui X astfel:

```
1: function Geom2(p)
2:     u=urand()
3:     return int(log(1-u)/log(1-p))+1;
4: end function
```

1.7 Simularea unei distribuţiii de probabilitate continuă prin metoda inversării

Metoda inversării de simulare a variabilelor aleatoare continue se aplică pentru variabilele ce au funcţia de repartiţie inversabilă. Reamintim definiţia funcţiei de repartiţie pentru o variabilă aleatoare continuă.

Funcţia de repartiţie a unei variabile aleatoare continue, X, având densitatea de probabilitate f_X este funcţia $F_X : \mathbb{R} \to \mathbb{R}$ definită astfel:

$$F_X(x) = P(X \leq x) = \int_{-\infty}^{x} f(x)\, dx \tag{3}$$

Funcţia de repartiţie, F_X, a unei variabile aleatoare continue este:
a) o funcţie continuă;
b) nedescrecătoare, adică dacă $x_1 < x_2$, atunci $F_X(x_1) \leq F_X(x_2))$ şi
c) $\lim_{x\to-\infty} F_X(x) = 0$, iar $\lim_{x\to\infty} F_X(x) = 1$.

De interes deosebit pentru simulare este funcţia de repartiţie F_U a unei variabile aleatoare, $U \sim$Unif0,1):

$$F_U(x) = \begin{cases} 0 & x < 0 \\ x & x \in [0,1) \\ 1 & x \geq 1 \end{cases}$$

Se mai spune că funcţia de repartiţie, F_U, restricţionată la intervalul $[0,1)$, este funcţia identică, pentru că $F_U(x) = x$.

Şirurile de numere pseudo–aleatoare uniform distribuite pe $[0,1)$ constituie baza oricărei simulări a unui fenomen sau proces aleator. Prin transformări inversabile, $h : \mathbb{R} \to \mathbb{R}$, adecvat alese, un şir (u_n) de numere uniform distribuite pe $[0,1)$ poate fi transformat într-un şir $(x_n = h(x_n))$ ale cărui elemente sunt valori de observaţie asupra unei variabile aleatoare $X = h(U)$, cu $U \sim$Unif[0,1).

Cea mai simplă metodă de transformare a şirului (u_n) este metoda inversării. Ea se aplică pentru a genera numere pseudo–aleatoare ca valori de observaţie asupra unei variabile aleatoare X, ce are funcţia de repartiţie inversabilă. Evident că dacă F_X este strict crescătoare atunci ea este inversabilă. Pentru orice $u \in (0,1)$, $F_X^{-1}(u) \in \mathbb{R}$. Interpretând u ca valoare de observaţie asupra unei variabile aleatoare $U \sim$Unif[0,1), x este valoare de observaţie asupra variabilei $F_X{-}1(U)$. Să determinăm distribuţia de probabilitate a variabilei $Y = F_X^{-1}(U)$:

Propoziţia 1.7.1 *Fie U o variabilă aleatoare uniform distribuită pe $[0,1)$ şi F_X o funcţie de repartiţie strict crescătoare şi continuă pe intervalul de lungime minimă din $\mathbb{R}$, pe care variabila aleatoare X ia valori cu probabilitatea 1. Atunci variabila aleatoare*

$$Y = F_X^{-1}(U)$$

are aceeaşi funcţie de repartiţie ca şi variabila X, adică Y şi X sunt identic distribuite şi nu se disting din punct de vedere probabilist.

Demonstraţie: Deoarece $U \sim \text{Unif}[0,1)$, funcţia sa de repartiţie, F_U, este funcţia identică pe $[0,1)$, adică $F_U(x) = x$, $\forall\, x \in [0,1)$ şi $F_U(x) = 0$, în rest. Să determinăm funcţia de repartiţie G_Y a variabilei $Y = F_X^{-1}(U)$:

$$G_Y(x) = P(Y \leq x) = P(F_X^{-1}(U) \leq x) = P(U \leq F_X(x)) = F_U(F_X(x)) = F_X(x) \qquad (4)$$

Prin urmare $G_Y(x) = F_X(x)$, oricare ar fi x, adică funcţia de repartiţie a variabilei aleatore $Y = F_X^{-1}(U)$ este chiar F_X.

