Monte Carlo Markovljevi lanci

Bradvica, Marija Kristina

Master's thesis / Diplomski rad

2020

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Department of Mathematics / Sveučilište Josipa Jurja Strossmayera u Osijeku, Odjel za matematiku**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:126:112000

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**



Repository / Repozitorij:

Repository of School of Applied Mathematics and Computer Science







Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni diplomski studij matematike, smjer: Financijska matematika i statistika

Marija Kristina Bradvica

Monte Carlo Markovljevi lanci

Diplomski rad

Osijek, 2020.

Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni diplomski studij matematike, smjer: Financijska matematika i statistika

Marija Kristina Bradvica

Monte Carlo Markovljevi lanci

Diplomski rad

Mentor: izv. prof. dr. sc. Nenad Šuvak

Osijek, 2020.

Sadržaj

1 Uvod **3**

2 Monte Carlo metoda **4**

3 Markovljevi lanci **8**

4 Monte Carlo Markovljevi lanci (MCMC) **13**

4.1 Metropolis-Hastings algoritam 13

4.2 Nezavisni Metropolis-Hastings algoritam 15

4.3 Metropolis-Hastings algoritam sa slučajnom šetnjom 18

5 Primjer dekriptiranja poruke **24**

Dodatci **30**

Literatura **38**

Sažetak **39**

Abstract **39**

Životopis **40**

1 Uvod

Glavna je ideja Monte Carlo metoda procjena nekog parametra od interesa generiranjem slučajnih uzoraka. One se najčešće koriste kao metoda numeričke integracije, pri čemu je podintegralna funkcija takva da je njezin integral teško egzaktno izračunati. Tada se taj problem svodi na procjenu očekivanja transformacije te slučajne varijable, pri čemu se taj integral zapisuje kao matematičko očekivanje neke slučajne varijable.

Nadalje, Monte Carlo Markovljevi lanci (MCMC) metoda je čiji je zadatak pomoću zadane distribucije π na skupu S generirati slučajni uzorak iz S iz distribucije π. To se radi na način da se konstruira Markovljev lanac čija stacionarna distribucija dobro aproksimira distribuciju s gustoćom π. Na početku se na slučajan način bira vrijednost parametra koji razmatramo. Zatim simulacija generira slučajne vrijednosti (to je Monte Carlo dio). Ključ je u tome što se za svaku generiranu vrijednost može odrediti koliko je ona "dobra" poštujući neko pravilo, o čemu će biti govora u nastavku. Ukoliko je slučajno generirana vrijednost "bolja" od prethodne, dodaje se u lanac s određenom vjerojatnošću(to je dio vezan za Markovljeve lance), što ćemo zvati vjerojatnost prihvaćanja. Bitno je odrediti pravilo po kojemu se vrijednost prihvaća, odnosno dodaje u lanac, ili odbija, tj. lanac ostaje u prethodnom stanju.

O tome će biti govora u 4. poglavlju gdje je predstavljena klasa Metropolis-Hastings algoritama koji se smatraju najopćenitijim MCMC algoritmima. Započeli smo sa osnovnim algoritmom koji je jednostavan za razumijevanje, ali i za implementaciju. Zatim smo pobliže opisali dvije najčešće verzije, a to su nezavisni Metropolis-Hastings algoritam i Metropolis-Hastings algoritam sa slučajnom šetnjom. Dani su i primjeri za lakše razumijevanje.

Zadnje poglavlje ovog rada posvećeno je problemu dešifriranja poruke koja je šifrirana supstitucijskom šifrom. Na početku je bilo potrebno stvoriti matricu prijelaznih vjerojatnosti za Markovljev lanac koji će biti simuliran. To smo napravili prebrojavanjem bigrama u romanu *Zločin i kazna* ruskog pisca Dostojevskog. Bigram je slijed od dva susjedna slova. Na primjer, u riječi *ŠKOLA* bigrami su: *ŠK, KO, OL* i *LA*. Zatim ćemo vidjeti na koji način će nas simulacije iz Metropolis-Hastingsovog algoritma dovesti do rješenja.

2 Monte Carlo metoda

Monte Carlo simulacije numeričke su metode nazvane po popularnoj kockarskoj destinaciji u Monacu. Razlog je tomu što je slučajnost temelj ovih simulacija, baš kao što je i temelj kockarskih igara na sreću. U njihovoj je osnovi generiranje slučajnih brojeva iz pretpostavljene distribucije. Točnije, generiraju se pseudoslučajni brojevi koji se čine slučajnima, ali zapravo nisu istinski slučajni. Razlog je tomu što svaki generator slučajnih brojeva nakon nekog vremena počinje ponavljati brojeve jer on ponavlja jedan deterministički algoritam. Dakle, način njihova nastanka nije slučajan.

