***Definitia numerelor aleatoare***

Este un fact sustinut de toti cei care au studiat numerele aleatoare, faptul ca oamenii sunt generatoare de numere aleatoare foarte slabe.

O definitie a numerelor aleatoare se bazeaza pe faptul ca nu se poate prezice urmatoarea secventa pe baza a ceea ce a fost generat.

O alta definitie spune faptul ca un numar aleator este un numar care a fost extras dintr-un set de valori posibile, dintre care fiecare este egal probabil(uniform distribuit), statistic independent de celelalte.

Un generator de numere aleatoare este un dispozitiv computational sau fizic conceput pentru a genera o secventa de numere sau de simboluri care nu pot fi prezise rezonabil decat prin sansa aleatoare.[1]

Analizand toate aceste definitii intr-un mod pur matematic, am avea urmatoarele:

Se presupune ca avem o multime finite de elemente X si o multime finite de descrieri ale obiectelor X in multimea Y. Fie functia D de la Y la X astfel incat pt fiecare obiect x apartinand lui X sa existe o descriere y, apartinand Y astfel incat D(y)=x. Fiecare descriere sa fie un sir de caractere finit. Complexitatea descrierii este data de lungimea sirelui de caractere care descrie obiectul. Cele mai multe siruri de caractere de descriere pentru n nu au lungimea mai scurta de n.

De exemplu:

1000100010001000100010001000100010001000100010001000100010001000

Poate fi descries ca`16 copii de 1000', in timp ce sirul:

0101011110100011001011010000110001100001100111110100010001010110

Nu are o descriere mai scurta decat sirul in sine.[1]

Knuth a dat urmatoarea definitie a unei secvente infinita de numere aleatoare:

O secventa de numere reale U0,U1,U2,…in intervalul [0,1) este definite ca fiind aleatoare daca oricare ar fi P, o proprietate astfel incat P([Vn]) are proprietatea de 1 pentru o secventa V0,V1,V2, … de mostre independente de variabile aleatoare din distributia uniforma, atunci P([Un]) este adevarat.[2]

Totusi aceasta definitie nu va fi niciodata intalnita in viata reala.

***Generarea de numere aleatoare***

Nu putem trece mai departe pana nu analizam ce inseamna de fapt generarea de numere aleatoare. Daca am inteles in esenta ce presupune un numar aleator putem trage urmatoare concluzie:

Exista doua metode principale de a genera numere aleatoare. Pe de-o parte avem numere aleatoare “adevarate”. Acestea sunt generate avand la baza un fenomen fizic. De exemplu daca intram pe unul dintre cele mai folosit generator de numere aleatoare in momentul de fata, si anume Random.org, vom vedea faptul ca numerele aleatoare sunt generate pe baza zgomotului atmosferic.

Pe de alta parte avem pseudo-numere aleatoare. Aceste numere sunt generate pe baza unor algoritmi computationali care pot produce secvente lungi de numere aparent aleatoare, care de fapt pot fi complet determinate de la o secventa initiala mai scurta.

Mai mult decat atat unele sisteme au o abordare hibrida, culegand aleatorismul din fenomene natural, iar atunci cand rata de citire a datelor aleatoare ajunge la limita methodei natural, sistemul trece pe abordarea cu generatoare de numere aleatoare cu reinsamantare criptologica securizata bazata pe software.[3]

Cele mai multe calculatoare au la baza pseudo-numere aleatoare, ceea ce inseamna ca desi pot genera secvente foarte lungi de numere aleatoare, la un moment dat acea secventa se va repeta.

Modul in care generam numerele aleatoare se mai pot imaprtii si in deterministice si non deterministice, si anume, pseudo-numerele aleatoare sunt numere deterministice iar numerele aleatoare “adevarate” sunt non-deterministice, ceea ce inseamna cu alte cuvinte ca, pe de-o parte nu avem o metoda determinista pentru a genera urmatorul numar sau urmatoarea secventa, si in acelasi timp nu putem determina pe baza unei secvente, secventa anterioara sau secventa urmatoare.