□

Bazat pe acest rezultat, avem următorul algoritm de simulare a unei variabile aleatoare X ce are funcţia de repartiţie F_X inversabilă:

1: **function MetodaInversarii(())**
2: **u=urand();**
3: $x = F_X^{-1}(u)$**;**
4: **return x;**
5: **end function**

Simularea distribuţiei exponenţiale de parametru θ: Funcţia de repartiţie a unei variabile aleatoare, $X \sim \text{Exp}(\theta)$, (Fig.3) este:

$$F_X(x) = \begin{cases} 0 & \text{dacă } x < 0 \\ 1 - e^{-x/\theta} & \text{dacă } x \geq 0 \end{cases} \qquad (5)$$

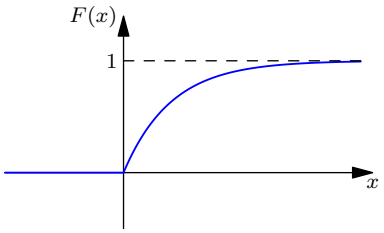


Fig.3: Funcţia de repartiţie a unei variabile aleatoare $X \sim Exp(\theta)$

Observăm că funcţia F_X este strict crescătoare pe intervalul $[0,\infty)$. Deci pentru orice $u \in [0,1)$, $F^{-1}(u) \in [0,\infty)$. Cum o variabilă aleatoare exponenţial distribuită ia valori pozitive cu probabilitatea 1:

$$P(X \geq 0) = 1 - P(X < 0) = 1 - F_X(0) = 1 - 0 = 1,$$

rezultă că putem aplica metoda inversării pentru simularea lui X.

Din $1 - e^{-x/\theta} = u$ rezultă că $x = F^{-1}(u) = -\theta \ln(1-u)$ Astfel putem simula variabila aleatoare, X,în modul următor:

```
1: function SimulExp(theta)
2:     u=urand();
3:     x = −theta ∗ log(1 − u);
4:     return x;
5: end function
```

Să arătăm însă că putem înlocui pe $1-u$ cu u. Mai precis arătăm că:

O dată cu U şi $1-U$ este variabilă aleatoare uniform distribuită pe $[0,1)$. În acest scop calculăm funcţia sa de repartiţie:

$$\begin{aligned} F_{1-U}(x) &= P(1-U \leq x) = P(-U \leq x-1) = P(U > 1-x) \\ &= 1 - P(U \leq 1-x) = 1 - F_U(1-x) = 1-(1-x) = x, \end{aligned}$$

pentru $1-x \in [0,1) \Leftrightarrow x \in [0,1)$. Conform demonstraţiei precedente, înseamnă că funcţia de repartiţie a variabilei $F^{-1}(1-U)$ este F şi prin urmare avem că:

Dacă U este o variabilă aleatoare uniform distribuită pe $[0,1)$, atunci variabila aleatoare $Y = F^{-1}(1-U)$ are drept funcţie de repartiţie pe F.

Prin urmare dacă (u_n), $n = 0, 1, \ldots N$ este un şir de numere pseudo–aleatoare uniform distribuit pe $[0,1)$, atunci sirul $(1-u_n)$, este de asemenea uniform distribuit pe $[0,1)$.

Astfel în simularea unei variabile aleatoare $X \sim Exp(\theta)$ putem înlocui pe $1-u$ cu u: şi avem următorul pseudocod de simulare:

```
1: function SimulExpN(theta)
2:     u=urand();
3:     x = −theta ∗ log(u);
4:     return x;
5: end function
```