UNIVERZITET U NOVOM SADU
PRIRODNO-MATEMATIČKI FAKULTET
DEPARTMAN ZA
MATEMATIKU I INFORMATIKU

Smilja Rakić

Neke Monte Karlo metode i njihove primene

-MASTER RAD-

Mentor: dr Danijela Rajter-Ćirić

Novi Sad, 2021.

Predgovor

U master radu, koji je pred Vama nastojaću da predstavim neke Monte Karlo metode i njihove primene. Markovljev lanac Monte Karlo (MCMC) je sve popularnija metoda za dobijanje informacija o raspodelama, ova metoda obuhvata klasu algoritama za uzorkovanje iz raspodele verovatnoća, kada je direktno uzorkovanje teško. Dobijeni uzorak se može koristiti za aproksimiranje raspodele ili za izračunavanje integrala i obično se koriste za uzorkovanje iz višedimenzionalnih raspodela, posebno kada je broj dimenzija visok. Prvo poglavlje ovog rada je posvećeno uvodu i istorijatu, dok naredno sadrži neke osnovne definicije i teoreme Markovljevih lanaca, koje koristimo kasnije u petoj glavi kada budemo demonstrirali Metropolis – Hejstings algoritam koji koristi Monte Karlo Markovljeve lance. Treće poglavlje je prikaz Monte Karlo metode u integraciji, koja predstavlja jedan probabilistički pristup problematici numeričke integracije. Potom, u četvrtoj glavi su vršene raznorazne Monte Karlo simulacije koje služe za generisanje slučajnih promenljivih. Poslednje poglavlje predstavlja kratak uvod u jednu vrlo zanimljivu primenu Monte Karlo metoda na teoriju igara, predstavljenu algoritmom Monte Karlo pretrage stabla (MCTS).

Svojoj mentorki, prof. dr Danijeli Rajter-Ćirić se zahvaljujem na predloženoj temi za ovaj master rad. Obzirom da se u dosadašnjem školovanju nisam susrela sa stohastičkom analizom, samim tim ni sa Monte Karlo metodama, ova tema je za mene bila veoma inspirativna. Posebno joj se zahvaljujem na stručnoj pomoći, savetima i razumevanju, koje mi je pružila tokom izrade ovog rada, kao i tokom osnovnih i master studija.

Zahvaljujem se svim svojim prijateljima i kolegama, koji su mi na bilo koji način pružili pomoć i podršku tokom studiranja. Posebno se zahvaljujem svojoj porodici na neizmernoj podršci, razumevanju i ljubavi koju mi pružaju.

Sadržaj

1. Uvod i istorijat

1.1 Definicija

Monte Karlo metode su zajedničko ime za široku klasu metoda koje su vezane za rešavanje deterministickih problema na probabilistički način, tj. rešavanje problema koji imaju konkretno numeričko rešenje koje je često nemoguće odrediti vršenjem ogromnog broja eksperimenata zarad dobijanja aproksimacije primenom centralne granicne teoreme.

Sama metoda ima vrlo širok spektar primena, od tipičnih primena u matematici (u vidu numeričke integracije, optimizacije i generisanju uzoraka) do primena u računarskoj fizici i hemiji, računarskoj grafici i teoriji igara.

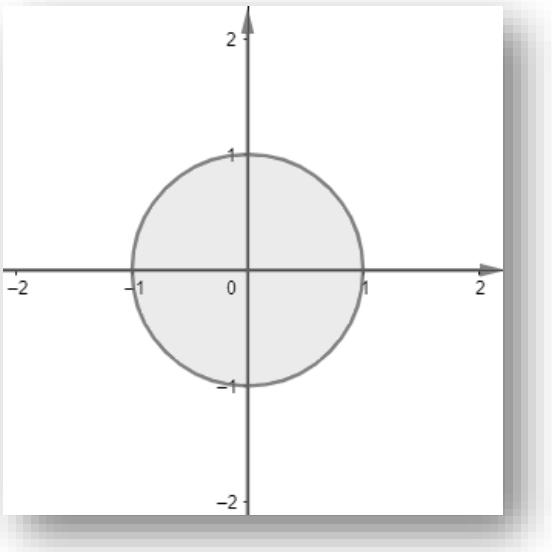
Budući da se odnosi na široku klasu metoda za rešavanje raznovrsnih problema, ne postoji strogo formalna definicija ovih metoda, ali se mogu izdvojiti zajednički koraci u algoritmima koji tu klasu metoda predstavljaju:

1. Definisati problem i domen ulaznih podataka problema;
2. Generisati niz slučajnih promenljivih iz tog domena;
3. Izvršiti determinističke proračune s tim slučajnim nizom;
4. Dati procenu rešenja problema na osnovu dobijenog rezultata

1.2 Uvodni primer – aproksimacija broja pi

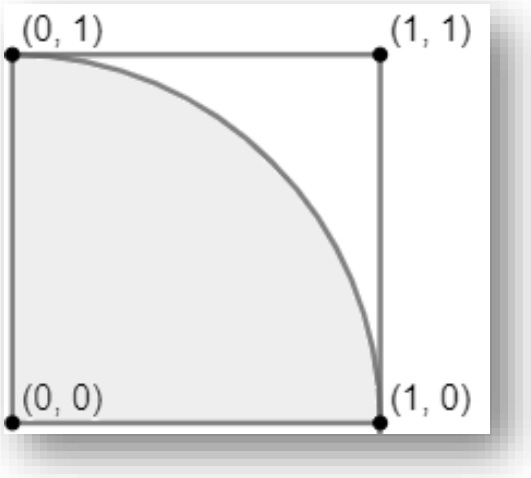
Ilustrativni primer toga vidimo na ***Slika 1.1***, a i primer koji će biti od značaja u jednom od daljih poglavlja.

Broj $\boldsymbol{\pi}$ možemo videti kao površinu jediničnog kruga:



Slika 1.1

Dakle, sam broj $\boldsymbol{\pi}$ možemo aproksimirati ako damo neku aproksimaciju za površinu četvrtine kruga (*Slika 1.2*) i pomnožimo na kraju sa 4.



Slika 1.2

Ideja metode koju ćemo za to primeniti je sledeća:

Nasumično ćemo generisati tačke iz kvadrata $[0,1] \times [0,1]$ i na kraju uporediti broj onih unutar četvrtine kruga i onih izvan i pomnožiti sa 4. Za ovaj eksperiment koristićemo sledeći C-kod:

```c
#include <stdio.h>
#include <stdlib.h>



int main(int argc, char *argv[]) {

int i,N;
int s=0;
scanf("%d",&N);
for(i=0;i<N;i++){
double d1=((double)rand() * 1.0 ) / (double)RAND_MAX ;
double d2=((double)rand() * 1.0 ) / (double)RAND_MAX;
if((d1*d1+d2*d2)<=1)
s++;


}
printf("%f",4.0*s/N);

return 0;
}
```

Kod 1.1

Dobijeni rezultati u eksperimentu:

n	10	100	1000	100000	1000000	10000000	100000000
rezultati	3.600000	3.200000	3.112000	3.137400	3.139088	3.140932	3.141353

Tabela 1.1

1.3 Istorijat

Jedan od najstarijih primera Monte Karlo metoda ja vezan za Bufonovu iglu. Ona je jedan vid Monte Karlo simulacije koji potiče od problema koji je postavio Žorž Luis Lekler (1707-1788.).

Ako bacamo drvce dužine l na ravan u vidu rešetke, gde je razmak dve uzastopne rešetke d, koja je verovatnoća da drvce padne na rešetku. On dolazi do rešenja $\frac{2l}{\pi d}$.

Laplas (1749 -1827.) na osnovu toga daje sledeću ideju za aproksimaciju broja π:

Neka se eksperiment bacanja drvceta izvodi *n* puta i neka je *N* broj slučajeva kada je drvce palo na rešetku. Tada je $\sim \frac{2l}{d \cdot \frac{N}{n}}$. Očito, kako $n \to \infty$, to će se dobijati sve preciznija aproksimacija za broj $\boldsymbol{\pi}$.

Monte Karlo simulacijama se bavio italijanski fizičar i nobelovac Enriko Fermi (1901.-1954.) 30-ih godina prošlog veka radeći na difuziji neutrona. Njegova primena Monte Karlo metoda je bila poprilično elementarna, koristeći digitron, ručne proračune i što veći broj eksperimenata. S obzirom na nedostatak kvalitetnije tehnologije za obimnije proračune, nije mogao dobiti finije procene i zato nije objavio taj svoj rad.

Formalno, začetak Monte Karlo metoda je vezan za Stanislava Ulama (1909.-1984.), koji je radio s Monte Karlo Markovljevim lancima u Los Alamos laboratoriji u S.A.D. prilikom Menhetn projekta (projekta vezanog za konstrukciju atomske bombe).

Kuriozitet je da je Ulam počeo da koristi Monte Karlo metode i pre toga; po anegdoti dok je bio bolestan voleo je da igra Kenfild pasijans (izuzetno teška varijanta popularne kartaške igre pasijans). Indisponiran porazima, počeo je da računa kolika je verovatnoća pobede u nasumičnoj partiji Kenfild pasijansa. No, kombinatorni račun je ubrzo postao prekomplikovan, te je došao na ideju da odigra ogroman broj partija i da proceni prema tome traženu verovatnoću.

Oduševljen Ulamovom idejom, njegov kolega iz Los Alamos laboratorije i saradnik na Menhetn projektu DZon Fon Nojman (1903-1957.) je koristeći svoj ENIAC kompjuter isprogramirao neke od prvih programa koji su vršili proračune koristeći Monte Karlo Metode. Budući da su imali veoma kvalitetnu tehnologiju na raspolaganju, napravili su značajne rezultate na polju istraživanja difuzije neutrona.

Kako su radili u vojnoj bazi, njihov projekat je zahtevao šifrovano ime. Predloženo je ime Monte Karlo, jer je ujak Stanislava Ulama veoma voleo da provodi vreme po brojnim kazinima Monte Karla i samo ime je sjajno oslikavalo probabilistički duh eksperimenta i samih metoda korišćenih u njemu.

2. Markovljevi lanci

2.1 Slučajni procesi

Definicija 2.1. *Slučajni (stohastički) proces je familija slučajnih promenljivih* $\{X_t\}_{t\in T}$ *definisana na istom prostoru verovatnoća* (Ω, A, P), *gde je* Ω *– prostor verovatnoća,* A *– sigma algebra na* Ω *i* $P\colon A \to [0,1]$ *funkcija verovatnoće.*

Najčešće, u primenama, indeks T ima ulogu vremena, pa je $T = [0, +\infty)$ ili neki podskup od $[0, +\infty)$.

Ukoliko je T diskretan skup, $\{X_t\}_{t\in T}$ je diskretan slučajni process. U suprotnom, radi se o neprekidnom slučajnom procesu.

Primetimo da stohastički proces zavisi od $t \in T$ i $\omega \in \Omega$. Jedan od ishoda slučajnog procesa je trajektorija (realizacija) slučajnog procesa.

Fiksiranjem n-torke $(t_1, \dots, t_n)$ dobijamo slučajni vector $(X_{t_1}, \dots, X_{t_n})$.

Definicija 2.2. *Konačnodimenzione raspodele slučajnog procesa* $\{X_t\}_{t\in T}$ *su funkcije raspodele slučajnih vektora* $(X_{t_1}, \dots, X_{t_n})$ *za* $(t_1, \dots, t_n) \in T^n$. *Dakle, tipična konačnodimenziona funkcija raspodele slučajnog procesa* $\{X_t\}_{t\in T}$ *je oblika:*

$$F(t_1, \dots, t_n\,;\, x_1, \dots, x_n) \overset{\text{def}}{=} P\{X_{t_1} \le x_1, \dots, X_{t_n} \le x_n\}$$

Konačnodimenzione raspodele imaju fundamentalnu vezu u izučavanju slučajih procesa, pa kao što svaki slučajni proces ima pridruženi skup konačnodimenzionih raspodela, to i svakom skupu konačnodimenzionih raspodela odgovara jedan slučajni proces. Preciznije, važi sledeće tvrđenje:

Teorema 2.1. *Neka je data familija funkcija* $\{F_n(t_1, \dots, t_n\,;\, x_1, \dots, x_n)\}_{n\in\mathbb{N}}$ $(t_1, \dots, t_n) \in T^n$, $(x_1, \dots, x_n) \in \mathbb{R}^n$ *za koju važi:*

a) $\forall (t_1, \dots, t_n) \in T^n$ *funkcija* $(x_1, \dots, x_n) \to F_n(t_1, \dots, t_n\,;\, x_1, \dots, x_n)$ *je funkcija raspodele nekog slučajnog vektora.*

b) Za svaku permutaciju $\big(j(1), \dots, j(n)\big)$ *skupa* $\{1, \dots, n\}$ *važi:*

$$F_n\big(t_{j(1)}, \dots, t_{j(n)}\,;\, x_1, \dots, x_n\big) = F_n(t_1, \dots, t_n\,;\, x_1, \dots, x_n)$$

c) $(\forall\, n \in \mathbb{N})\ \lim_{x\to+\infty} F_n(t_1, \dots, t_n\,;\, x_1, \dots, x_n) = F_{n-1}(t_1, \dots, t_{n-1}\,;\, x_1, \dots, x_{n-1})$

Tada postoji slučajni proces $\{X_t\}_{t\in T}$ *čije su konačnodimenzione funkcije raspodele* F_n.

Definicija 2.3. *Slučajni proces* $\{X_t\}_{t\in T}$ *je stacionaran ukoliko važi:*
$(\forall\, s \in T)\ (X_{t_1}, \dots, X_{t_n}) = (X_{t_1+s}, \dots, X_{t_n+s})$*, tj. konačnodimenzione raspodele su mu invarijantne na translaciju u vremenu.*

Definicija 2.4. *Neka je X diskretna slučajna promenljiva sa vrednostima iz skupa* $\{x_1, x_2, \dots\}$

$$X\colon \begin{pmatrix} x_1 & x_2 & x_3 \\ p_1 & p_2 & p_3 \dots \end{pmatrix}$$

Ukoliko red $\sum_k x_k\, P\{X = x_k\}$ *apsolutno konvergira, tada definišemo:*

$$E(X) \stackrel{\text{def}}{=} \sum_k x_k\, P\{X = x_k\}$$

Neka je X slučajna promenljiva sa gustinom verovatnoće $\rho(x)$*. Ukoliko* $\int_{-\infty}^{+\infty} |x\rho(x)|dx < +\infty$*, definišemo:*

$$E(X) \stackrel{\text{def}}{=} \int_{-\infty}^{+\infty} xf(x)dx$$

$E(X)$ *je matematičko očekivanje slučajne promenljive* X.

Definicija 2.5. *Slučajni proces* $\{X_t\}_{t\in T}$ *je stacionaran u širem smislu ako je* $E(X_t) = const$ *i* $E(X_sX_t)$ *zavisi samo od* $t - s$.

Definicija 2.6. *Proces sa nezavisnim priraštajima je proces* X_t *za koji važi da su slučajne promenljive (tzv. priraštaji)*

$$X_{t_0}, X_{t_1} - X_{t_0}, X_{t_2} - X_{t_1}, \dots, X_{t_n} - X_{t_{n-1}}, \dots$$

nezavisne za bilo koji izbor $t_0 \le t_1 \le \cdots \le t_n \le \cdots$

2.2 Primeri slučajnih procesa

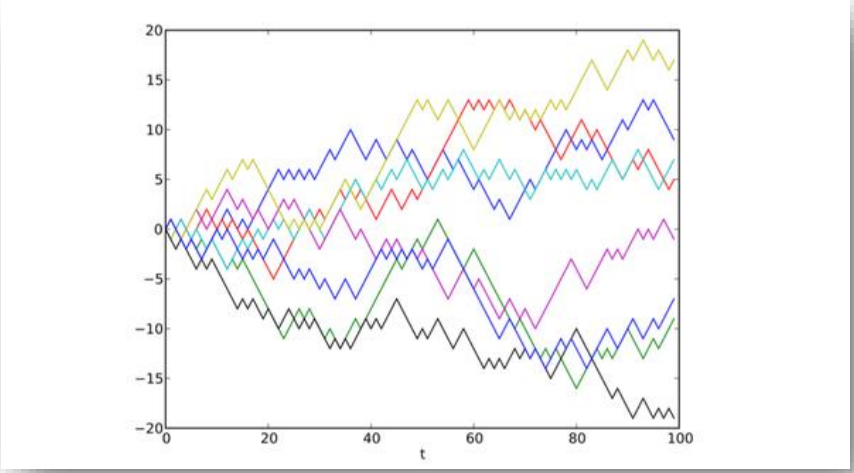
Primer 2.1. *(Bernulijev slučajni proces)* Najprostiji primer slučajnih procesa je Bernulijev, koji predstavlja prirodno uopštenje Bernulijeve-indikator slučajne promenljive:

$I_A: \begin{pmatrix} 0 & 1 \\ 1-p & p \end{pmatrix}$ indikator događaja A.