Integral je nekih funkcija vrlo teško egzaktno izračunati te se on procijenjuje raznim numeričkim metodama. To nazivamo numeričkom integracijom. Za jednodimenzionalne integralne domene postoje metode koje daju dobre rezultate poput Simpsonove i trapezne metode. Problem nastaje kada je domena višedimenzionalna. Kod većine se metoda za numeričku integraciju povećavanjem dimenzije povećava i greška procjene, ali kod Monte Carlo simulacija greška procjene ne ovisi o dimenziji. Zbog toga su one pogodne za procjenu integrala nad domenama viših dimenzija i tu se najviše primjenjuju. Pomoću njih se integral svodi na procjenu matematičkog očekivanja slučajne varijable.

Monte Carlo simulacije mogu se koristiti za rješavanje niza problema u područjima poput financija, kemije, fizike, inženjerstva, optimizacije i slično. Na primjer, mogu se koristiti za razumijevanje utjecaja rizika u modelima predviđanja nekog vremenskog niza, aproksimaciju površine lika nepravilnog oblika, predviđanje vremenske prognoze, izračunavanje vjerojatnosti pobjede na izborima i u mnogim drugim problemima.

Pod Monte Carlo simulacijom ili eksperimentom smatramo bilo koju metodu koja koristi velik broj pseudoslučajnih generiranih brojeva kako bi ispitala sve moguće ishode eksperimenta.
Osnovni je primjer izračunavanje integrala oblika:

$$\int_D f(x)\,dx, \tag{1}$$

pri čemu je f produkt dviju funkcija g i h takvih da je h funkcija gustoće neprekidne slučajne varijable X s vrijednostima u skupu $D \subset \mathbb{R}$, tj. $P(X \in D) = 1$. Dakle,

$$\int_D f(x)\,dx = \int_D g(x)h(x)\,dx = E[g(X)] = \mu. \tag{2}$$

Vidimo da je za izračunavanje gornjeg integrala zapravo potrebno procijeniti očekivanje transformacije slučajne varijable X funkcijom g. Općenito, ako imamo jednostavni slučajni uzorak $(X_1, \ldots, X_n)$, veličine $n \in \mathbb{N}$, njegovo očekivanje procijenjujemo aritmetičkom sredinom uzorka:

$$\overline{X_n} = \frac{1}{n}\sum_{i=1}^{n} X_i. \tag{3}$$

U našem slučaju, Monte Carlo procjenitelj za μ je:

$$\widehat{\mu}_n = \frac{1}{n}\sum_{i=1}^{n} g(X_i) \ . \tag{4}$$

To znači da vrijednost integrala 1 procjenjujemo aritmetičkom sredinom slučajnog uzorka $g(X_i), i = 1, ..., n$. Po jakom zakonu velikih brojeva, znamo da procjenitelj $\widehat{\mu}_n$ konvergira gotovo sigurno prema pravoj vrijednosti očekivanja μ. To znači da što više podataka simuliramo, procjena će biti bliža pravoj vrijednosti. Također, ovaj je procjenitelj nepristran:

$$E[\widehat{\mu}_n] = E\left[\frac{1}{n}\sum_{i=1}^{n} g(X_i)\right] = \frac{1}{n}\sum_{i=1}^{n} E[g(X_i)] = E[g(X)] = \mu \ . \tag{5}$$

Gornje jednakosti slijede zbog linearnosti očekivanja te zbog jednake distribuiranosti slučajnih varijabli X_i, $i = 1, \dots n$.

Koristeći ovaj primjer, navodimo osnovne korake Monte Carlo metode:

1. Na početku je potrebno definirati vezu između ulaznih i izlaznih varijabli, tj. napraviti model. Ovdje je ulazna varijabla X, a izlazna $g(X)$.