Exista numeroase metode de a genera numere aleatoare non-deterministe, pornind de la cele mai simple, aruncatul cu zarul, datul cu banul, a alege numere dintr-o palarie, extragerea bilelor dintr-o urna, s.a.m.d pana la cele mai complexe cum a fi zogomotul atmosferic , lampe cu lava, efecte radioactive sau chimice s.a.m.d.

Pe de alta parte si metodele nondeterministice ar putea fi imbunatatite astfel incat generarea de pseudo-numere aleatoare sa fie cat mai buna si numere sa se repeta cat mai tarziu.

***Aplicatii ale numerelor aleatoare***

Pentru an intelege in profunzime numerele aleatoare trebuie sa vedem si utilitatea acestora, pentru ca pentru aplicatii diferite avem nevoie si de tipuri diferite de generatoare de numere aleatoare.[4]

1. Jocuri, pariuri – Probabil cea mai raspandita aplicatie a numerelor aleatoare este cea recreationala. Jocurile au fost cele care au pus prima data probleme aleatorismului, asadar nu e de mirare ca cea mai intalnita definitie a aleatorismului este acea a “Aruncarii zarurilor”

2. Simulare – Numerele aleatoare joaca un rol foarte important in crearea de modele cat mai aproape de realitate. Aceste simulari pot fi de la simulari la nivel economic, traffic pana la fizica nucleara si multe altele. Pentru aceste aplicatii sunt folosite in generar pseudo-generatoare de numere aleatoare, deoarece ele pot sa inceapa de fiecare data la fel, permitand diferitilor parametrii sa varieze. [1]

3. Mostre statistice – pentru a testa o colectie mare de lucruri cea mai buna metoda este sa luam o mostra aleatoare astfel incat rezultatul sa fie cat mai precis.

4. Criptografie –Sistemul criptografic de chei publice foloseste un numar foarte mare de date aleatoare.De exemplu RSA are nevoie de de foarte multe numere prime aleatoare pentru securitate. De asemenea si One-Time-Password foloseste un keystream foarte mare de numere intregi aleatoare.

5. Computer Programming – Foarte multi algoritmi necesita numere aleatoare sau secvente aleatoare, de asemenea numerele aleatoare pot fi folsite si ca date de intrare pentru testare eficientei unui algoritm.

6. Analiza numerica- Foarte multe probleme care sunt foarte greu de rezolvat se pot aproxima prin tehnici bazate pe numere aleatoare

7. Arta si muzica – Cel mai concret exemplu al aleatorismului este curentul Dadaist care producea arta facand totul la intamplare. De exmplu poeziile dadaiste erau concepute extragand la intamplare diferite cuvinte dintr-un sac sau o palarie.

***Testarea generatoarelor de numere aleatoare***

Pentru a vedea daca un numar aleator este cu adevarat aleator sau mai putin aleator, avem nevoie de o suita de teste care sa ne ajute sa decidem acest lucru. Pentru asta avem doua tipuri de teste: teste empirice si teste teoretice.

Testele empirice sunt aplicate pe o secventa generata de un generator de numere aleatoare si nu necesita nicio cunostinta despre modul in care au fost generate numerele aleatoare. Dupa cum spune si numele lor aceste teste nu au niciun fel de substrat teoretic ci sunt bazate numai pe date furnizate in urma unor experimente.

Testele teoretice , care sunt mai bune atunci cand ele exista, necesita o oarecare cunostinta despre structura generatorului de numere aleatoare, si nu necesita neaparat ca secventa de numere aleatoare sa fie generate.