Bernulijev slučajni proces je niz $\{X_n\}_{n=1}^{+\infty}$ nezavisnih i isto raspodeljenih Bernulijevih slučajnih promenljivih.

Prirodno uopštenje Bernulijevog slučajnog procesa je slučajna šetnja, koja predstavlja niz nezavisnih i isto raspodeljenih slučajnih promenljivih $\{X_t\}_{t\in[0,+\infty)}$ $X_t: \begin{pmatrix} -1 & 1 \\ 1-p & p \end{pmatrix}$

Dakle, ako posmatramo da se čestica može pomeriti u jedinici vremena "naviše" s verovatnoćom p i "naniže" sa verovatnoćom $1-p$ dobijamo fizičku interpretaciju slučajne šetnje. Naravno, ako je $t = \mathbb{N}$, radi se o diskretnoj šetnji, mada neprekidan slučaj ima mnogo očigledniju fizičku interpretaciju. Ovde se može videti par konkretnih realizacija slučajnog procesa koji predstavlja slučajnu šetnju:



Slika 2.1 - Slika preuzeta iz [20]

Sa *Slika 2.1* se jasno vidi zašto se ove slučajne promenljive zovu "šetnja pijanca".

Primer 2.2. *(Gausov slučajni proces)* Slučajni proces čije su sve konačnodimenzione raspodele Gausove (normalne) se zove Gausov slučajni proces. Koristeći činjenicu da su višedimenzione normalne raspodele potpuno određene očekivanjem i kovarijansnom matricom, to je Gausov slučajni proces $\{x_t\}_{t\in T}$ potpuno određen preko funkcija:

$b(t) \stackrel{\text{def}}{=} E(X_t), \forall t \in T$ – očekivanje

$C(s,t) \stackrel{\text{def}}{=} E((X_s - b(s))(X_t - b(t))), \forall s,t \in T$ – kovarijaciona funkcija

Važi sledeće tvrđenje:

Teorema 2.2. *Funkcija $C(s,t)$ je kovarijaciona funkcija nekog Gausovog procesa akko* $\forall(t_1,\dots,t_n) \in T^n \quad \forall(a_1,\dots,a_n) \in \mathbb{R}^n \qquad \sum_{i,j=1}^{n} a_i C(t_i,t_j)\, a_j \geq 0$ *tj. matrica* $\left[C(t_i,t_j)\right]_{i,j=1}^{n}$ *je pozitivno definitna.*

Funkcije $F(t_1,\dots,t_n;x_1,\dots,x_n)$ su funkcije raspodele n-dimenzione normalne raspodele čija je matrica kovarijanse matrica $\left[C(t_i,t_j)\right]_{i,j=1}^{n}$ iz **Teoreme 2.2.** i čije je očekivanje $(b(t_1),\dots,b(t_n))$.

Primer 2.3. *(Vinerov proces)* Gausov slučajni process $\{X_t\}_{t\in T}$ za koju važi $T = [0,+\infty)$, $b(t) = 0$ $C(s,t) = E\,(X_s X_t) = min\,(s,t) \quad \forall s,t \in T$, naziva se Vinerov proces (ili process Braunovog kretanja).

Ime nosi po Norbertu Vineru (Norbert Wiener, 1894-1964.) američkom matematičaru koji ga je formalno opisao, mada je sam proces uočio ranije škotski biolog Robert Braun (Robert Brown, 1773-1858.) posmatrajući kretanje čestica polena u vodi.

Jedno od korisnih svojstava Vinerovog procesa je sledeće:

Teorema 2.3. *Za Vinerov slučajni proces* $\{X_t\}_{t\in[0,+\infty)}$ *važi:*

$(\forall s < t \leq u < v)\; E\big((X_t - X_s)(X_u - X_v)\big) = E(X_t - X_s)E(X_u - X_v)$

Dokaz:

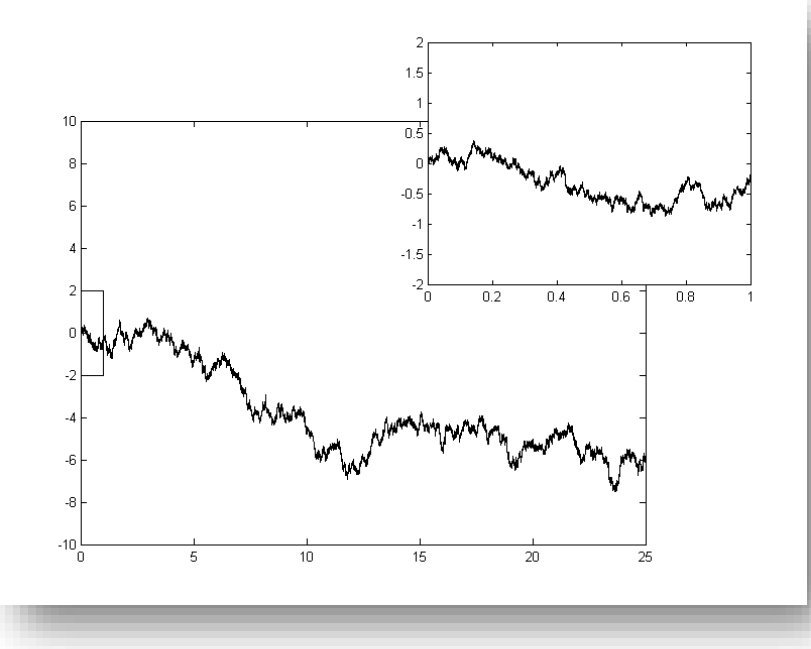
$$E\big((X_t - X_s)(X_u - X_v)\big) =$$
$$= E(X_t X_u) - E(X_t X_v) - E(X_s X_u) + E(X_s X_v) =$$
$$= min(t,u) - min(t,v) + min(s,v) - min(s,u) =$$
$$= \underbrace{min(t,u) - min(s,u)}_{t-s} + \underbrace{min(s,v) - min(t,v)}_{s-t} =$$
$$= t - s + s - t =$$
$$= 0$$

■

$E(X_t - X_s) = E(X_u - X_v) = 0$ (Ovo potiče iz činjenice da su priraštaji Vinerovog, a opštije

Gausovog slučajnog procesa, slučajne promenljive $X_t - X_s$ nezavisne, normalno raspodeljene sa očekivanom vrednošću 0).

Jedna od zanimljivosti Vinerovog slučajnog procesa je da su mu trajektorije neprekidne, a nigde diferencijabilne funkcije. To potiče od toga što se u neku ruku te trajektorije dobijaju graničnim postupkom nasumične šetnje. Jedan primer Vinerove trajektorije vidimo na *Slika 2.2*:



Slika 2.2 - Slika preuzeta iz [21]

Primer 2.4. *(Poasonov slučajni proces)* Slučajni proces $\{X_t\}_{t\in T}$ je Poasonov sa stopom rasta λ, $\lambda > 0$ ako:

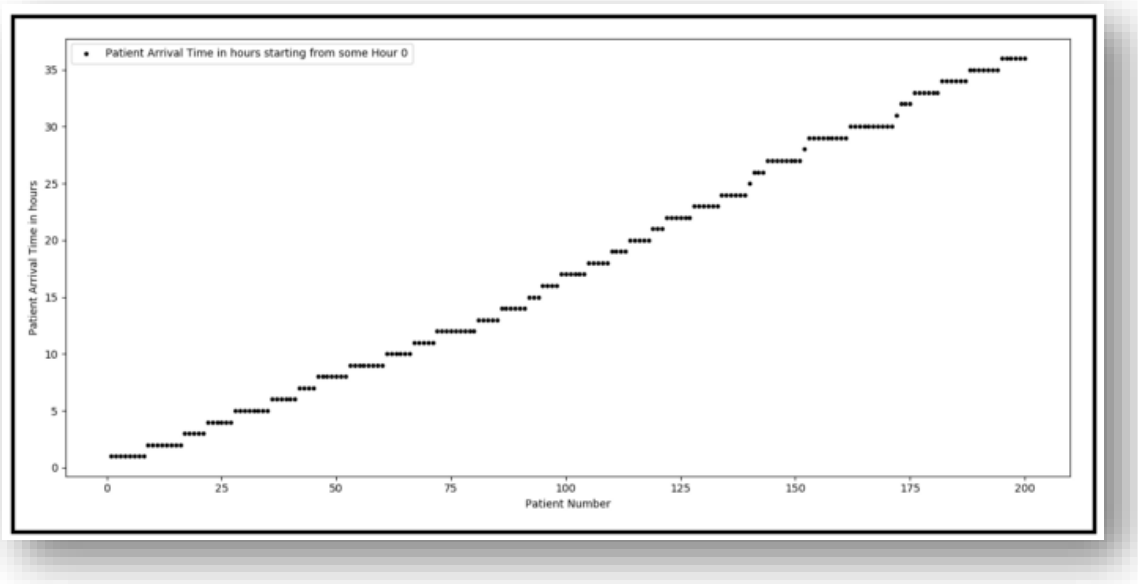
a) $T = [0, +\infty)$ i $P\{X_0 = 0\} = 1$;

b) $X_t - X_s$ i $X_u - X_v$ su nezavisne slučajne promenljive za $s < t \leq u < v$;

c) $(\forall s, t \in [0, +\infty))$ $(s < t)$ $X_t - X_s$ ima Poasonovu raspodelu sa parametrom $\lambda(t - s)$.

Ovi slučajni procesi se u praksi koriste pri modeliranju događaja za koje je mala verovatnoća da ih se realizuje veći broj u kraćem vremenskom intervalu (broj poziva na telefon, prolazak autobusa kroz stanicu i sl.)

Jedna Poasonova trajektorija je:



Slika 2.3 - Slika preuzeta iz [22]

Slika 2.3 opisuje broj pristiglih pacijenata u bolnicu mereno od fiksnog trenutka.

2.3 Markovljevi procesi

Markovljevi procesi nose ime po Ruskom matematičaru Andreju Markovu (Андрей Андреевич Марков, 1856-1922.) koji je napisao prvi rad na tu temu 1907. pri statističkoj analizi pesme "Evgenije Onjegin".

Definicija 2.7. *Slučajni proces* $\{X_t\}_{t\in T}$ *je Markovljev proces ako:*
$(\forall\, n \in \mathbb{N})\ \forall (s_1, \dots, s_n) \in T^n$ *td.* $s_1 < s_2 < \cdots < s_n < t$ *važi* $P\{X_t \in B | X_{s_1}, \dots, X_{s_n}\} = P\{X_t \in B | X_{s_n}\}$ *za svaki Borelov skup* $B \in A$.

Još jedan primer je veličina neke populacije u jedinici vremena, jer buduća veličina populacije zavisi od veličine buduće populacije samo preko veličine sadašnje populacije.

Teorema 2.4. *Neka je $\{X_t\}_{t\in T}$ slučajni proces takav da $T=[0,+\infty)$, $X_0=0$ i X_t-X_s su nezavisne slučajne promenljive. Tada je $\{X_t\}_{t\in T}$ Markovljev proces.*

Dokaz: Neka je $s_1<s_2<\cdots<s_n<t$ i B Borelov skup

$$X_0=0$$
$$\downarrow$$
$$P\{X_t\in B|X_{s_1}=x_1,\dots,X_{s_n}=x_n\}=$$
$$=P\{X_t-X_{s_n}\in B-x_n|X_{s_1}-X_0=x_1,X_{s_2}-X_{s_1}=x_2-x_1,\dots,X_{s_n}-X_{s_{n-1}}=x_n-x_{n-1}\}=$$
$$=P\{X_t-X_s\in B-x_n\}=P\{X_t-X_s\in B-x_n|X_{s_n}-X_0=x_n\}=P\{X_t\in B|X_{s_n}=x_n\}$$
$$\Downarrow$$
$\{X_t\}_{t\in T}$ je Markovljev proces. ■

Posledica 2.5. *Poasonov proces je Markovljev proces.*

Teorema 2.6. *Vinerov proces je Markovljev proces.*

Dokaz: $E(X_sX_t)=min(s,t)$

$$\Downarrow$$
$$E(X_0^2)=min(0,0)=0$$
$$\Downarrow$$
$D(X_0)=E(X_0^2)-\big(E(X_0)\big)^2=0-0=0 \quad \Rightarrow$ X_0 je konstantna slučajna promenljiva

$\Rightarrow P\{X_0=0\}=1$ i X_t-X_s su nezavisne slučajne promenljive

$\overset{T\,2.4}{\Longrightarrow}$ Vinerov slučajni proces je Markovljev proces ■

Nalaženje konačnodimenzionih raspodela Markovljevih procesa opisan je sa:

Teorema 2.7. *Neka je $\{X_t\}_{t\in T}$ diskretan Markovljev proces. Neka je $t\geq t_0\in T$ za svako $t\in T$. Tada je:*

$$P\{X_{t_1}=x_1,X_{t_2}=x_2,\dots,X_{t_n}=x_n\}=$$
$$=\sum_j P\{X_{t_0}=x_j\}P\{X_{t_1}=x_1|X_{t_0}=x_j\}P\{X_{t_2}=x_2|X_{t_1}=x_1\}\dots P\{X_{t_n}=x_n|X_{t_{n-1}}=x_{n-1}\}$$

gde je $t_1<t_2<\cdots<t_n\in T$ i $x_1,\dots,x_n$ dozvoljene vrednosti za $X_0\overset{\text{def}}{=}X_{t_0}$

Dokaz: formula uslovne verovatnoće

$$\downarrow$$

$$P\{X_{t_1} = x_1, \dots, X_{t_n} = x_n\} = P\{X_{t_n} = x_n | X_{t_1} = x_1, \dots, X_{t_{n-1}} = x_{n-1}\} \cdot P\{X_{t_1} = x_1, \dots, X_{t_{n-1}} = x_{n-1}\}$$

Markovljev proces $\rightarrow$ $= P\{X_{t_n} = x_n | X_{t_{n-1}} = x_{n-1}\} \cdot P\{X_{t_1} = x_1, \dots, X_{t_{n-1}} = x_{n-1}\}$

$$= P\{X_{t_n} = x_n | X_{t_{n-1}} = x_{n-1}\} \cdot P\{X_{t_{n-1}} = x_{n-1} | X_{t_1} = x_1, \dots, X_{t_{n-2}} = x_{n-2}\} \dots$$

Odatle sukcesivnom primenom dobijamo tvrđenje. ■

Analogna verzija **Teoreme 2.7.** važi i za neprekidne Markovljeve procese.

2.4 Markovljevi lanci

Definicija 2.8. *Markovljevi lanci su diskretni Markovljevi procesi koji mogu uzimati konačno mnogo različitih vrednosti. Dakle, slučajni proces je Markovljev lanac ako važi:*

$$P\{X_{k+1} = i | X_0, X_1, \dots, X_k\} = P\{X_{k+1} = i | X_k\}$$

Po tvrđenju **Teorema 2.7.** za opisivanje Markovljevog lanca dovoljno je znati početnu raspodelu $p_0(i) \stackrel{\text{def}}{=} P\{X_0 = i\}$, kao i verovatnoće prelaza $p_{ij}(k,l) \stackrel{\text{def}}{=} P\{X_l = j | X_k = i\}$.

Definicija 2.9. *Markovljev lanac je homogen ako verovatnoće prelaza* $p_{ij}(k,l)$ *zavise samo od razlike* $l - k$.

Za homogene Markovljeve lance dovoljno je posmatrati verovatnoće $p_{ij} = P\{X_{k+1} = j | X_k = i\}$, $k = 0,1, \dots$ $i, j = 1,2, \dots, n$ tj. matricu $[p_{ij}]_{i,j=1}^{n}$ koja se naziva *matrica prelaza Markovljevog lanca.*

Da je Markovljev lanac potpuno određen polaznim vektorom $p^{(0)}$ i matricom prelaza pokazuje sledeća teorema.

Teorema 2.8. *Neka je* $p^{(0)}$ *vektor početnih verovatnoća i* Π *matrica prelaza Markovljevog lanca* X_k. *Tada je* $p^{(k)} = p^{(0)}\Pi^k$ $k = 1,2, \dots$ *gde je* $p^{(k)} = (p_1^{(k)}, p_2^{(k)}, \dots, p_n^{(k)})$, *pri čemu je* $p_i^{(k)} = P\{X_k = i\}$.