2. Zatim definiramo distribuciju ulaznih varijabli.

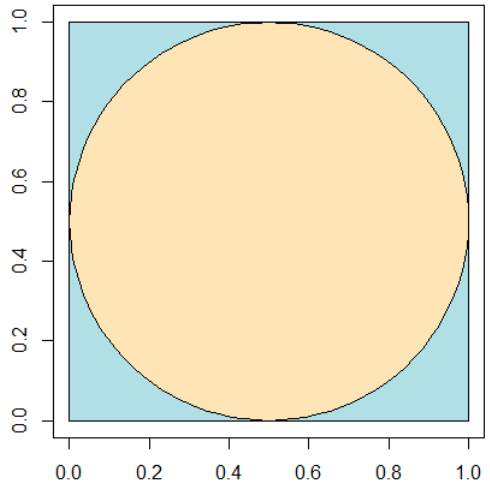
3. Generiramo n pseudoslučajnih brojeva iz distribucije zadane u koraku 2, tj. uzorak podataka $(x_1, \dots, x_n)$ koji su realizacija jednostavnog slučajnog uzorka $(X_1, \dots, X_n)$ veličine $n \in \mathbb{N}$ iz distribucije ulazne slučajne varijable X.

4. Pomoću podataka iz prethodnog koraka i modela koji smo zadali u 1. koraku, dobivamo n vrijednosti uzorka, tj. $g(x_i), i = 1, \dots, n$.

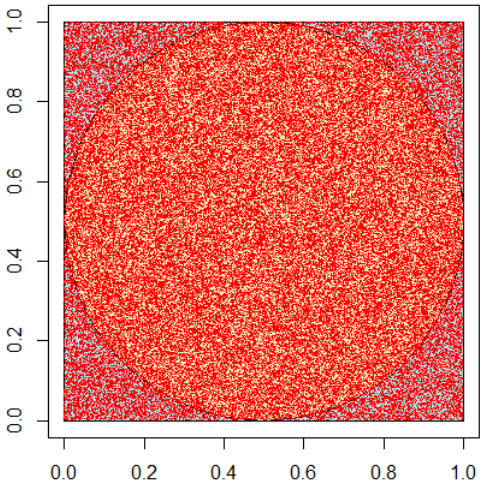
5. Pomoću dobivenih se podataka procijenjuje funkcija distribucije izlazne slučajne varijable $g(X)$. U našem primjeru, procjenjuje se i očekivanje $E[g(X)]$.

Za kraj ovog poglavlja navodimo jedan kratak primjer primjene ove metode.

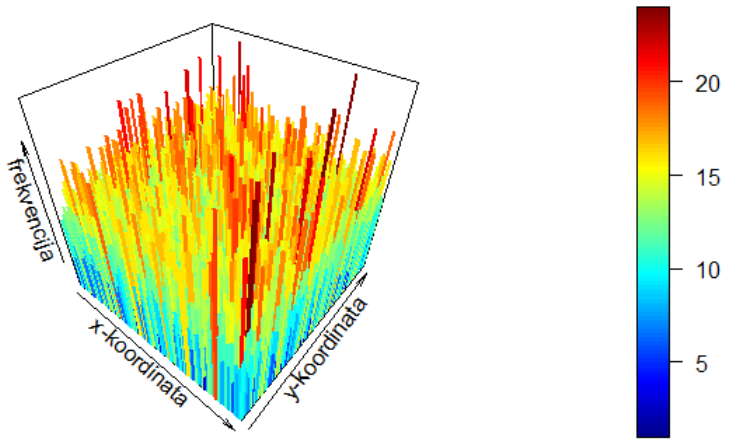
Primjer 2.1. *U ovom primjeru cilj nam je aproksimirati vrijednost konstante π. Na Slici 1 prikazan je kvadrat stranice 1 cm i njemu upisan krug čija nas površina zanima. Kako bismo aproksimirali površinu kruga, simulirali smo 100 000 točaka koje pripadaju kvadratu sa slike. To smo napravili na način da smo za x i y koordinate točaka generirali slučajne brojeve iz uniformne distribucije na segmentu $[0, 1]$. Dakle, generirali smo dva međusobno nezavisna jednostavna slučajna uzorka $(X_1, \dots, X_{100000})$ i $(Y_1, \dots, Y_{100000})$ iz distribucija $X, Y \sim U(0, 1)$. Na Slici 2, crvenom su bojom prikazane simulirane točke. Vidimo kako se neke nalaze unutar kruga, a neke ne.*