***Chi-Square Test***

Testul Chi-Square(x2) a fost initial publicat in anul 1900 de catre Karl Pearson.[1]

Pentru a explica cat mai bine modul de testare am sa folosesc acelasi exemplu dat de Knuth in cartea sa The Art of Computer Programming [2] si anume, sa presupunem ca avem sarcina de arunca doua zaruri, atunci avem:

Valorile lui s = 2 3 4 5 6 7 8 9 10 11 12

Probabilitatea, ps=

Pentru a explica valorile de mai sus avem de exemplu valoarea 4 care poate fi obtinuta aruncand cu zarul si obtinand 1 si 3, 2 si2, 3 si 1. Numarulor total de valori care pot fi optinute este egal cu 36(6x6) asadar probabilitatea ca sa obtinem o valoare de 4 este de 3/36=1/12.

Daca aruncam cu zarul de n ori, ar trebui sa obtinem valoarea s de nps ori in medie. Deexemplu, daca n este 144, ar trebui sa obtinem valoarea lui 4 de exact 12 ori in cele 144 de aruncari, conform cu formula, dar in viata reala nu este asa. Avem asadar o secventa de 144 de numere aleatoare generate care au fost impartite in categoriile s astfel incat Ys reprezinta numarul de numere aleatoare din fiecare categorie s

Valorile lui s = 2 3 4 5 6 7 8 9 10 11 12

Valorile observate, Ys= 2 4 10 12 22 29 21 15 14 9 6

Valorile asteptate,nps= 4 8 12 16 20 24 20 16 12 8 4

Pentru a intelege tabelul de mai sus trebuie sa intelegem ca am plecat de la premisa ca Ys≈nps, deoarece Ys reprezinta numarul de observatii care se gasesc in categoria s , iar ps reprezinta probabilitatea ca o observatie sa fie in categoria s.

Mai trebuie subliniat si faptul ca :

Y1+Y2+...+Yk=n, unde k reprezinta numarul de categorii

P1+p2+...+pk=1

Revenind la tabelul de mai sus, ceea ce trebuie dedus este faptul ca Ys nu va fi niciodata identic cu nps . Asadar cum ne putem da seama daca numere generate sunt numere aleatoare “bune” sau “rele”. Raspunsul dat de chi-square test este unul de natura probabilistica, si anume ca putem sa spunem cat de probabil este unu anume eveniment sa se produca si cat nu.

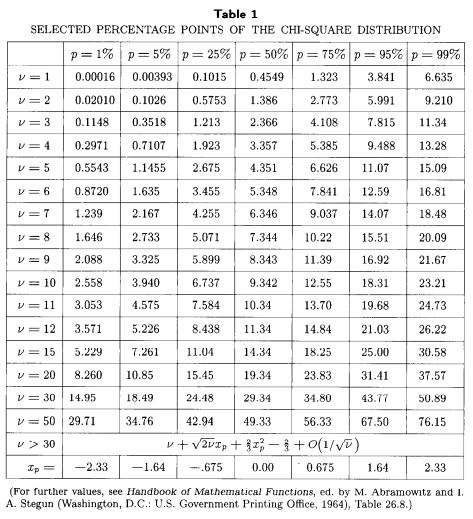
Un mod corect si natural pentru a continua cu exemplul de mai sus asa cum este si explicat in cartea lui Knuth, este in a considera suma patratelor diferentei dintre Ys si nps asadar avem:

V=(Y2-np2)2 +(Y3-np3)2+....+(Y12-np12)2

Un set “rau” de date ar duce asadar la o valoare relativ mare a lui V, si pentru orice valoarea lui V ne putem intreba “Care este probabilitatea ca V sa fie atat de mare, folosind zaruri adevarate?” Daca aceasta probabilitate este foarte mica, sa zicem 1/100, atunci vom sti ca zarurile adevarte doar 1/100 ar da rezulate atat de departe de numerele asteptate, si am defini limite de suspiciune. Totusi testul ar trebui repetat pentru ca si pana un zar adevarat ar da acele erori 1 la 100.