Dokaz: $k \in \{0,1,\dots\}$

$$p_j^{(k+1)} = P\{X_{k+1} = j\} = \sum_{i=1}^{n} P\{X_{k+1} = j | X_k = i\} \cdot P\{X_k = i\} = \sum_{i=1}^{n} p_i^{(k)}\, p_{ij}$$

Dakle, $p^{(k+1)} = p^{(k)}\Pi = p^{(k-1)}\Pi^2 = \cdots = p^{(0)}\Pi^k$. ■

Primer 2.5. Neka su 2 bele i 2 crne kuglice u dve kutije. Proces je definisan tako što u svakom koraku uzimamo po jednu kuglicu iz svake kutije i stavljamo u drugu. Neka je X_n-stanje sistema posle n koraka.

X_n može biti:
1. 2 crne – 2 bele
2. 1 crna, 1 bela – 1 bela, 1 crna
3. 2 bele – 2 crne

X_n je homogeni Markovljev lanac.

Matrica prelaza je $\Pi = \begin{bmatrix} 0 & 1 & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ 0 & 1 & 0 \end{bmatrix}$.

Kako je početni vektor $p^{(0)} = [0,1,0]$, to je npr.

$$p^{(11)} = [0,1,0]\Pi^{10} = [0,1,0]\begin{bmatrix} \frac{171}{1024} & \frac{341}{512} & \frac{171}{1024} \\ \frac{341}{2048} & \frac{683}{1024} & \frac{341}{2048} \\ \frac{171}{1024} & \frac{341}{512} & \frac{171}{1024} \end{bmatrix} = \left[\frac{341}{512}, \frac{683}{1024}, \frac{341}{512}\right]$$

3. Monte Karlo integracija

3.1 Osnovne ideje i ilustrativni primeri pri integraciji funkcije jedne promenljive

Posmatrajmo, za početak, Rimanov integral funkcije jedne promenljive $\int_a^b f(x)dx$.

Pri aproksimaciji ovih integrala imamo na raspolaganju brojne kvadraturne formule, koje su vrlo precizne kada je podintegralna funkcija dovoljnog kvaliteta:

a) Formula srednje tačke:

$$a < x_1 < x_2 < \cdots < x_{n-1} < b \qquad h \overset{\text{def}}{=} x_i - x_{i-1}$$

$$\int_a^b f(x)dx \approx h\left(f\left(\frac{x_1 - a}{2}\right) + f\left(\frac{x_2 - x_1}{2}\right) + \cdots + f\left(\frac{b - x_{n-1}}{2}\right)\right)$$

b) Trapezna formula

$$a < x_1 < x_2 < \cdots < x_{n-1} < b \qquad h \overset{\text{def}}{=} x_i - x_{i-1}$$

$$\int_a^b f(x)dx \approx h\left(\frac{f(a) + f(b)}{2} + f(x_1) + \cdots + f(x_{n-1})\right)$$

sa ocenom greške $R \leq \left|\frac{b-a}{12}h^2 f''(\xi)\right| \quad \xi \in [a, b]$

c) Simpsonova formula

$$a < x_1 < x_2 < \cdots < x_{2n-1} < b \qquad h \overset{\text{def}}{=} x_i - x_{i-1}$$

$$\int_a^b f(x)dx \approx \frac{h}{3}\Big(f(a) + f(b) + 4\big(f(x_1) + f(x_3) + \cdots + f(x_{2n-1})\big) + 2\big(f(x_2) + f(x_4) + \cdots + f(x_{2n-2})\big)\Big)$$

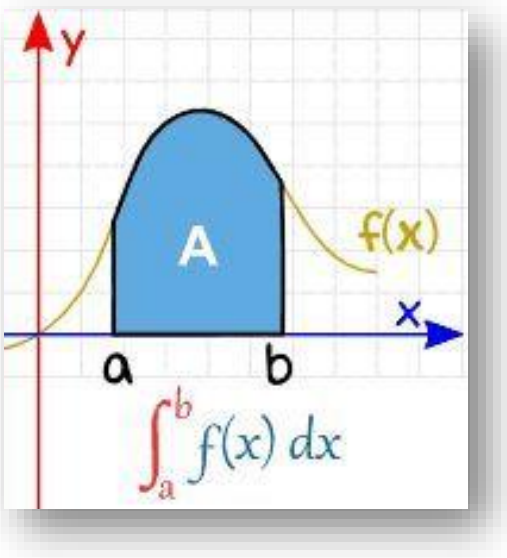
sa ocenom greške $R \leq \left|\frac{b-a}{180}h^4 f^{(\mathrm{IV})}(\xi)\right| \quad \xi \in [a, b]$

Monte Karlo metode integracije funkcije jedne promenljive su zadivljujuće prosečne, s'obzirom na svoju jednostavnost. Naravno, u jednoj dimenziji one nisu ni približno precizne kao kvadraturne formule, sem ako funkcija f nije dovoljno kvalitetna (klase C^1, samo neprekidna ili samo Riman integrabilna).

U ovoj podsekciji ćemo dati dve osnovne ideje za Monte Karlo integraciju, od kojih ćemo jednu naknadno razraditi u sledećoj podsekciji.

Prva ideja je znatno u korelaciji sa pristupom iz primera korišćenog u prvom poglavlju pri aproksimaciji broja π.

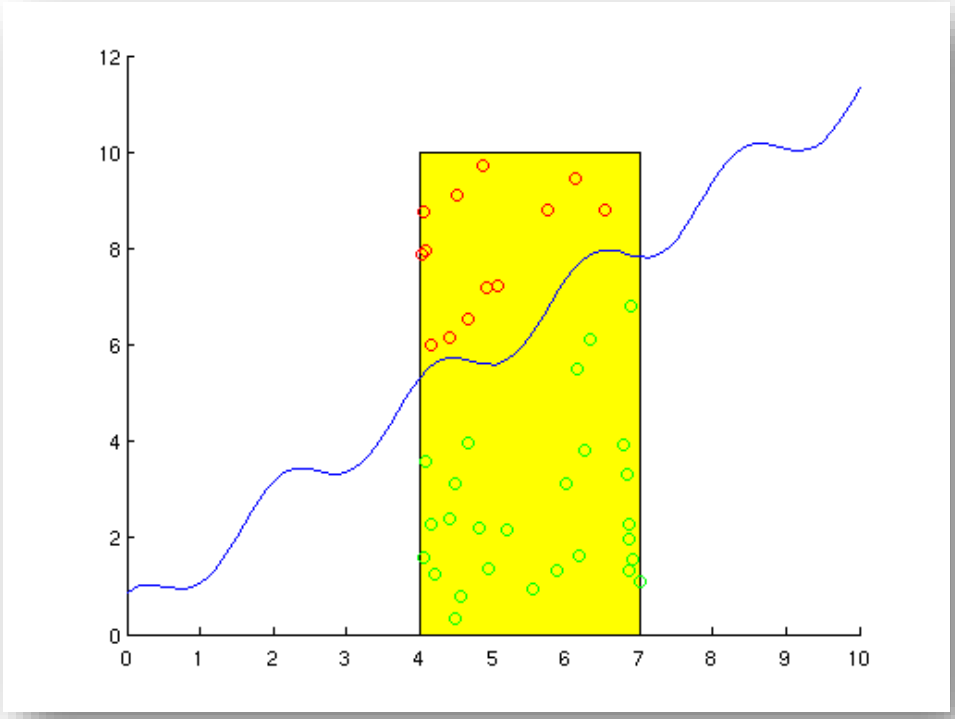
Osnovna geometrijska interpretacija određenog integrala je površina ispod krive što možemo videti sa *Slika 3.1*:



Slika 3.1 - Slika preuzeta iz [23]

I ideja je sasvim slična ideji iz aproksimacije broja π:

Uzeti dovoljno veliki pravougaonik čija je dužina $[a, b]$, a širina dovoljna da obuhvati površinu ispod krive $f(x)$ na $[a, b]$. Zatim se nasumično bira niz tačaka iz tog pravougaonika. Upoređivanjem broja tačaka ispod i iznad grafika i množenjem sa površinom pravougaonika dobija se aproksimacija integrala:



Slika 3.2

Ovakav algoritam se zove pogodak-promašaj algoritam (veoma podseća na gađanje u pikado metu).

Primer 3.1. Primenimo prethodnu ideju na aproksimaciju integrala

$$\int_0^2 e^{-x^2} dx \approx 0{,}8820813907624217$$

Nasumično ćemo birati tačke iz oblasti $[0,2] \times [0,1]$ i sprovesti pogodak-promašaj algoritam. C-kod algoritma izgleda ovako:

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>



int main(int argc, char *argv[]) {
double x,y;
int i, N;
int s=0;
scanf("%d",&N);
for(i=0;i<N;i++){
double x=((double)rand() * 2.0 ) / (double)RAND_MAX ;
double y=((double)rand() * 1.0 ) / (double)RAND_MAX ;
if(y<=exp(-x*x))
s++;
}
printf("%f",(2.0)*s/N);

return 0;
}
```

Kod 3.1

Dobijeni rezultati su:

n	10	100	1000	10000	100000	1000000	10000000	100000000	1000000000
rezultati	1,200000	1,000000	0,892000	0,867200	0,879220	0,879654	0,881953	0,881884	0,882095

Tabela 3.1

Dakle, kao što je i najvljeno, algoritam je prilično prosečan za standardne ulaze i tek na unosu od milijardu uzoraka dostiže tačnost 10^{-5}. Tu tačnost bismo kvadraturnom formulom bilo kog tipa znatno brže dobili.

Za razliku od geometrijski jasnog i sasvim jedinstvenog pogodak-promašaj Monte Karlo algoritma, imamo malo složeniji algoritam koji je takođe prosečan u jednoj dimenziji, ali je neuporedivo kvalitetniji i efikasniji od kvadraturnih formula u većim dimenzijama – Sirovi Monte Karlo algoritmi.

Ilustrujmo sirovi Monte Karlo metod integracije na integralu funkcije jedne promenljive $\int_a^b f(x)\,dx$. Znamo da važi sledeće tvrđenje:

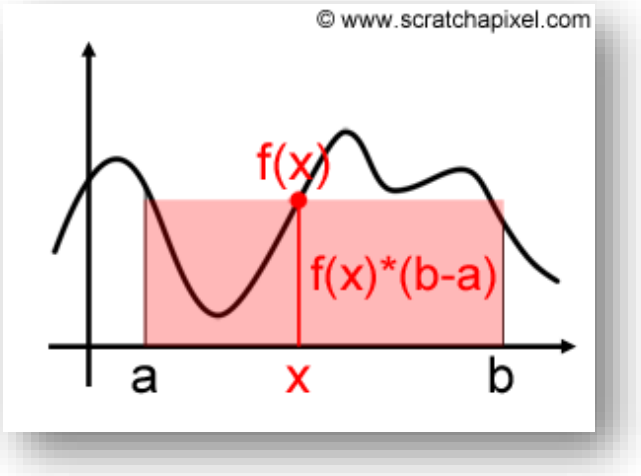
Teorema 3.1. *(Teorema o srednjoj vrednosti integrala) Neka je f Riman integrabilna na $[a,b]$ i $m \overset{\text{def}}{=} \inf_{x\in[a,b]} f(x)$ i $M \overset{\text{def}}{=} \sup_{x\in[a,b]} f(x)$, tada postoji $\mu \in [m,M]$, takva da je:*

$$\int_a^b f(x)\,dx = \mu(b-a)$$

Sledeći ovo tvrđenje, to μ je neka srednja vrednost funkcije f na intervalu $[a,b]$. Ako odaberemo $x_1, x_2, \dots, x_N \in [a,b]$ ravnomerno raspodeljene, tada možemo srednju vrednost μ zameniti izrazom $\sum_{i=1}^N f(x_i)$, te dobijamo formulu:

$$\int_a^b f(x)\,dx \approx \frac{b-a}{N}\sum_{i=1}^N f(x_i)$$

Ako posmatramo *Slika 3.3*:



Slika 3.3 - Slika preuzeta iz [10]

Jasno se vidi da kada $N \to \infty$ mi pokrivamo traženu površinu pravougaonika visine $f(x_i)$ i dužine $b-a$. Odnosno kada $N \to \infty$, razmak između nasumično biranih $x_i \in [a,b]$ teži 0, te bukvalno dobijamo samu definiciju Rimanovog integrala $\int_a^b f(x)\,dx$. Ovakva metoda predstavlja sirovu Monte

Karlo integraciju.

Primer 3.2. Aproksimaciju broja π iz prvog poglavlja smo zapravo vršili aproksimacijom četvrtine kruga, a to je problem koji možemo svesti na problem numeričke integracije, tako što je zapravo:

$$\int_0^1 \sqrt{1-x^2}\,dx = \frac{\pi}{4} \approx 0{,}78539816333$$

Sprovedimo sirovi Monte Karlo algoritam na računanje integrala $\int_0^1 \sqrt{1-x^2}\,dx$.

Za proizvoljno N biraćemo N nasumičnih tačaka $x_1, x_2, \ldots, x_N$ iz intervala $[0,1]$, sabrati vrednosti funkcije $\sqrt{1-x^2}$ u tim tačkama i podeliti sa N. C-kod koji sprovodi ovaj algoritam vidimo na *Kod 3.2*:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>



int main(int argc, char *argv[]) {

int i,N;
double s=0.0;
scanf("%d",&N);
for(i=0;i<N;i++){
double d1=((double)rand() * 1.0 ) / (double)RAND_MAX ;
s+=sqrt(1-pow(d1,2));
}
printf("%f",1.0*s/N);

return 0;
}
```

Kod 3.2

Dobijamo sledeće rezultate:

N	10	100	1000	10000	100000	1000000	10000000	100000000
rezultati	0,769804	0,788769	0,786987	0,782655	0,784452	0,785346	0,785395	0,785335

Tabela 3.2

Kao i u **Primru 3.1.** vidimo da metoda sporije postiže veliku tačnost i da rezultati variraju nepravilno zavisno od unosa N.

Jedna modifikacija prethodnog algoritma bi bilo sprovođenje istog algoritma fiksiran broj puta i uzimanje aritmetičke sredine dobijenih rezultata.

3.2 Sirovi Monte Karlo algoritam, definicija i ocena greške

U prethodnom poglavlju smo uveli sirovi Monte Karlo algoritam za računanje Rimanovog integrala $\int_a^b f(x)\,dx$.

$$\int_a^b f(x)\,dx \approx \frac{b-a}{N}\sum_{i=1}^{N} f(x_i)$$

Teorema 3.2.

a)

$$E\left(\frac{b-a}{N}\sum_{i=1}^{N} f(x_i')\right) = \int_a^b f(x)\,dx$$

b)

$$P\left\{\lim_{N\to\infty}\frac{b-a}{N}\sum_{i=1}^{N} f(x_i') = \int_a^b f(x)\,dx\right\} = 1$$

Dokaz:

a) x_i su nezavisni, uniformno raspodeljeni na $[a,b]$

↓

$$E\left(\frac{b-a}{N}\sum_{i=1}^{N} f(x_i')\right) = \frac{b-a}{N}E\left(\sum_{i=1}^{N} f(x_i')\right) = \frac{b-a}{N}\cdot N\cdot E\big(f(x_i')\big) = (b-a)\int_a^b \frac{f(x)}{b-a}\,dx$$

$$= \int_a^b f(x)\,dx$$

b) Kako je:

$$E\left(\frac{b-a}{N}\sum_{i=1}^{N} f(x_i')\right) = \int_a^b f(x)\,dx$$

imamo da tvrđenje direktno sledi po zakonu velikih brojeva.

■

U prethodnom poglavlju smo videli da su kvadraturne formule vrlo efikasne u jednoj dimenziji, dok je sirova Monte Karlo integracija prilično prosečna. Međutim, s povećanjem broja dimenzije, numeričko računanje integrala postaje znatno složenije. Recimo ako je za računanje jednodimenzionog integrala potrebno 10 tačaka, tada bi za računanje integrala funkcije 100 promenljivih bilo potrebno računati vrednosti funkcije u 10^{100} tačaka. Ovaj fenomen da je klasa problema lako rešiva u nižim, a neuporedivo teža ili nerešiva u višim zove se prokletstvo dimenzije.