Slika 1: Kvadrat i njemu upisani krug



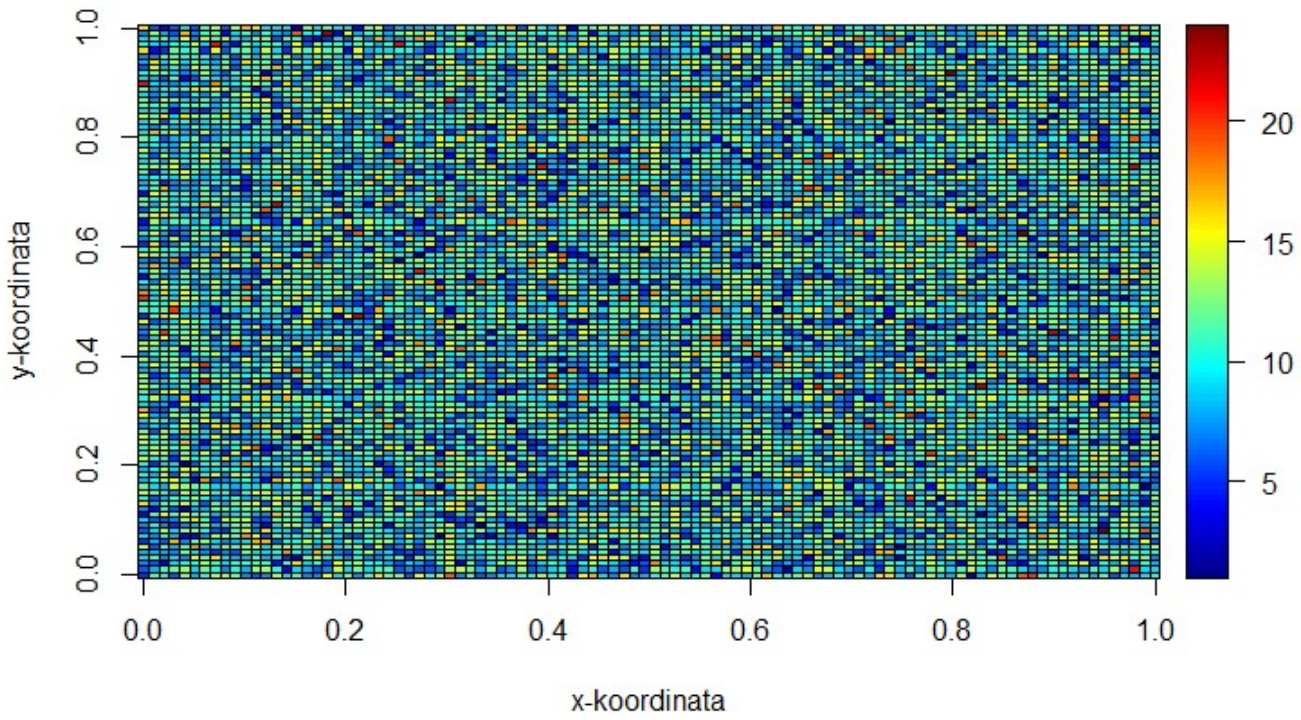
Slika 2: Simulirane točke



Slika 3: 3-D histogram frekvencija točaka

Slika 3 prikazuje frekvencije točaka na određenom području unutar kvadrata na način koji je prikazan na skali desno, tj. crvena područja sadrže više simuliranih točaka, a plava manje. Radi bolje preglednosti, pogledajmo 2-dimenzionalni prikaz na slici 4. Vidimo izmiješane boje, što znači da niti jedno područje nije posebno preferirano, a tako bi i trebalo biti.
Nadalje, za svaku smo točku (x_i, y_i) *izračunali njezinu euklidsku udaljenost od središta kruga, koje je u točki* $(0.5, 0.5)$*, prema formuli:*

$$d_2((x_i, y_i), (0.5, 0.5)) = \sqrt{(x_i - 0.5)^2 + (y_i - 0.5)^2}. \tag{6}$$



Slika 4: Shematski prikaz frekvencije točaka

Prebrojavanjem točaka čija je euklidska udaljenost od središta kruga manja ili jednaka $0.5cm$*, dobili smo prvu aproksimaciju površine kruga.*

$$g(x_1, \ldots, x_{100000}, y_1, \ldots, y_{100000}) = \frac{1}{100000} \sum_{i=1}^{100000} I_{\{\sqrt{(x_i-0.5)^2+(y_i-0.5)^2} \leq 0.5\}}(x_i, y_i) \qquad (7)$$

Imamo 78 516 točaka koje leže unutar kruga, što znači da je njegova površina $0.78516cm^2$*. Dijeljenjem tog broja sa kvadratom radijusa kruga, dobivamo* $\pi \approx 3.14064$*.*
Nadalje, ponavljanjem gornjeg postupka 1000 puta, dobivamo nove aproksimacije. Prvih nekoliko prikazano je u sljedećoj tablici:

Broj točaka u krugu	78516	78764	78682	78340	78442	78545	78556	78611	78644
Procjena broja π	3.14064	3.15056	3.14728	3.13360	3.13769	3.14180	3.14224	3.14444	3.14576