In general V se poate calcula cu formula:

Dupa cum spuneam acesta este un test probabilistic asadar, avem un tabel de referinta obtinut in urma calcularii valorilor lui V



Avand o valoare V, cum ne dam seama daca numere aleatoare generate de noi sunt bune sau sunt rele? Tabelul 1 ne ajuta sa facem asta. Fiecare linie are definiata o valoare v care reprezinta k-1, cu unu mai putin decat numarul de categorii. Comparati V cu intrarea din tabel, de exemplu, daca aveti 9 categorii, k=9(v=k-1=8), atunci avem la 99% 20.09 ceea ce inseamna ca V<20.09 in aproximativ 99% din cazuri si ne asteptam ca V>20.09 doar in 1% din cazuri. De mentionat faptul ca testul chi-square ofera doar probabilitati aproximative cu toate acestea o valoare a lui V de 35 ar fi destul de suspicioasa, si testul ar trebui repetat, si in cazul de 1% si in cazurile care ridica suspiciuni.

Pentru a sumariza testul chi-square: este nevoie sa avemin fiecare categorie cel putin 5 observatii. Se calculeaza V, in functie de gradul de libertate(v=k-1) ne uitam in tabelul 1 si daca V este mai mic de 1% sau mai mare de 99% respingem secventa ca nefiind destul de random. Daca V este intre 1% si 5% sau intre 99% si 95% atunci numerele sunt suspecte. Daca V este intre 5% si 10% sau intre 90% si 95% aunci numere sunt aproape suspecte . In rest numerele sunt suficient de aleatoare.

Testul chi-squere este de obicei efectuau de cel putin 3 ori pe seturi de date diferite, si daca celputin 2 din cele 3 sunt suspecte atunci ele sunt declarate ca fiind insuficient de aleatoare.

***Testul Kolmogorov-Smirnov***

Testul Kolmogorov-Smirnov este foarte util in cazurile in care numere aleatoare nu pot fi impartite in categori finite, sau nu avem o secventa finita de numere pe care sa o impartimin categori.

Pentru a intelege testul lui Kolmogorov-Smirnov trebuie sa definim functia distribuita:

Avand variabila aleatoarea X avem functia cumulativa distribuita (fcd)Fc(x) definita dupa cum urmeaza:

FX(x)=probabilitatea ca (X≤x), FX(x) ϵ[0,1], x ϵ(-∞,+∞)

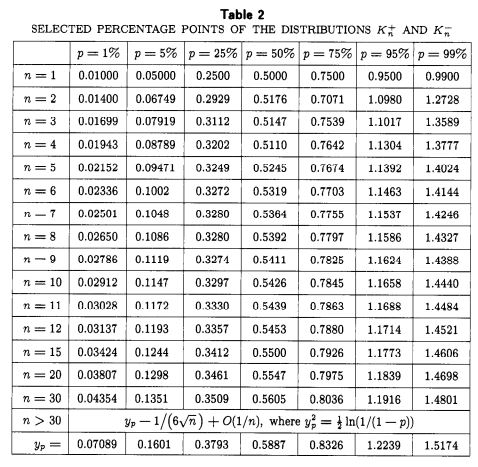
X reprezinta valorile generate de RNG. Daca generam n observatii independente pentru cantitatile random X, avand astfel X1, X2, ... Xn, putem forma astfel distributia empirica Fn(x) unde:

Testul KS compara FX(x) cu Fn(x) masurand diferenta dintre cele doua functii dstribuite. Cand n este suficient de mare , ne asteptam ca cele doua functii sa fie similare daca secventa pe care o examinam este intradevar aleatoare. Avem asadar urmatoarea statistica:

Kn+=√nmax((Fn(x)-FX(x)), -∞ < x < +∞

Kn-=√nmax((FX(x)-Fn(x)), -∞ < x < + ∞

Aceste statistici se folosesc la fel ca testul chi-squer, comparand valorile optinute cu un tabel de referinta.