A naravno, i sam višedimenzioni integral može imati vrlo komplikovane granice. Tu sirova Monte Karlo metoda pokazuje enormno bolje rezultate. Da bismo preformulisali sirovu Monte Karlo integraciju iz sekcije 3.1. u višedimenzioni koristimo:

Teorema 3.3. *(O srednjoj vrednosti) Neka je* $D \subset \mathbb{R}^n$ *merljiv skup i* f *Riman integrabilna na* D*. Neka su* $m \stackrel{\text{def}}{=} \inf_{x\in D} f(x)$ *i* $M \stackrel{\text{def}}{=} \sup_{x\in D} f(x)$*. Tada postoji* $\xi \in [m, M]$ *takva da:*

$$\int_D f(x)\,dx = \xi \cdot \mu(D)$$

$$\downarrow$$

Mera skupa D

Sada potpuno analogno kao u sekciji 3.1. izvodimo formulu za sirovu Monte Karlo integraciju:

$$\int_D f(x_1, \dots, x_n)\,d\mu \approx \frac{\sum_{i=1}^{N} f\left(x_1^{(i)}, \dots, x_n^{(i)}\right)}{N}\mu(D)$$

Teorema 3.4.

a)

$$E\left(\frac{\mu(D)}{N}\sum_{i=1}^{N} f\left(x_1^{(i)}, \dots, x_n^{(i)}\right)\right) = \int_D f(x_1, \dots, x_n)\, d\mu$$

b)

$$P\left\{\lim_{N\to\infty} \frac{\mu(D)}{N}\sum_{i=1}^{N} f\left(x_1^{(i)}, \dots, x_n^{(i)}\right) = \int_D f(x_1, \dots, x_n)\, d\mu\right\} = 1$$

Dokaz:

a)

Sve tačke su uniformno raspodeljene u oblasti $D \subset \mathbb{R}^n$

↓

$$E\left(\frac{\mu(D)}{N}\sum_{i=1}^{N} f\left(x_1^{(i)}, \dots, x_n^{(i)}\right)\right) = \frac{\mu(D)}{N} E\left(\sum_{i=1}^{N} f\left(x_1^{(i)}, \dots, x_n^{(i)}\right)\right) = \frac{\mu(D)}{N} \cdot N \cdot E\left(f\left(x_1^{(i)}, \dots, x_n^{(i)}\right)\right) =$$

$$= \mu(D) \int_D \frac{f(x_1, \dots, x_n)}{\mu(D)} d\mu = \int_D f(x_1, \dots, x_n)\, d\mu$$

b) Direktno sledi iz zakona velikih brojeva (zbog dela pod a)).

■

Sada želimo da ocenimo grešku ove integracije. Za ocenu greške koristićemo varijansu, tj. disperziju:

Teorema 3.5.

$$Var\left(\frac{\mu(D)}{N}\sum_{i=1}^{N} f\left(x_1^{(i)}, \dots, x_n^{(i)}\right)\right) = \frac{\mu^2(D)}{N} Var(f)$$

Dokaz:

Uzorci su nezavisni i uniformno raspodeljeni na D

↓

$$Var\left(\frac{\mu(D)}{N}\sum_{i=1}^{N} f\left(x_1^{(i)}, \dots, x_n^{(i)}\right)\right) = \frac{\mu^2(D)}{N^2} Var\left(\sum_{i=1}^{N} f\left(x_1^{(i)}, \dots, x_n^{(i)}\right)\right)$$

$$= \frac{\mu^2(D)}{N^2} \cdot N \cdot Var\left(f\left(x_1^{(i)}, \dots, x_n^{(i)}\right)\right) =$$

$$= \frac{\mu^2(D)}{N} Var(f)$$

■

Direktna posledica prethodnog tvrđenja daje nam standardnu devijaciju:

$$\sigma\left(\frac{\mu(D)}{N}\sum_{i=1}^{N} f\left(x_1^{(i)}, \dots, x_n^{(i)}\right)\right) = \frac{\mu(D)\sigma(f)}{\sqrt{N}}$$

Ova formula nam pokazuje značaj sirove Monte Karlo metode, a to je da greška u potpunosti ne zavisi od dimenzije prostora u kome se integrali, te je jasno zašto je ova metoda efikasnija od standardnih kvadraturnih formula u višim dimenzijama. Međutim, red konvergencije metode je $\mathcal{O}\left(\frac{1}{\sqrt{N}}\right)$, tj. potrebno je kvadrirati broj ulaza za prepolovljenje greške.

3.3 Modifikacije

Očigledno je da modifikacije koje treba da smanje grešku sirove Monte Karlo metode moraju dati manju varijansu. Jedna od ideja je tzv. uzorovanje po značajnosti. Metod čija je glavna ideja da se tačke $\mathrm{x}_1, \dots, \mathrm{x}_\mathrm{N}$ ne uzimaju uniformno, već da izaberemo takve tačke u kojima $f(x)$ uzima veće vrednosti.

U aproksimaciji se teži da se $f(x)$ zameni sa $\frac{f(x)}{\rho(x)}$, gde je $\rho(x)$ gustina raspodele koja je skoncentrisana oko tačaka u kojima $f(x)$ uzima veće vrednosti.

$$\int_D f(x)\, d\mu \approx \frac{\mu(D)}{N}\sum_{i=1}^{N}\frac{f(x_i)}{\rho(x_i)} \qquad x_1, \dots, x_N \in D \qquad D \subset \mathbb{R}^m$$

Idealan izbor je kada $f(x) \sim \rho(x)$, jer je tada varijansa minimalna.

4. Primena Monte Karlo simulacija za generisanje slučajnih promenljivih

4.1 Generisanje diskretnih raspodela

Pri generisanju slučajnih promenljivih Monte Karlo simulacija koristićemo niz $\{U_n\}_{n=1}^{+\infty}$ (u programima i simulacijama biće ograničene veličine) slučajnih promenljivih takav da $(\forall n \in \mathbb{N}) \quad U_n \colon \mathcal{U}(0,1)$.

Teorema 4.1. *Ako* $X \colon \mathcal{U}(0,1)$, *tada* $1 - X \colon \mathcal{U}(0,1)$.

Dokaz:

$$F_{1-X}(x) = P\{1 - X \leq x\} = P\{-X \leq x - 1\} = P\{X \geq 1 - x\} =$$

$$= \begin{cases} 0, & 1 - x > 1 \\ 1, & 1 - x < 0 \\ \int_0^{1-x} dt\,, & 0 \leq 1 - x \leq 1 \end{cases}$$

$$= \begin{cases} 0, & x < 0 \\ 1 - x, & 0 \leq x \leq 1 \\ 1, & x > 1 \end{cases}$$

$$\downarrow$$

A ovo je funkcija raspodele slučajne promenljive iz $\mathcal{U}(0,1)$

$$\Rightarrow \quad 1 - X \colon \mathcal{U}(0,1)$$

■

Ukoliko raspodela može uzimati konačno mnogo vrednosti, generisanje slučajne promenljive se vrši na sledeći način:

Neka su $x_1, \dots, x_n$ odgovarajuće vrednosti, a $p_1, \dots, p_n$ odgovarajuće verovatnoće, respektivno. Tada

delimo $[0,1]$ na podintervale $\{A_j\}_{j=1}^{n}$.

$$[0,1] = \underbrace{[0, p_1)}_{A_1} \cup \underbrace{[p_1, p_1 + p_2)}_{A_2} \cup \underbrace{[p_1 + p_2, p_1 + p_2 + p_3)}_{A_3} \cup \ldots \cup \underbrace{[p_1 + p_2 + \cdots + p_{n-1}, 1]}_{A_n}$$

i definišemo $x = x_i$ ako $\mathcal{U} \in A_i$.

Primer 4.1. Ako želimo da simuliramo bacanje homogene kocke, tada možemo poći od niza $\{U_n\}_{n=1}^{+\infty}$ slučajnih brojeva uniformno raspodeljenih u $(0,1)$. I niz

$$X_n \overset{\text{def}}{=} \begin{cases} 1, & U_n \in \left(0, \frac{1}{6}\right) \\ 2, & U_n \in \left[\frac{1}{6}, \frac{1}{3}\right) \\ 3, & U_n \in \left[\frac{1}{3}, \frac{1}{2}\right) \\ 4, & U_n \in \left[\frac{1}{2}, \frac{2}{3}\right) \\ 5, & U_n \in \left[\frac{2}{3}, \frac{5}{6}\right) \\ 6, & U_n \in \left[\frac{5}{6}, 1\right) \end{cases}$$

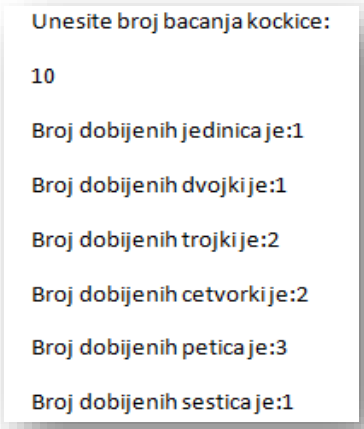
će nam dati simulaciju niza bacanja kockice 6 puta.

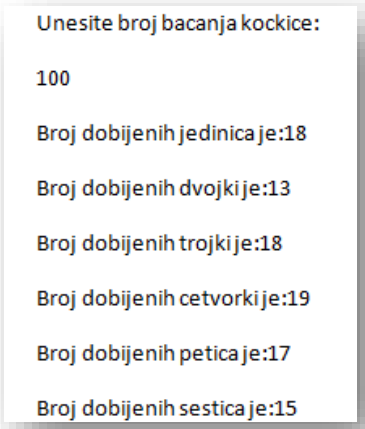
C-kod koji testira ovu simulaciju je:

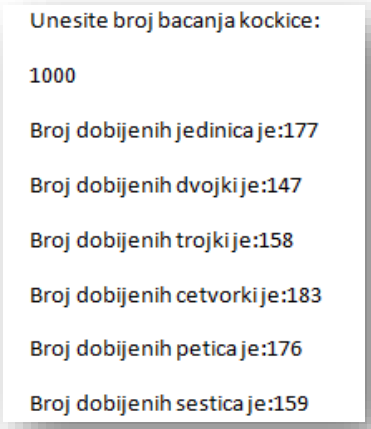
```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {

int i,n;
int jedinice=0, dvojke=0, trojke=0, cetvorke=0,petice=0, sestice=0;
printf("Unesite broj bacanja kockice:\n");
scanf("%d",&n);
double a[n];
for(i=0;i<n;i++)
a[i]=((double)rand() * 1.0 ) / (double)RAND_MAX ;
for(i=0;i<n;i++){
if(a[i]<(1.0/6))
jedinice++;
else if (a[i]>=(1.0/6)&&a[i]<(1.0/3))
dvojke++;
else if (a[i]>=(1.0/3)&&a[i]<0.5)
trojke++;
else if (a[i]>=0.5&&a[i]<(2.0/3))
cetvorke++;
else if (a[i]>=(2.0/3)&&a[i]<(5.0/6))
petice++;
else
sestice++;
}

printf("Broj dobijenih jedinica je:%d\n",jedinice);
printf("Broj dobijenih dvojki je:%d\n",dvojke);
printf("Broj dobijenih trojki je:%d\n",trojke);
printf("Broj dobijenih cetvorki je:%d\n",cetvorke);
printf("Broj dobijenih petica je:%d\n",petice);
printf("Broj dobijenih sestica je:%d\n",sestice);
return 0;
}
```

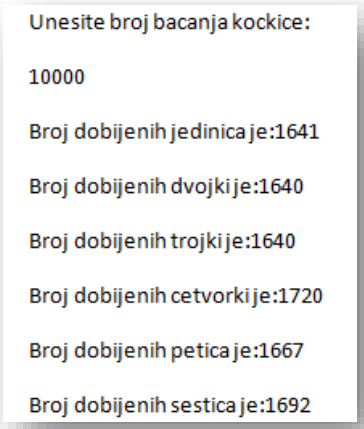
Kod 4.1

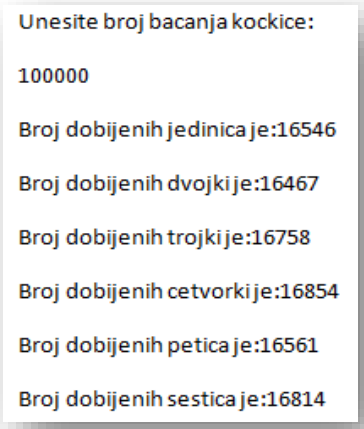
Kod 4.1 nam je dao sledeće rezultate:











Potpuno analogno bismo mogli konstruisati niz bacanja nehomogene kockice.

Primer 4.2. *(Generisanje binomne raspodele)* Binomna raspodela $\mathcal{B}(n,p)$ predstavlja broj uspešnih izvođenja eksperimenata u n pokušaja, gde je verovatnoća jednog uspeha p.

Za generisanje binomne raspodele preko Monte Karlo simulacije ključno je da $\mathcal{B}(n,p) = I_1 + \cdots + I_n$, tj. da je binomna raspodela jednaka zbiru indikatora uspeha u određenom pokušaju.

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

Ako imamo niz $\{U_n\}_{n=1}^{+\infty}$ uniformno raspodeljenih brojeva u $(0,1)$, tada definišemo niz $\{X_n\}_{n=1}^{+\infty}$.

$$X_n \overset{\text{def}}{=} \begin{cases} 1, & ako\ je\ U_n < p \\ 0, & ako\ je\ U_n \geq p \end{cases}$$

I tada će slučajna promenljiva $X = \sum_{j=1}^{n} X_j$ imati binomnu $\mathcal{B}(n,p)$ raspodelu.

C-kod kojim ilustrujemo ovu simulaciju je:

```
#include <stdio.h>
#include <stdlib.h>


int main(int argc, char *argv[]) {
int i,n;
int s=0;
float p;
printf("Unesite broj eksperimenata:\n");
scanf("%d",&n);
double a[n];
printf("Unesite verovatnocu uspeha u pojedinacnom eksperimentu:\n");
scanf("%f",&p);
for(i=0;i<n;i++){
a[i]=((double)rand() * 1.0 ) / (double)RAND_MAX ;
}
for(i=0;i<n;i++){
if(a[i]<p)
s++;
}
printf("Broj uspenih izvodjenja u %d eksperimenata je %d",n,s);

return 0;
}
```

Kod 4.2

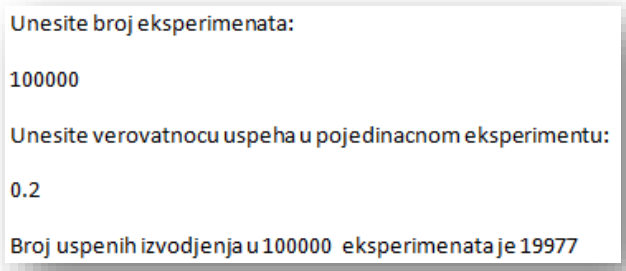
Dobijeni su sledeći rezultati od *Kod 4.2*:

```
Unesite broj eksperimenata:

10

Unesite verovatnocu uspeha u pojedinacnom eksperimentu:

0.7

Broj uspenih izvodjenja u 10 eksperimenata je 6
```

```
Unesite broj eksperimenata:

100

Unesite verovatnocu uspeha u pojedinacnom eksperimentu:

0.4

Broj uspenih izvodjenja u 100 eksperimenata je 40
```

```
Unesite broj eksperimenata:

1000

Unesite verovatnocu uspeha u pojedinacnom eksperimentu:

0.3

Broj uspenih izvodjenja u 1000 eksperimenata je 295
```

```
Unesite broj eksperimenata:

10000

Unesite verovatnocu uspeha u pojedinacnom eksperimentu:

0.8

Broj uspenih izvodjenja u 10000 eksperimenata je 7968
```



Primer 4.3. *(Generisanje geometrijske raspodele)* Geometrijsku distribuciju $\mathcal{G}(p)$ posmatramo kao broj izvedenih eksperimenata pre prvog uspešnog izvođenja, pri čemu je verovatnoća uspeha u jednom pokušaju p. Dakle imamo:

$$P\{X = k\} = p \cdot (1-p)^{k-1} \qquad k = 1,2,\dots$$

Definišemo $X = k$, tako da važi:

$$(\forall k \in \mathbb{N}) \quad p\sum_{i=1}^{k-1}(1-p)^{i-1} \leq U < p\sum_{i=1}^{k}(1-p)^{i-1}$$

$$\downarrow$$

Gde u ima uniformnu raspodelu na $(0,1)$

$$\Leftrightarrow \quad 1-(1-p)^{k-1} \leq U < 1-(1-p)^{k}$$
$$\Leftrightarrow \quad (1-p)^{k} < 1-U \leq (1-p)^{k-1} \quad / ln$$
$$\Leftrightarrow \quad k\,ln(1-p) < ln(1-U) \leq (k-1)ln(1-p)$$
$$\Leftrightarrow \quad k-1 \leq \frac{ln(1-U)}{ln(1-p)} < k$$

Dakle,

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

Odakle, koristeći **Teoremu 4.1.** imamo da je slučajna promenljiva $X \overset{\text{def}}{=} 1 + \left[\frac{\ln U}{\ln(1-p)}\right]$ promenljiva koja ima $\mathcal{G}(p)$ raspodelu.