Acest tabel se foloseste exact ca tabelul1,diferenta este ca acesta nu ofera doar o aproximare si valori exacte. De exemplu probabilitatea este ca 75% ca K20- sa fie 0.7975 sau mai putin.

Testul KS poate fi folosit impreuna cu testul X2 pentru a duce la o mai buna procedura . O diferenta importanta intre teste KS si teste chi-squere este aceea ca testele KS snt aplicabile doar pe distributii fara continue, fara niciu fel de discontinuitate, in timp ce X2 este aplicat numai pe date discontinue.

***Teste Empirice***

Fiecare test este aplicat pe o secventa (Un)=U0,U1,... care reprezinta numere reale independente si uniform distribuite intre 0 si 1.

Pentru teste construite pe numere intrege secventa destinta teste (Yn)=Y0,Y1,... care este definita de regula:

Yn=[dUn], acesta secventa are valori intregi si uniform distribuite intre0 si d-1, este ales dupa cum convine.

***Equidistribution sau Frequency test***

Pentru a putea aplica acest test, secventa trebuie sa fie uniform distribuita. Exista doua metode de a aplica acest test, una folosind KS si una folosind testul X2 . In esenta testul spune ca daca avem o secventa de numere aleatoare de 1 si 0 , ne asteptam sa avem acelasi numar de 0 ca si de 1.

***Serial Test***

Testul equidistribuit verifica daca un anume numar dintr-o secventa apare de mai multe ori decat ar trebui , dar ar trebui sa stim si daca o anumita secventa sau pereche de numere este uniform distribuita.

Pentru a realiza testul se numara de cate ori se intalneste perechea:

(Y2j+,Y2j+1)=(q,r) pentru 0≤j<n

Calculul ar trebui facut pentru orice pereche (q,r). Avem astfel d2 categories pentru X2 teste cu o probabilitate de 1/ d2 asignate la fiecare categorie.

Aceasta metoda poate fi generalizata si pentru triplete, quadruple,etc.

***Gap Test***

Gap test masoara care este distantele dintre doua aparitii ale lui Uj intr-un anumit range.

Algoritmultmul pentru Gap test:

G1.[Initializare] Set j <- -1, s <- 0, and set COUNT[r] <-0 for 0<=r<=t

G2[Set r zero] Set r <- 0.

G3. [α≤Uj< β?] Increase j by 1. If Uj ≥ α and Uj< β, go to step G5.

G4.[Increase r.] Increase r by 1, and retur to step G3.

G5.[Recorde gap length.] (A gap of length r has now been found.) If r≥t, increase COUNT[t] by 1, otherwise increase COUNT[r] by 1.

G6.[n gaps found?] Increase s by 1. If s<n, return to step G2.

Dupa ce se efectueaza acest algoritm, se aplica chi-square test pe k=t+1 valori dn COUNT[0],COUNT[1],...COUNT[t] folosind probabilitatile:

p0=p, p1=p(1-p), p2=p(1-p)2, ...,

pt-1=p(1-p)t-1, pt=(1-p)t.

Aici p= β- α, reprezinta probabilitatea ca α≤Uj< β. Testul este de obicei aplicat cu α=0 si β=1.

***Poker Test***

Testul Poker are la baza, dupa cum ii spune si numele, regulile jocului de Poker. Se considera n grupe de cate 5 intregi succesivi. Regulile care se observa pe acesti intregi sunt una din urmatoarele:

Toate diferite: abcde Full house: *aaabb*

O pereche: *aabcd* Patru de acelasi fel: *aaaab*

Doua perechi: *aabbc* Cinci de acelasi fel: *aaaaa*

Trei de acelasi fel: *aaabc*

Avand aceste categorii, putem folosi testul chi-square folosind numarul de tuple din fiecare categorie.