Primer 4.4. *(Generisanje Poasonove raspodele)* Neka je $\{N_t\}_{t\in(0,+\infty)}$ Poasonov slučajni proces definisan u drugom poglavlju čiji je parametar λ. Tada vreme između dve realizacije ovog procesa (tj. grafički gledano dva skoka na grafiku) ima eksponencijalnu $\mathcal{E}(\lambda)$ raspodelu.

Ako imamo na raspolaganju niz slučajnih brojeva $\{U_n\}_{n=1}^{+\infty}$ uniformno raspodeljenih na $(0,1)$ možemo sprovesti sledeći algoritam:

Neka je $N \in \mathbb{N}$ prvi prirodan broj za koji važi:

$$U_1 \cdot U_2 \cdot \ldots \cdot U_N < e^{-\lambda}$$

Tada slučajna promenljiva $X = N - 1$ ima $\mathcal{P}(\lambda)$ raspodelu.

Dakle, X je najveći prirodan broj n (ili 0 ako ne postoji) takav da važi $U_1 \cdot U_2 \cdot \ldots \cdot U_n \geq e^{-\lambda}$, tj. logaritmovanjem ovoga dobijamo da je to najveći n za koji važi:

$$\sum_{i=1}^{n} ln\, U_i \geq -\lambda\ ln\, e \Leftrightarrow \sum_{i=1}^{n} (-\ln U_i) \leq \lambda$$

Odavde vidimo da je $-ln\, U_i$ vreme između dva Poasonova procesa sa parametrom 1, a njihov zbir je vreme do n-tog događaja.

4.2 Generisanje neprekidnih raspodela

Teorema 4.2. *Neka je u uniformna raspodela na $[0,1]$ i F rastuća, neprekidna funkcija raspodele. Tada slučajna promenljiva $X \stackrel{\text{def}}{=} F^{-1}(U)$ ima raspodelu sa funkcijom raspodele F.*

Dokaz:

$$P\{X \leq x\} = P\{F^{-1}(U) \leq x\} = P\{U \leq F(x)\} = F(x)$$

Dakle, $F^{-1}(U)$ ima funkciju raspodele F.

■

Ovo tvrđenje nam pokazuje da možemo generisati one raspodele kojim možemo naći inverz funkcije raspodele.

Primer 4.5. *(Generisanje eksponencijalne raspodele)* Eksponencijalna raspodela $\mathcal{E}(\lambda)$ definisana je svojom gustinom raspodele:

$$g(x) = \begin{cases} \lambda\, e^{\lambda x}, & x \geq 0 \\ 0, & inače \end{cases}$$

tj. funkcijom raspodele:

$$F(x) = \begin{cases} 1 - e^{-\lambda x}, & x \geq 0 \\ 0, & inače \end{cases}$$

$$y = 1 - e^{-\lambda x} \quad \Longrightarrow e^{-\lambda x} = 1 - y \quad \Longrightarrow -\lambda x = ln(1-y) \quad \Longrightarrow x = -\frac{ln(1-y)}{\lambda}$$

Dakle, $F^{-1}(U) = -\frac{1}{\lambda}\, ln(1-U)$. Kako u ima uniformnu raspodelu, to po **Teoremi 4.1.** i $1-U$ ima $\mathcal{U}[0,1]$ raspodelu, te će slučajna promenljiva $X \overset{\text{def}}{=} -\frac{1}{\lambda}\, ln(U)$ imati eksponencijalnu raspodelu $\mathcal{E}(\lambda)$.

U praksi je generisanje raspodele preko inverzne funkcije često teže izvodljivo jer je nalaženje inverzne funkcije neretko komplikovano.

Teorema 4.3. *Neka X: $\mathcal{U}(0,1)$ i neka je Y slučajna promenljiva koncentrisana na nekom skupu D, gde ima gustinu g, pri čemu su X i Y nezavisne. Neka je f gustina neke slučajne promenljive koncentrisane na skupu D. Ako postoji $c > 0$ tako da $f(y) \leq c \cdot g(y)$ za $y \in D$, tada je:*

$$P\left\{ Y \leq x \,\middle|\, X \leq \frac{f(y)}{c \cdot g(y)} \right\} = \int_{-\infty}^{x} f(y)\, dy$$

Dokaz:

$$P\left\{ X \leq \frac{f(y)}{c \cdot g(y)} \right\} = \int_D \int_0^{\frac{f(y)}{c \cdot g(y)}} g(y)\, dU\, dy = \int_D \frac{f(y)}{c \cdot g(y)}\, g(y) dy = \frac{1}{c} \int_D f(y)\, dy = \frac{1}{c}$$

⇓

$$P\left\{Y \le x \,\middle|\, X \le \frac{f(y)}{c \cdot g(y)}\right\} = \frac{P\left\{Y \le x,\ X \le \frac{f(y)}{c \cdot g(y)}\right\}}{\underbrace{P\left\{X \le \frac{f(y)}{c \cdot g(y)}\right\}}_{\frac{1}{c}}} = c \cdot P\left\{Y \le x,\ X \le \frac{f(y)}{c \cdot g(y)}\right\} =$$

$$= \int_{Dx} \int_{0}^{\frac{f(y)}{c \cdot g(y)}} g(y)\, dU\, dy = \int_{-\infty}^{x} f(y)\, dy$$

■

Na osnovu ove teoreme zasniva se *metod odbacivanja*:

Neka je Y slučajna promenljiva sa gustinom g i neka je f gustina koju treba generisati. Ako su f i g koncentrisane na istom skupu D i $f(y) \le c \cdot g(y)$, $y \in D$ za neko $c > 0$ tada:

a) Generišemo Y sa gustinom g i slučajan broj U;

b) Ako $U \le \frac{f(Y)}{c \cdot g(Y)}$, $X = Y$. Inače se ponavlja korak a).

Ponavljanjem ovog postupka dobija se slučajna promenljiva X i gustina g. I zaista:

$$P\{X \le x\} = P\left\{Y \le x \,\middle|\, U \le \frac{f(Y)}{c \cdot g(Y)}\right\} = \int_{-\infty}^{x} f(Y)\, dy$$

↓

Teorema 4.3.

Tj.

$$F_X(x) = \int_{-\infty}^{x} f(Y)\, dy$$

$\Longrightarrow f(x)$ je gustina slučajne promenljive X

Primer 4.6. *(Generisanje β-raspodele)* Znamo da $\mathcal{B}(\alpha, \beta)$ data gustinom raspodele

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

Neka je

$$g(X) = \begin{cases} 1, & x \in (0,1) \\ 0, & inače \end{cases}$$

tj. gustina $\mathcal{U}(0,1)$ raspodele

Kako važi $f(x) \leq c \cdot g(x)$ to nam metod odbacivanja omogućava generisanje slučajne promenljive X sa beta-raspodelom:

a) Generišemo nezavisne U_1, U_2: $\mathcal{U}(0,1)$;
b) Ako je $U_2 < {U_1}^{\alpha-1}(1-U_1)^{\beta-1}$ $X = U_1$, inače se ponavlja korak a).

Primer 4.7. *(Generisanje normalne raspodele)* Neka su $U_1, \dots, U_{12}$ nezavisne i uniformno raspodeljene na $(0,1)$.

$$Z \stackrel{\text{def}}{=} U_1 + \cdots + U_{12} - 6$$

$$E(Z) = E(U_1 + \cdots + U_{12} - 6) = 12 \underbrace{E(U)}_{\frac{1}{2}} - 6 = 0$$

$$Var(Z) = Var(U_1 + \cdots + U_{12}) = 12 \underbrace{Var(U)}_{1/12} = \frac{12}{12} = 1$$

$$\Rightarrow \quad Z: \mathcal{N}(0,1)$$

Naravno, odatle je lako i generisati proizvoljnu $\mathcal{N}(\mu, \sigma^2)$ raspodelu:

$$Z: \mathcal{N}(0,1) \quad \Rightarrow X \stackrel{\text{def}}{=} \sigma Z + \mu : \begin{matrix} E(X) = \mu \\ Var(X) = \sigma^2 \end{matrix} \quad \Rightarrow X: \mathcal{N}(\mu, \sigma^2)$$

5. Primena Markovljevih lanaca Monte Karlo (MCMC) za generisanje uzoraka iz raspodela (Metropolis - Hejstings algoritam)

5.1 Uvod i glavna ideja algoritma

Metropolis – Hejstings algoritam je prvobitno uveo Nikolas Metropolis (1915-1999.) u svom radu iz 1953. dok je algoritam uopštio V. K. Hejstings (1930-2016.) 1970-e godine. Sam algoritam je od velikog značaja u statistici i statističkoj fizici.

Suština algoritma je u generisanju slučajnih uzoraka iz neke nama nepoznate raspodele, putem konstruisanja Markovljevog lanca čije elemente dobijamo koristeći raspodelu koja nije ista, ali je slična željenoj raspodeli.

Metode izložene u četvrtom poglavlju su korisnije pri generisanju uzoraka iz nepoznate jednodimenzione raspodele, mada Metropolis – Hejstings algoritam pokazuje znatno veću korisnost i efikasnost od višedimenzionalnih raspodela.

Pretpostavka algoritma je da nam je poznata neka funkcija $f(x)$ koja je proporcionalna gustini $\pi(x)$ ciljne raspodele. Algoritam tu pokazuje korisnost, jer je sam uslov proporcionalnosti mnogo lakše ispuniti u praksi negoli eksplicitno naći odgovarajuću konstantu uvedenu u **Teoremi 4.3.** iz četvrtog poglavlja.

Sam algoritam generiše vrednosti iterativno, pri čemu raspodela svakog sledećeg uzorka zavisi isključivo od raspodele prethodnog, što je glavno svojstvo da ceo postupak predstavlja jedan Markovljev lanac. Preciznije, u svakom sledećem koraku, algoritam predlaže sledećeg kandidata na osnovu prethodne vrednosti uzorka. Sa nekom verovatnoćom se kandidat prihvata ili odbacuje (u kom slučaju se koristi prethodna vrednost i u sledećoj iteraciji). Sama verovatnoća prihvatanja direktno zavisi od vrednosti funkcije $f(x)$ i vrednost verovatnoće prelazi tog Markovljevog lanca. Ono što je bitno primetiti da ovde uzorci nisu nezavisni.

Ilustrujmo ideju algoritma na originalnom koji je dao Metropolis (*algoritam sa simetričnom predloženom raspodelom*):

A. Biramo polaznu tačku x_0 i gustinu predložene raspodele $g(x \mid y)$. Uslov Metropolisove varijante algoritma je uslov simetričnosti – funkcija $g(x \mid y)$ mora biti simetrična, tj. $g(x \mid y) = g(y \mid x)$. Odnosno, verovatnoća dolaska iz tačke x u tačku y mora biti jednaka verovatnoći dolaska iz tačke y u tačku x. Iz ovoga je jasno da se u praksi ovog algoritma koriste normalne raspodele. Preciznije, za $g(x \mid y)$ uzimamo normalnu raspodelu centriranu u y sa manjom varijansom, te Metropolis algoritam simulira nasumičnu šetnju uvedenu u drugom poglavlju.

B. U koraku t:

- Generišemo kandidata x' preko raspodele $g(x' \mid x_t)$;
- Računamo $\alpha \stackrel{\text{def}}{=} \frac{f(x')}{f(x_t)}$ koji koristimo u zaključivanju hoćemo li prihvatiti x' ili ne.

Pri odluci da li se prihvata x' ili ne koristimo ideju sličnu mnogim do sada prikazanim Monte Karlom algoritmima:

1. Generišemo slučajan broj Y: $\mathcal{U}[0,1]$;
2. Ako je $Y \leq \alpha$, prihvatamo kandidata x', $x_{t+1} \stackrel{\text{def}}{=} x'$ i ponavljamo korake A. i B. ;
3. Ako je $Y > \alpha$ odbacujemo kandidata x', $x_{t+1} \stackrel{\text{def}}{=} x_t$ i ponavljamo korake A. i B. .

Jedna od mana algoritma je korelisanost uzoraka. Mada će uzorci sa ogromnim brojem ponavljanja eksperimenata upasti u ciljanu raspodelu π(x), uzastopni uzorci neće odražavati pravu raspodelu. Praktično se ovo poboljšava odbacivanjem velikog broja uzoraka i uzimanja svakog n-tog, za neko n određeno posmatranjem autokorelacije između uzoraka. Takođe, ponekad se odbacuju inicijalni uzorci (prvih 1000 ili slično); taj proces se naziva proces sagorevanja.

Međutim, i pored svih mana Metropolis – Hejstings algoritam pokazuje značajnu prednost u odnosu na ostale algoritme za generisanje raspodela kako se broj dimenzija povećava. Složenost Metropolis – Hejstings algoritma ostaje fiksna sa rastom dimenzija (jako slično Monte Karlo integraciji), dok kod većine algoritama za generisanje raspodela uvedenih u četvrtom poglavlju, složenost raste eksponencijalno s povećanjem dimenzije.

5.2 Formalna definicija algoritma i specijalni slučajevi

Samu definiciju Metropolis – Hejstings algoritma dao je Hejstings u svom radu 1970-e godine uopštavajući Metropolisov algoritam uveden u 5.1.

Pretpostavka ovog algoritma je da za predloženu funkciju $g(x \mid y)$ mora važiti uslov:

$$g(x \mid y) > 0 \quad akko \quad g(y \mid x) > 0$$

Neka je π(x) ciljna raspodela (preciznije skalirana ciljna raspodela). Algoritam glasi:

A. Biramo x_0 ;

B. U svakom sledećem koraku, $j+1$ ($j = \{0,1,2, \dots\}$), biramo kandidata x_* putem raspodele $g(x \mid x_j)$. Računamo koeficijent prihvatanja:

$$\alpha \stackrel{\text{def}}{=} min\left\{1, \ \frac{\pi(x_*)}{\pi(x_j)} \cdot \frac{g(x_j \mid x_*)}{g(x_* \mid x_j)}\right\} \qquad (1)$$

Generišemo slučajan broj U: $\mathcal{U}[0,1]$. Sada imamo dva slučaja:

1. Kada je $\alpha \geq U$, imamo $x_{j+1} \stackrel{\text{def}}{=} x_*$, tj. prihvatamo kandidata x_*;
2. Kada je $\alpha < U$, imamo $x_{j+1} \stackrel{\text{def}}{=} x_j$, tj. zadržavamo staru vrednost i odbacujemo kandidata x_*.

Modifikacije samog algoritma se zasnivaju na izmenama koeficijenta prihvatanja α.

Barker (1965.) daje ovakav predlog:

$$\alpha \stackrel{\text{def}}{=} \frac{\pi(x_*)\, g(x_* \mid x_j)}{\pi(x_*)\, g(x_* \mid x_j) + \pi(x_*)\, g(x_j \mid x_*)} \qquad (2)$$

Dok je Čarls Stejn (1920-2016.) daje:

$$\alpha \stackrel{\text{def}}{=} \frac{\delta(x_j \mid x_*)}{\pi(x_j)\, g(x_j \mid x_*)} \qquad (3)$$

pri čemu je δ bilo koja simetrična funkcija za koju važi $\alpha \leq 1$.

U ovom slučaju verovatnoće prelazasu:

$$p(x_i, x_j) = g(x_i \mid x_j)\, \alpha = g(x_i \mid x_j)\, \frac{\delta(x_i \mid x_j)}{\pi(x_i)\, g(x_i \mid x_j)} = \frac{\delta(x_i \mid x_j)}{\pi(x_i)} \qquad (4)$$

Zbog simetričnosti funkcije δ dobijamo:

$$\pi(x_i) \cdot p(x_i, x_j) = \delta(x_i \mid x_j) = \delta(x_j \mid x_i) = \pi(x_j) \cdot p(x_j, x_i)$$

tj. važi:

$$\pi(x_i) \cdot p(x_i, x_j) = \pi(x_j) \cdot p(x_j, x_i) \qquad (5)$$

Ova jednačina zapravo pokazuje da Stejnov algoritam ima svojstvo detaljne uravnoteženosti, tj. da je Markovljev lanac koji se tada konstruiše reverzibilan.