***Coupon Test***

Testul Coupon poate fi exemplificat subforma colectionarii de cupoane din cutiile de cereale cu scopul de a avea cate un cupon din fiecare fel.

Algoritmul care realizeaza testul cuponului este prezentat in detaliu in cartea Empirical Tests[2]. Acest algoritm are ca si date de iesire doua numere COUNT[r], care reprezinta numarul de segmente de lungime r, COUNT[t], care reprezinta numarul de segmente de lungime >=t.

Testul chi-square este aplicat pe COUNT[d], COUNT[d+1]... COUNT[t] avand k=t-d+1m dupa ce algoritmul a calculat n lungimi.( 0 -> d-1 reprezinta intervalul de valori )

***Permutation Test***

Testul imparte secventa de intrare in n grupe de t elemente fiecare (Ujt,Ujt+1,...,Ujt+t-1) pentru 0<=j<n. Elementele din fiecare grup pot avea t! ordonari relative posible; numarul de ori fiecare ordonare apare este numarata, si testul chi-square este aplicat cu k=t! si probabilitatea 1/t! pentru fiecare ordonare.

De exemplu, daca t=3 vom avea 6 posibile categori, in conformitate cu U3j <U3j+1<U3j+2 sau U3j <U3j+2<U3j+1 sau .... sau U3j+2 <U3j+1<U3j . Am presupus in acest test faptul ca nu se va intampla sa obtinem doua U cu aceeasi valoare, probabilitatea ca acest lucru sa se intample este 0.[2]

***Run Test***

O secventa poate fi de asemenea testata pentru “urcari” si “coborari”. Acest lucru inseamnca ca examinam lungimea de subsecvente monotone din secventa originala.

Ca si exemplu , putem considera secventa de zece numere “1298536704:, punand cate o linie verticala la stanga, la dreapta si intreXj si Xj+1 oricand Xj>Xj+1 obtinem:

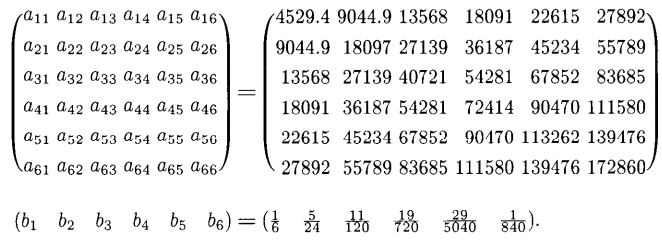
| 1 2 9 | 8 | 5 | 3 6 7 | 0 4 |

Asadar am araratam care sunt “urcarile”. Avem un run de lungime 3, urmat de doua run-uri de lungime 1, urmat de un alt run de lungime 3 si un ultim run de lungime 2.

Spre deosebire de testul gap si coupon collector, nu ar trebui sa aplicam testul chi-square pe datele de mai sus , observand ca run-urile adicante nu sunt independente. Un run mai lung are tendinta sa fie urmat de un run mai scurt si viceversa. Aceasta dependenta este destula ca sa invalideze testul chi-square. In schimb se poate compune urmatoare statistica:

V=

Unde n este lungimea secventei si coeficinetii aij si bj sunt egali cu:



Implementare testare:

Teste scrise in python.

Dependenta de scipy.

Pip install numpy

Pip install egg

Install Cygwin

Pip install Cython h5py

Pip install tempita

pip install scipy

Referinte:

[1]- Testing Random Number Generators Dan Biebighauser University of Minnesota - Twin Cities REU Summer 2000

http://www-users.math.umn.edu/~garrett/students/reu/pRNGs.pdf

[2]- Knuth D. E. Arta programării calculatoarelor vol.2, Algoritmi seminumerici, Editura Teora, București, 2000.

http://library.aceondo.net/ebooks/Computer\_Science/algorithm-the\_art\_of\_computer\_programming-knuth.pdf

[3]-Random number generation-Wikipedia

[4]- Applications of randomness-Wikipedia