Teorema 5.1. *Metropolis – Hejstings algoritam zadovoljava svojstvo detaljne uravnoteženosti.*

Dokaz:

$$p(x_i, x_j) = g(x_i \mid x_j)\, \alpha = g(x_i \mid x_j) \min\left\{1,\ \frac{\pi(x_i)}{\pi(x_j)} \cdot \frac{g(x_j \mid x_i)}{g(x_i \mid x_j)}\right\}$$

$$\Downarrow$$

$$\pi(x_i)\, p(x_i, x_j) = \pi(x_i)\, g(x_i \mid x_j) \min\left\{1,\ \frac{\pi(x_i)}{\pi(x_j)} \cdot \frac{g(x_j \mid x_i)}{g(x_i \mid x_j)}\right\} =$$

$$= \min\{\pi(x_i)\, g(x_i \mid x_j),\ \pi(x_j)\, g(x_j \mid x_i)\} =$$

$$= \pi(x_j)\, p(x_j \mid x_i)$$

■

Ukoliko je predložena funkcija g simetrična, tj. $g(x_i \mid x_j) = g(x_j \mid x_i)$, to (1) postaje:

$$\alpha = min\left\{1, \ \frac{\pi(x_*)}{\pi(x_j)}\right\} \qquad (6)$$

tj. Metropolis algoritam uveden u 5.1. se može izvesti iz Metropolis – Hejstings algoritma.

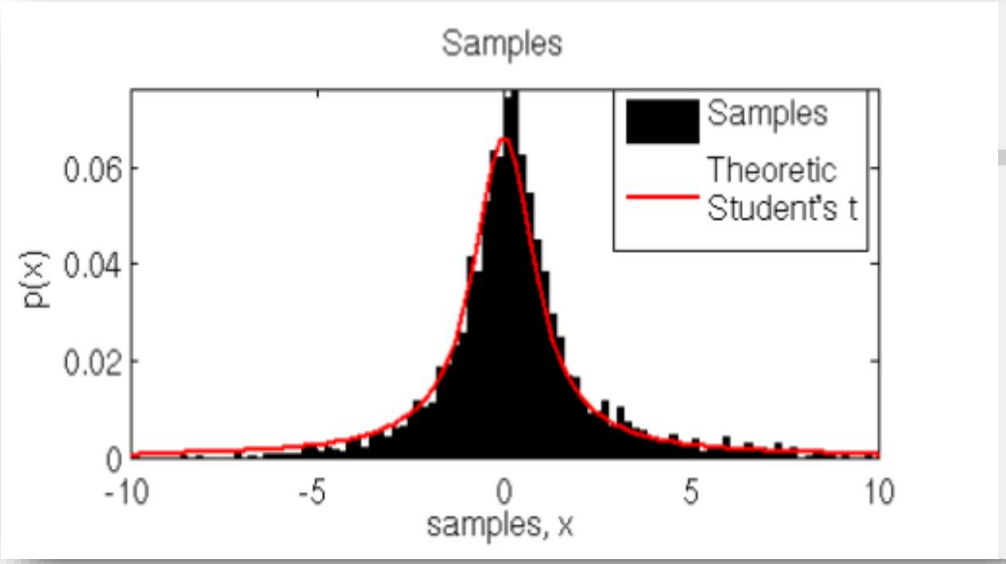
Ukoliko predložena funkcija ne zavisi od prethodne vrednosti, tj. ako je $g(x_* \mid x_j) = g(x_*)$, tada (1) postaje:

$$\alpha = min\left\{1, \ \frac{\pi(x_*)}{\pi(x_j)} \cdot \frac{g(x_j)}{g(x_*)}\right\} \qquad (7)$$

5.3 Primeri

A. Primenimo Metropolis – Hejstings algoritam za generisanje jedne poznate raspodele – Studentove raspodele sa jednim stepenom slobode t_1, čija je gustina $\mathrm{p}(x) = \frac{1}{1+x^2} \cdot \frac{1}{\pi}$.

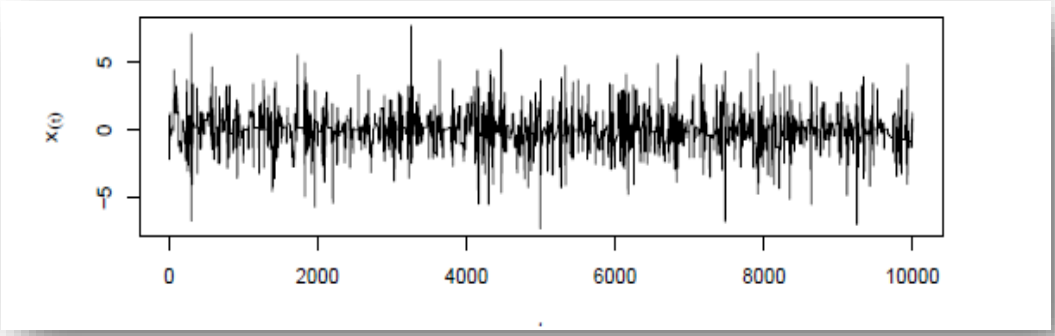
Neka X_0: $\mathcal{N}(0,1)$ i neka je probna raspodela $\mathcal{N}(x,1)$. Nakon 5000 iteracija dobija se grafikon koji vidimo na *Slika 5.1*:



Slika 5.1 - Slika preuzeta iz [17]

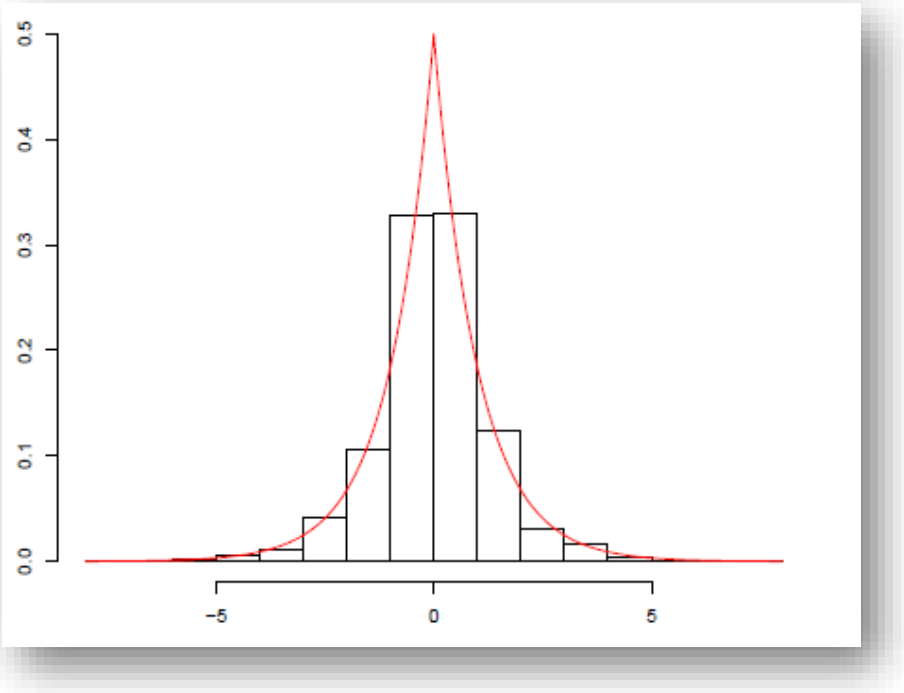
B. *(Metropolis slučajna šetnja)* Želimo da pomoću Metropolis nasumične šetnje generišemo Laplasovu distribuciju, čija je gustina raspodele $f(x) = \frac{1}{2}e^{-|x|},\ x \in \mathbb{R}$.

Koristićemo $x^* = x_t + \varepsilon$, gde ε: $\mathcal{N}(0,100)$. Dobijamo sledeći uzorak:



Slika 5.2 - Slika preuzeta iz [18]

Dok predstavljanje na histogramu vidimo na *Slika 5.3*:



Slika 5.3 - Slika preuzeta iz [18]

Gde je crvenom bojom obeležen grafik funkcije $f(x) = \frac{1}{2}e^{-x}, \quad x \in \mathbb{R}$. Pri pravljenju histograma odbačeno je prvih 200 uzoraka u procesu sagorevanja.

C. *(Generisanje standardne normalne distribucije koristeći Metropolis – Hejstings algoritam)* Poći ćemo od $X_0 = 0$, dok će predložena raspodela biti $\mathcal{N}(x_n, \sigma^2)$. Pomoću R – koda sa *Kod 5.1* možemo generisati različite uzorke:

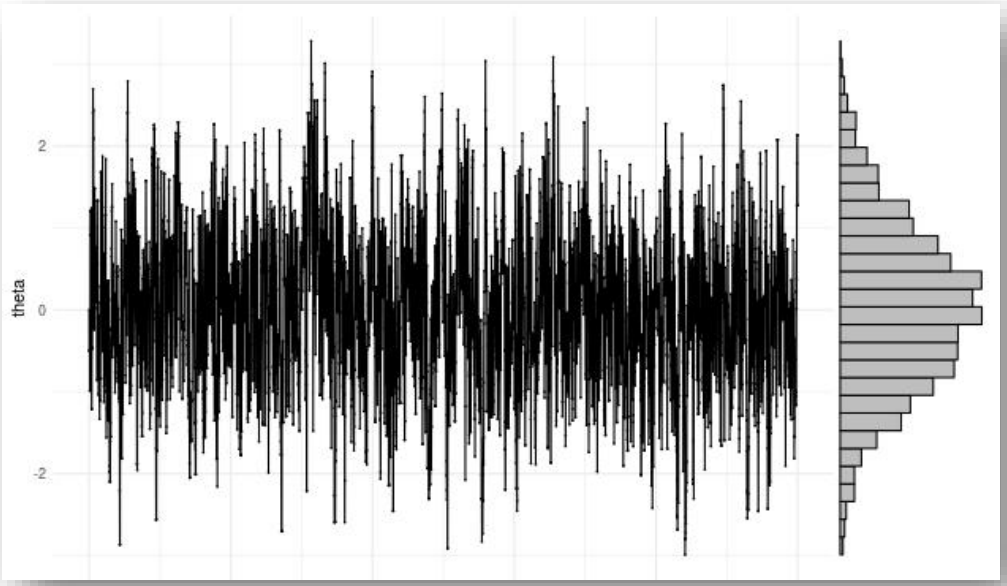
```
mh_sampler <- function(dens, start = 0, nreps = 1000, prop_sd = 1, ...){
  theta <- numeric(nreps)
  theta[1] <- start

  for (i in 2:nreps){
    theta_star <- rnorm(1, mean = theta[i - 1], sd = prop_sd)
    alpha = dens(theta_star, ...) / dens(theta[i - 1], ...)

    if (runif(1) < alpha) theta[i] <- theta_star
    else theta[i] <- theta[i - 1]
  }

  return(theta)
}
```

Kod 5.1 - Kod preuzet iz [20]

Uvek ćemo koristiti 5000 iteracija.

Dobijamo sledeće uzorke, tj. histograme uz prikazan procenat prihvaćenih uzoraka.

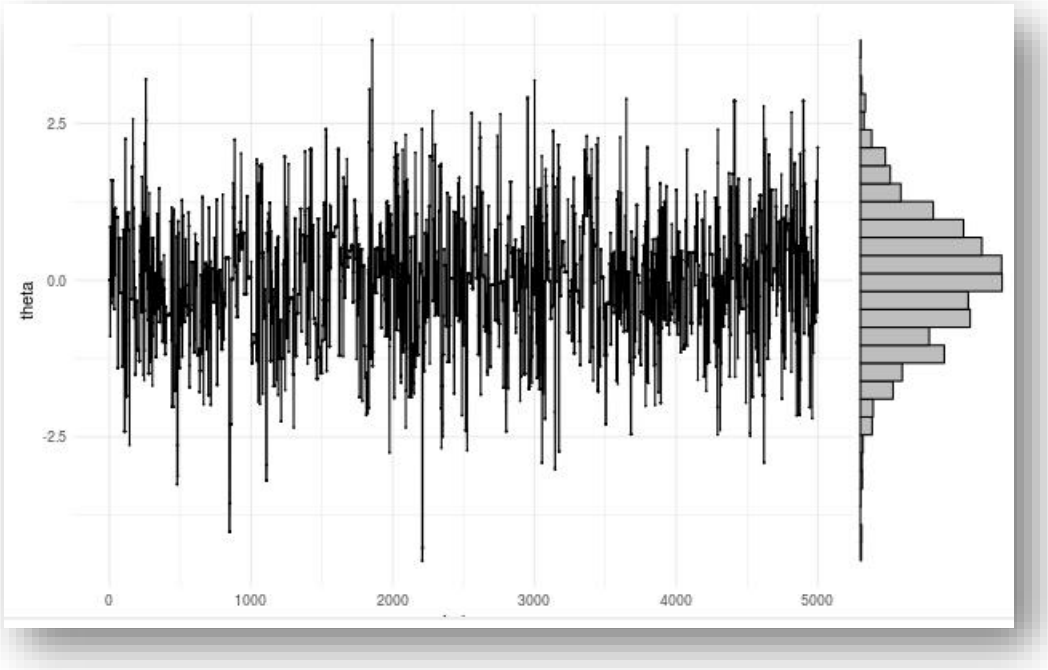
$\sigma = 1$:



Slika 5.4 - Slika preuzeta iz [20]

(71% prihvaćenih)

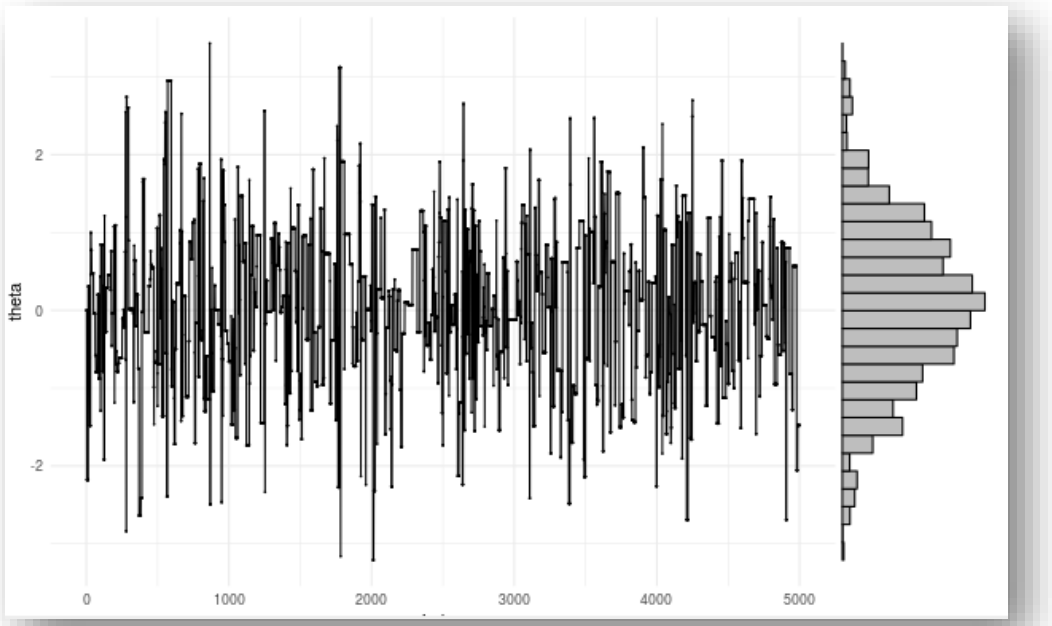
$\sigma = 5$:



Slika 5.5- Slika preuzeta iz [20]

(24% prihvaćenih)

$\sigma = 10$:



Slika 5.6 - Slika preuzeta iz [20]

(13% prihvaćenih)

6. Monte Karlo pretrag stabla (MCTS)

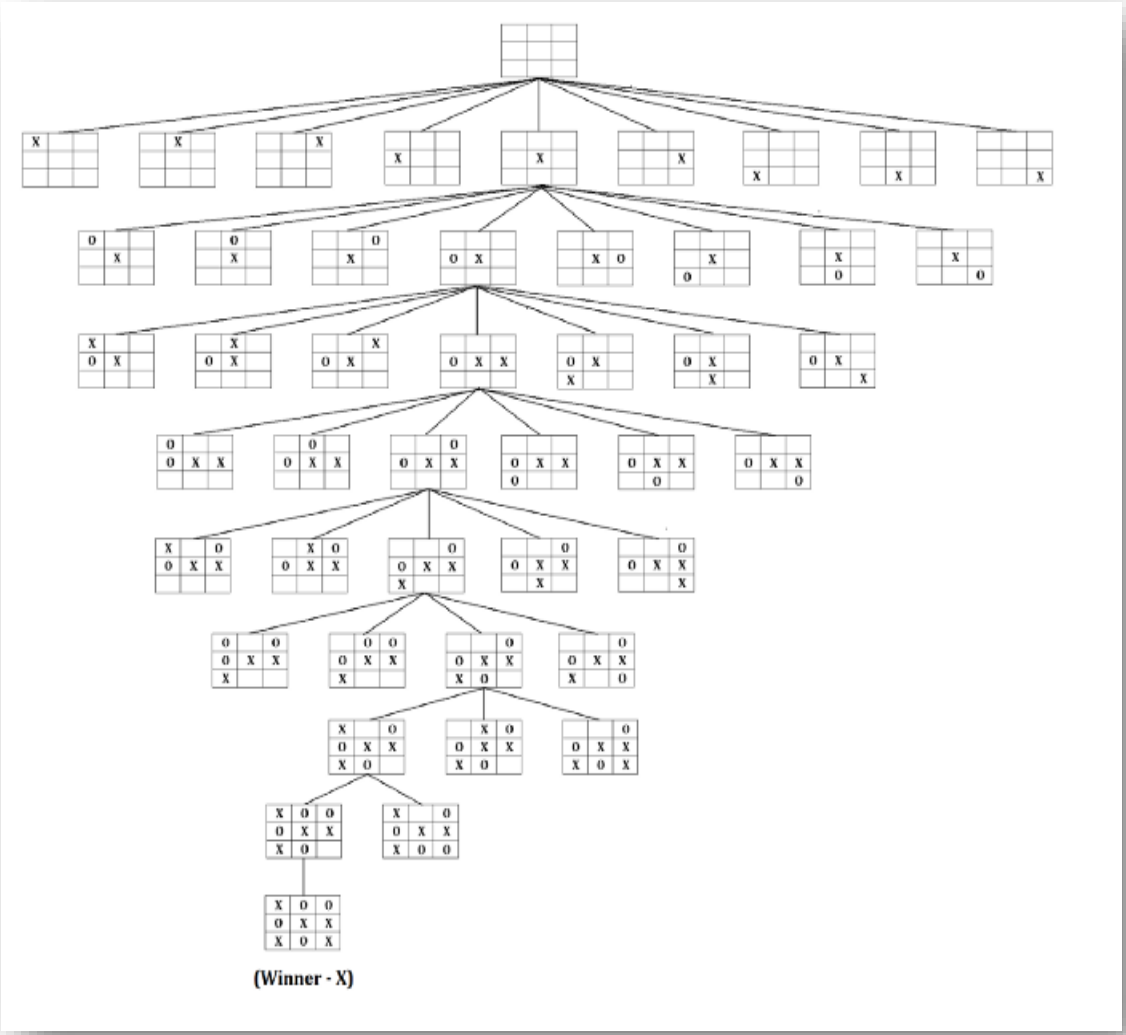
6.1 Uvod i istorijat

Monte Karlo pretraga stabla (MCTS) je heuristički algoritam pretrage koji se koristi u teoriji igara i u kompjuterskim naukama.

Sam algoritam je vrlo primenljiv u mnogim poteznim igrama kao što su šah i go, mada postiže odlične rezultate i u igrama zasnovanim na probabilizmu, kao što su poker i bridz. Primenljivost algoritma na kompjuterske nauke dovela je do toga da je ovaj algoritam korišćen pri modeliranju poteza veštačke inteligencije u kompjuterskim igrama “Rome: Total War II” i “Hearthstone”.

Algoritam služi u odabiru najboljeg poteza u nekoj igri, te rekurzivnom primenom stvaranja pobedničke, odnosno najbolje strategije. U informatičkoj terminologiji, MCTS algoritam služi za rešavanje stabla igre. Pod stablom igre se podrazumeva usmereni graf čiji koren predstavlja početno stanje u igri, dok svaki nivo grafa, tj. stabla predstavlja sva moguća stanja u igri nakon drugog, trećeg, četvrtog,… poteza.

Ako posmatramo stablo popularne dečije igre iks – oks:



Slika 6.1 - Slika preuzeta iz [15]

Pri čemu je ovo samo deo tog stabla zbog jednostavnosti prikaza.

Na *Slika 6.1* možemo videti da se i kod najjednostavnijih igara broj elemenata stabla igre eksponencijalno povećava sa svakim novim potezom. Broj grananja po stadijumu igre naziva se faktor grananja. Kod igara koje su složenije od iks – oksa, kao što je go, faktor grananja je u proseku 250. Ako npr. radimo analizu poteza u igri go, evo koliki broj partija treba prosečno analizirati da bi se odredio najoptimalniji potez:

A. Posle prvog koraka: 250 partija;

B. Posle drugog koraka: $250^2 = 62500$ partija;
C. Posle trećeg koraka: $250^3 = 15625000$ partija;

.

.

.

D. Posle desetog koraka: 250^{10} partija.

Ove brojke govore da je pri konstruisanju kvalitetnih algoritama za traženje pobedničkih poteza korišćenje bilo kakvih algoritama sirove sile (eng. brute force) veoma nepraktično i iziskuje obilno korišćenje resursa računara.

Zato se pristupilo heurističnijim algoritmima, gde se ne razmatraju svi čvorovi stabla igre, već se nekima daje prednost u odnosu na ostale.

Naravno, postojali su i neki oblici veštačke inteligencije koji su koristili metode sirove sile u donošenju odluke i imaju poprilično uspeha. Najčuveniji je svakako program "deep blue" koji je 1987. uspeo da pobedi šahovskog velemajstora Garija Kasparova.

Inspirisan programima poput "deep blue", francuski kompjuterski naučnik Remi Kulom (1974.-) je opisao algoritam za implementaciju Monte Karlo metoda u pretrazi stabla igre i nadenuo mu ime Monte Karlo pretraga stabla (MCTS – Monte Carlo Tree Search).

Mađarski naučnici Kočiš i Šepešvari su daljim usavršavanjem tog algoritma konstruisali program "MoGo", čije su kasnije verzije počele da pobeđuju profesionalne igrače u 9×9 verziji igre go. 2012 godine program "Zen" je pobedio igrača sa titulom drugi dan sa 3:1 u 19×19 verzijom igre go. Kao vrhunac implementacije MCTS algoritma, "Google Deepmind" je razvio program "AlphaGo", koji je 2015. postao prvi kompjuterski program koji je pobedio majstora igre go sa 4:1, nakon čega je program dobio počasnu titulu majstora 9.dan.

6.2 Formalna definicija algoritma

MCTS algoritam funkcioniše tako što korišćenjem nasumičnih uzoraka iteraktivno konstruiše statističko stablo igre, što je zapravo jedno stablo igre gde čvorovi stabla imaju statističku težinu, tj.svakom

čvoru u stablu dodeljna je neka verovatnoća.

Tokom MCTS algoritma vrši se veliki broj odigravanja partija (roll-outs) na osnovu kojih se vrednosti verovatnoća u čvorovima ažuriraju. Najosnovniji metod je u odigravanju istog broja partija za svaki čvor na jednom nivou stabla igre, i onda rekurzivno nastaviti postupak na onom najuspešnijem čvoru. Ovakva modifikacija se zove čista Monte Karlo pretraga. Kada broj upita po nivou stabla teži $+\infty$, metoda pokazuje najverodostojnije i najpreciznije rezultate. Sve varijante MCTS algoritma imaju 4 koraka, a to su:

A. *Odabir*
B. *Proširenje*
C. *Simulacija*
D. *Vraćanje rezultata*

Opišimo ove korake:

A. *Odabir* – Ovaj korak počinje iz korena stabla igre (početak igre ili tekuće stanje igre). Spuštamo se u dubinu stabla, konstantno birajući dozvoljen potez sve dok ne dođemo do lista stabla (čvor koji nema naslednika, tj. kraj igre). Cilj ovog dela algoritma je istraživanje novih varijanti igre da bi se dobile nove informacije i korišćenje poznatih puteva kroz stablo igre za koje znamo da su dobri potezi.

U praksi, poslove istraživanja i eksploatacije čvorova obavlja se konstruisanjem funkcije odabira.

Pri samom odabiru ovih puteva kroz stablo, mogu se putanje birati nasumično ili korišćenjem najbolje po statističkom proseku. Ovakve varijante daju dobru eksploataciju stabla, ali loše utiču na istraživanje, tj. daju slab učinak za proširivanje stabla i informacija o njemu. Zato se u praksi koristi UCT algoritam (Upper Confidence bounds applied to Trees). Njegova funkcija odabira je:

$$\frac{\omega_i}{s_i} + C \cdot \sqrt{\frac{\ln s_p}{s_i}} \qquad (6.1.)$$

Gde su: ω_i – broj simulacija sa početkom u tekućem čvoru koje su završene pobedom;

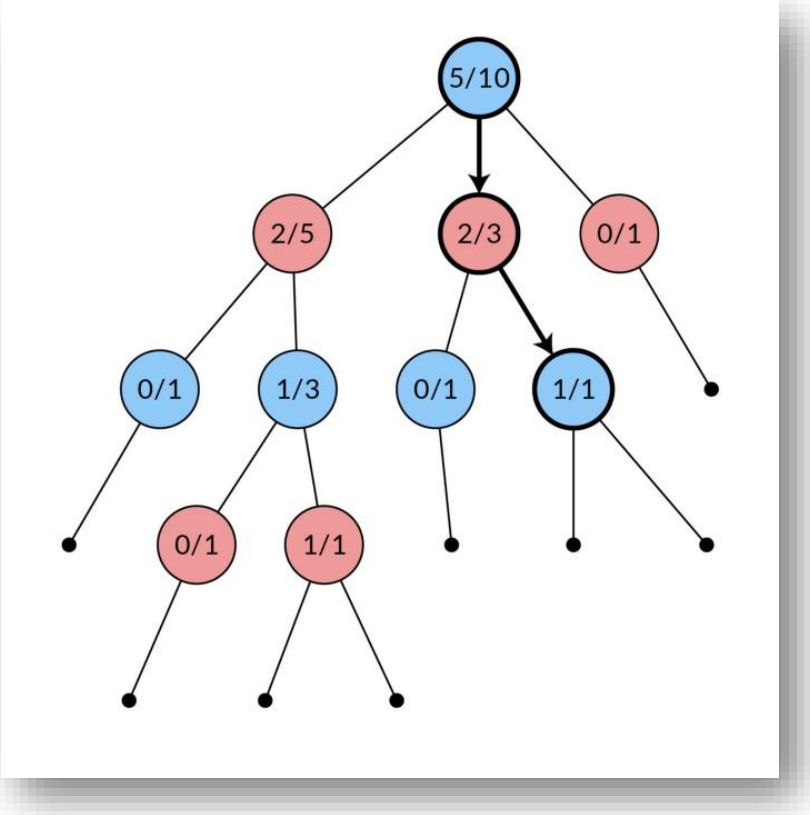
s_i – ukupan broj simulacija iz roditeljskog čvora;

s_p – ukupan broj simulacij iz roditeljskog čvora;

C – parametar (najčešće $\sqrt{2}$, mada se može i birati srazmerno uzorku)

Izraz $\frac{\omega_i}{s_i}$ se naziva član eksploatacije, i on predstavlja prosti statistički prosek pobeda iz nekog čvora. Izraz $\sqrt{\frac{\ln s_p}{s_i}}$ se zove član istraživanja. Parametar C služi za ostvarenje balansa između člana eksploatacije i istraživanja.

Ako posmatramo *Slika 6.2*:



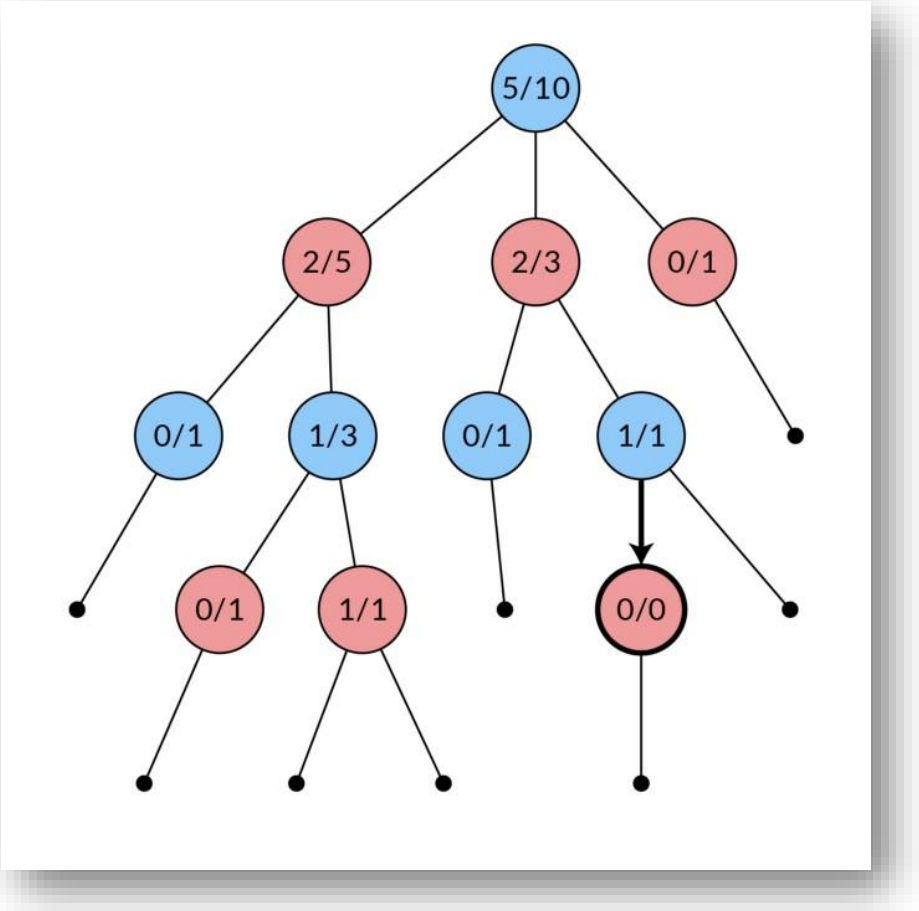
Slika 6.2 - Slika preuzeta iz [16]

Brojevi u čvorovima predstavljaju statistiku čvora, tj. $\frac{\omega_i}{s_i}$. Svaki put kada treba odabrati sledeći potez, tj. sledeći čvor, koristi se formula (6.1.) koja koristi statistiku čvora i statistiku

njegovog roditelja.

Ako posmatramo sliku i odabir prvog poteza, UCT algoritam kaže da je najbolje odabrati drugi čvor, tj. onaj sa verovatnoćom 2/3.

B. *Proširenje* – Nakon završenja procesa odabira, ostaće bar jedan neposećen čvor u stablu, koji predstavlja deo stabla koji se može još raširiti, tj. neočekivane poteze. U ovom koraku nasumično biramo sledeći potez i adekvatno tome dodajemo novi čvor u stablo. Ako gledamo *Slika 6.3*:

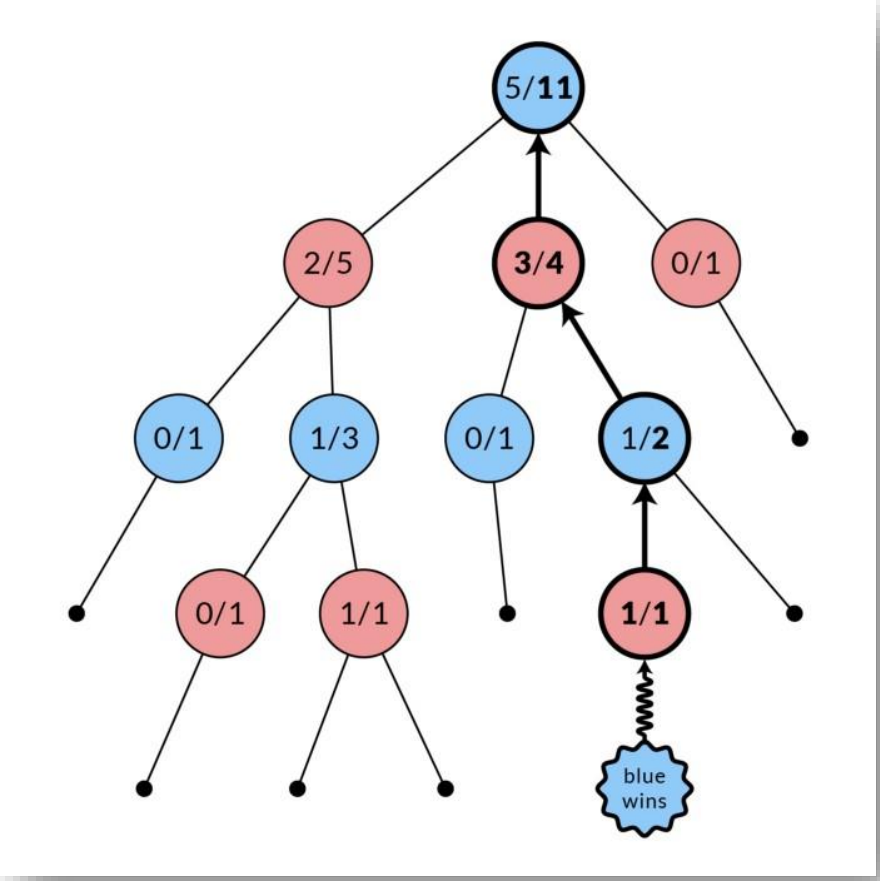


Slika 6.3- Slika preuzeta iz [16]

To je podebljani 0/0 čvor. Taj čvor se dodaje poslednjem čvoru izabranom u procesu odabira i statistika mu se postavlja na 0/0, i to će se statistički menjati u trećoj fazi.

C. *Simulacija* – Počevši od dodatog 0/0 čvora, potezi se povlače nasumično i vrše se razne simulacije igre dok se ne dođe do pobednika. U ovoj fazi algoritma se ne dodaju novi čvorovi, zato što stablo igre odgovara stanju tekuće igre, dok se simulacije odnose na mnoge scenarije koji se neće desiti u igri. Nakon izvršenih simulacija, statistika 0/0 čvora se ažurira.

D. *Vraćanje rezultata* – Nakon faze simulacije, statistika svih posećenih čvorova u igri se ažurira. Posmatramo na *Slika 6.4*:



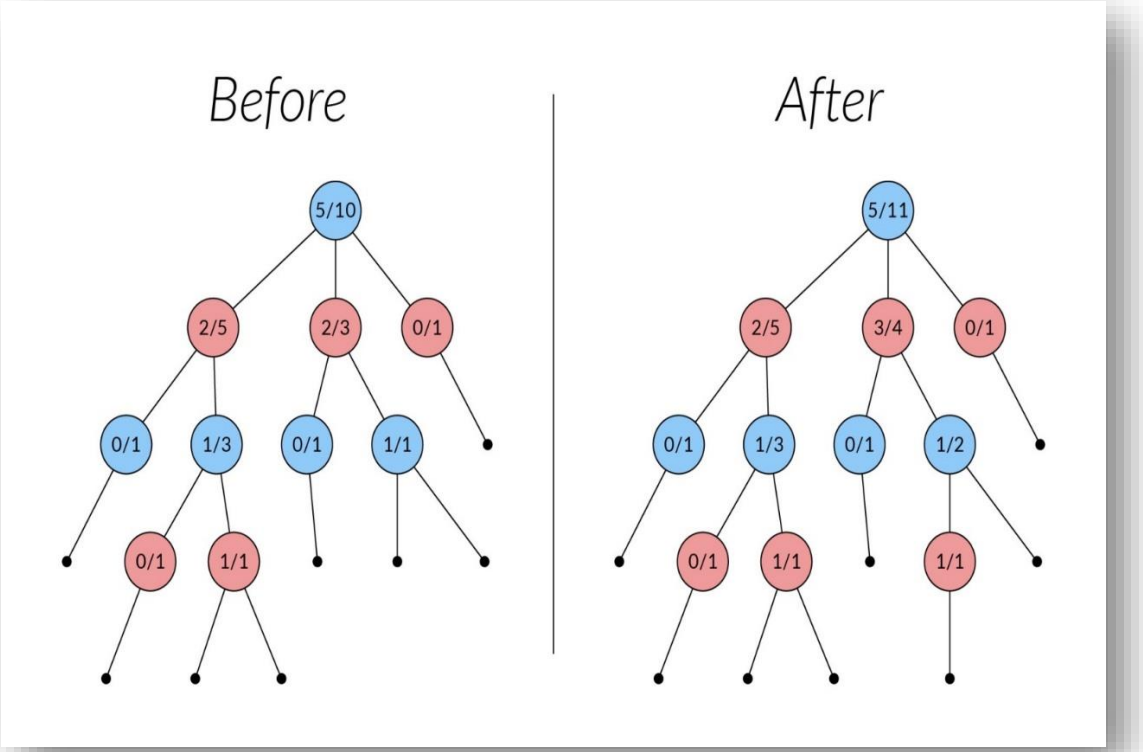
Slika 6.4 - Slika preuzeta iz [16]

S obzirom da je pobedio igrač “blue” , tj. onaj koji je prvi na potezu, te se broj pobeda u svakom drugom redu stabla povećava za 1.

Ako sada posmatramo formulu (6.1.):

MCTS algoritam koristi funkciju odabira da bi izabrao sledeći čvor koji će koristiti, zatim koristi broj pobeda ω_i i simulacija s_i dece tog čvora, kao i broj simulacija čvora iz koga smo pošli s_p. Na kraju koraka u algoritmu , sve te informacije se ažuriraju.

Konkretno, dodavanje jednog čvora na našem primeru (tj. jedne iteracije algoritma) i menjanje statistike u stablu možemo videti na *Slika 6.5*:



Slika 6.5 - Slika preuzeta iz [16]

Zaključak

U ovom radu smo prezentovali neke raznvrsne primene Monte Karlo metoda. Videli smo da spektar primena ovih metoda varira od Bajesove statistike, preko numeričke integracije, pa sve do strateške teorije igara.

Kao ubedljivo najpozitivnija stvar kod ovih metoda jeste njihova jednostavnost ideje i za algoritme su intuitivno jasne i razumljive. Premda smo ih primenjivali u najrazličitijim granama nauke, konceptualno algoritam je uvek bio isti: primeni ogroman broj simulacija, potom donesi neke zaključke o posmatranoj pojavi.

Laka prenosivost na najrazličitije probleme se naročito videla u primeni na probleme numeričke integracije, sam metod se potpuno jednostavno preneo na višedimezionu integraciju bez povećanja složenosti i tu postao znatno dominantniji u odnosu na druge algoritme numeričke integracije.

U radu je prikazan i značaj ovih metoda u statistici i programiranju, pri konstruisanju nekih raspodela. Od jednostavnijih simulacija i programa za generisanje raspodela stiglo se do ozbiljnijih simulacija kao što je Metropolis - Hejstings algoritam.

Naravno, Monte Karlo metode imaju i svojih mana koje se pre svega ogledaju u velokom broju simulacija koje treba izvršiti da bi se došlo do nekih zaključaka i predviđanja o rešenjima problema. Na primer, u programiranju dovodi do veće upotrebe memorije, pa time i veće vremenske i prostorne složenosti algoritma. A takođe se, na primenama u numeričkoj integraciji, videlo da Monte Karlo metode imaju slabije rezultate u oblasti gde već postoje neki deterministički algoritmi kao i u nižim dimenzijama.

Svakako i pored svih tih mana, prednosti Monte Karlo simulacija su brojne i sa sigurnošću se može reći da će njihov značaj biti još veći u budućnosti, pogotovu pri početnom pristupu problemima za čija rešenja ne postoje jasne pretpostavke. Sa razvojem računara pomoću kojih će moći sve veći broj simulacja da se vrši, primenljivost i kvalitet ove metode će se povećavati.

Literatura

[1] Milan Merkle, Verovatnoća i statistika za inženjere i studente tehnike, Akademska misao, 2002.

[2] Casella G., E. I. George, Explaining the Gibbs Sampler, The American Statistician, Vol. 46, No. 3. (Aug., 1992), pp. 167-174

[3] Koralov L.B., Sinai Y.G., Theory of Probability and Random Processes, Springer, 2007

[4] Eckhardt, Roger, Stan Ulam, John von Neumann, and the Monte Carlo method, Los Alamos Science, 1987

[5] D.J.C Mackay, Introduction to Monte Carlo methods, Cambridge University

[6] L. Tierney, Markov chains for exploring posterior distributions, Annals of Statistics, 1994

[7] Tierney, L. (1998). A note on Metropolis-Hastings kernels for general state spaces. Ann. Applied Prob., 8 (1): 1–9.

[8] https://en.wikipedia.org/wiki/Monte_Carlo_method

[9] https://cs.dartmouth.edu/~wjarosz/publications/dissertation/appendixA.pdf

[10] https://www.scratchapixel.com/lessons/mathematics-physics-for-computer-graphics/monte-carlo-methods-in-practice/monte-carlo-integration

[11] https://towardsdatascience.com/from-scratch-bayesian-inference-markov-chain-monte-carlo-and-metropolis-hastings-in-python-ef21a29e25a

[12] https://cepr.org/sites/default/files/40002_Session%202%20-%20MetropolisHastings.pdf

[13] https://bookdown.org/rdpeng/advstatcomp/metropolis-hastings.html

[14] https://towardsdatascience.com/monte-carlo-tree-search-158a917a8baa

[15] https://towardsdatascience.com/monte-carlo-tree-search-an-introduction-503d8c04e168

[16] https://medium.com/@quasimik/monte-carlo-tree-search-applied-to-letterpress-34f41c86e238

[17] http://elibrary.matf.bg.ac.rs/bitstream/handle/123456789/4492/masDjuricMarija.pdf?sequence=1

[18] https://www.hds.utc.fr/~tdenoeux/dokuwiki/_media/en/mcmc_slides.pdf

[19] https://www.nicksolomon.me/post/learn-metropolis-hastings-sampling-with-r/

[20] https://en.wikipedia.org/wiki/Random_walk

[21] https://en.wikipedia.org/wiki/Wiener_process

[22] https://towardsdatascience.com/the-poisson-process-everything-you-need-to-know-322aa0ab9e9a

[23] https://www.mathsisfun.com/calculus/integration-definite.html

Biografija

Smilja Rakić je rođena 04.02.1993. u Loznici, Republika Srbija. Osnovnu školu „Borivoje Ž. Milojević“ u Krupnju završava 2008. godine kao nosilac Vukove diplome. Iste godine upisuje i Gimnaziju „Vuk Karadžić“ u Loznici, prirodno - matematički smer. Posle četiri godine, po završetku gimnazije, upisuje Matematički fakultet u Beogradu na smer Profesor matematike i računarstva (L). Nakon 40 položenih ispita, prelazi na integrisane studije Master profesor matematike (M5) na Prirodno - matematičkom fakultetu u Novom Sadu 2018. godine. Sa položenim svim ispitima predviđenim planom i programom stekla je uslov za odbranu ovog rada.

Novi Sad, 2021. Smilja Rakić

UNIVERZITET U NOVOM SADU

PRIRODNO - MATEMATIČKI FAKULTET

KLJUČNA DOKUMENTACIJSKA INFORMACIJA

Redni broj (RBR):

Identifikacioni broj (IBR):

Tip dokumentacije (TD): Monografska dokumentacija

Tip zapisa (TZ): Tekstualni štampani materijal

Vrsta rada (VR): Master rad

Autor (AU): Smilja Rakić

Mentor (MN): dr Danijela Rajter-Ćirić

Naslov rada (NR): Neke Monte Karlo metode i njihove primene

Jezik publikacije (JP): srpski (latinica)

Jezik izvoda (JI): srpski i engleski

Zemlja publikovanja (ZP): Republika Srbija

Uže geografsko područje (UGP): Vojvodina

Godina (GO): 2021.

Izdavač (IZ): Autorski reprint

Mesto i adresa (MA): Novi Sad, Departman za matematiku i informatiku, Prirodno-matematički fakultet, Univerzitet u Novom Sadu, Trg Dositeja Obradovića 3

Fizički opis rada (FO): (6/70/0/3/35/0/0) (broj poglavlja/broj strana/broj citata/broj tabela/broj slika/broj grafika/broj priloga)

Naučna oblast (NO): Matematika

Naučna disciplina (ND): Verovatnoća, Stohastička analiza

Predmetna odrednica/ključne reči (PO/UDK): Monte Karlo metode, Markovljevi lanci, Monte Karlo integracija, Metropolis-Hejstings algoritam, Monte Karlo pretraga stabla (MCTS)

Čuva se (ČU): Biblioteka Departmana za matematiku i informatiku

Važna napomena (VN):

Izvod (IZ): Ovim master radom smo pokazali neke raznolike primene Monte Karlo metoda, kao i njihov značaj u statistici i programiranju. Prvo poglavlje pokazuje definiciju Monte Karlo metoda, uvodni primer za aproksimaciju broja $\boldsymbol{\pi}$ i istorijat. Sledećim poglavljem predstavljena je definicija Markovljevih lanaca, ali i definicije, teoreme i primeri potrebni za dalje delove rada. U trećoj glavi je prikaz Monte Karlo metode u integraciji koja predstavlja pristup problematici numeričke integracije. Naredno poglavlje sadrži

raznovrsne Monte Karlo simulacije za generisanje diskretnih i neprekidnih raspodela. U petom poglavlju, koristeći definicije iz drugog, predstavljamo primenu Markovljevih lanaca Monte Karlo za generisanje uzoraka iz raspodela, tj. pišemo o Metropolis-Hejstings algoritmu. Poslednja glava posvećena je Monte Karlo pretrazi stabla.

Datum prihvatanja teme od strane NN veća (DP): 02.09.2021.

Datum odbrane (DO):

Članovi komisije (KO):

Predsednik: dr Sanja Rapajić, redovni profesor, Prirodno-matematički fakultet, Univerzitet u Novom Sadu

Član: dr Danijela Rajter-Ćirić, redovni profesor, Prirodno-matematički fakultet, Univerzitet u Novom Sadu, mentor

Član: dr Dora Selesi, redovni profesor, Prirodno-matematički fakultet, Univerzitet u Novom Sadu

UNIVERSITY OF NOVI SAD

FACULTY OF SCIENCES

KEY WORDS DOCUMENTATION

Accession number (ANO):

Identification number (INO):

Document type (DT): Monograph type

Type of record (TR): Printed text

Contents Code (CC): Master's thesis

Author (AU): Smilja Rakić

Mentor (MN): dr Danijela Rajter-Ćirić

Title (TI): Some Monte Carlo methods and their applications

Language of text (LT): Serbian (Latin)

Language of abstract (LA): Serbian and English

Country of publication (CP): Republic of Serbia

Locality of publication (LP): Vojvodina

Publication year (PY): 2021.

Publisher (PU): Author's reprint

Publication place (PP): Novi Sad, Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad, Trg Dositeja Obradovića 3

Physical description (PD): (6/70/0/3/35/0/0) (number of chapters/number of pages/number of citations/number of tables/number of images/number of graphics/number of attachments)

Scientific field (SF): Mathematics

Scientific discipline (SD): Probability, Stochastic analysis

Subject/Key words (S/KW): Monte Carlo methods, Markov chains, Monte Carlo integration, Metropolis-Hastings algorithm, Monte Carlo tree search (MCTS)

Holding data (HD): The Library of the Department of Mathematics and Informatics

Note (N):

Abstract (AB): This master thesis has shown various applications of Monte Carlo method, but also their importance in statistics and programming. The first chapter shows the definition of Monte Carlo method, an introduction example for the approximation of number π and history. The definition of Markov chain is given in next chapter, as well as the definitions, theorems and examples necessary for the following parts of the thesis. The third chapter shows the Monte Carlo method in integration which represents the

approach to the problem of numerical integration. Next chapter consists of various Monte Carlo simulations for generating discrete and continuous distributions. In chapter five, using the definitions from chapter two, the author shows the application of Markov chains Monte Carlo for generating samples from distributions, that is, the author writes about Metropolis-Hastings algorithm. The last chapter focuses on the Monte Carlo tree search.

Accepted by the Scientific Board on (ASB): 02.09.2021.

Defended (DE):

Thesis defend board (DB):

President: dr Sanja Rapajić, full professor, Faculty of Science, University of Novi Sad

Member: dr Danijela Rajter-Ćirić, full professor, Faculty of Science, University of Novi Sad, mentor

Member: dr Dora Selesi, full professor, Faculty of Science, University of Novi Sad