Tablica 1: Aproksimacije broja π

Za svaku iteraciju, površinu kruga procijenjujemo relativnom frekvencijom točaka koje leže unutar njega. Zatim, uzevši prosjek svih 1000 iteracija, dobivamo da je očekivana površina kruga jednaka $0.78537125cm^2$*. Uz poznat radijus kruga, slijedi da aproksimacija broja* π *iznosi* 3.141485*, a vrijednost broja* π *zaokružena na 6 decimala je* 3.141593*. Dakle, dobili smo točnost na 3 decimale.*

3 Markovljevi lanci

U ovome ćemo se poglavlju podsjetiti osnovih pojmova vezanih za homogene Markovljeve lance u diskretnom vremenu, a koje ćemo korisiti u nastavku. Najprije navodimo definiciju Markovljevog lanca.

Definicija 3.1. Neka je S diskretan skup. Slučajni proces $X = (X_n, n \in \mathbb{N}_0)$ na vjerojatnosnom prostoru $(\Omega, \mathcal{F}, P)$ s vrijednostima u skupu S je **Markovljev lanac** ako vrijedi:

$$P(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, \ldots, X_0 = i_0) = P(X_{n+1} = j | X_n = i), \tag{8}$$

za sve $n \in \mathbb{N}_0$ i sve $i_0, \ldots i_{n-1}, i, j \in S$ za koje su obje gornje vjerojatnosti dobro definirane.

Svojstvo 8 naziva se Markovljevo svojstvo. Ono kaže da je vjerojatnosno ponašanje Markovljevog lanca u neposrednoj budućnosti uvjetno na sadašnjost i prošlost jednako ponašanju tog lanca u neposrednoj budućnosti uvjetno samo na sadašnjost. Tj., uvjetno na poznatu sadašnjost, neposredna budućnost i prošlost lanca su nezavisne. Za stanja $i, j \in S$ i $s, t \in \mathbb{N}_0, s \leq t$ definiramo funkciju

$$p(i, s; t, j) = P(X_t = j | X_s = i)$$

koju nazivamo **funkcija prijelaznih vjerojatnosti**. Zanimat će nas samo 1-koračne prijelazne vjerojatnosti, tj. one za koje je $t = s + 1$. Ako one ne ovise o vremenskom trenutku t, kažemo da je lanac **homogen**. U tom slučaju za vrijeme $n \in \mathbb{N}_0$ imamo skraćeni zapis:

$$p(i, n; n + 1, j) = p_{ij}, i, j \in S.$$

Vektor vjerojatnosti $\lambda = (\lambda_i, i \in S)$, pri čemu je $\lambda_i = P(X_0 = i)$, a X_0 slučajna varijabla kojom modeliramo početno stanje Markovljevog lanca $(X_n, n \in \mathbb{N}_0)$, naziva se **početna distribucija** tog lanca. Prijelazne vjerojatnosti zapisujemo u matricu, a ovdje za ilustraciju navodimo matricu 1-koračnih prijelaznih vjerojatnosti Markovljevog lanca s konačnim skupom stanja $S = \{1, \ldots, n\}, n \in \mathbb{N}$:

$$\Pi = \begin{bmatrix} p_{11} & p_{12} \cdots & p_{1n} \\ p_{21} & p_{22} \cdots & p_{2n} \\ \vdots & \vdots & \\ p_{n1} & p_{n2} \cdots & p_{nn} \end{bmatrix} = p_{ij}, i, j \in S.$$

koju zovemo **matrica 1-koračnih prijelaznih vjerojatnosti homogenog Markovljevog lanca** $(X_n, n \in \mathbb{N}_0)$. Matrice čija je suma u svakom retku jednaka 1 nazivamo stohastičkim matricama, a Π je baš takva matrica jer su u i-tom retku zapisane uvjetne vjerojatnosti $P(X_{n+1} = j | X_n = i)$, za sve $j \in S$, što kada se sumira po stanjima $j \in S$ daje 1.

Markovljev lanac s početnom distribucijom λ i matricom 1-koračnih prijelaznih vjerojatnosti Π kraće zapisujemo kao (λ, Π)-Markovljev lanac. Znamo da je za poznavanje ponašanja slučajnog procesa dovoljno poznavati njegove konačnodimenzionalne distribucije, a za Markovljeve lance vrijedi sljedeći teorem:

Teorem 3.1. *Homogen Markovljev lanac u potpunosti je određen matricom 1-koračnih prijelaznih vjerojatnosti* Π *i distribucijom* λ *slučajne varijable* X_0*. Tada je:*

$$P(X_0 = i_0, X_1 = i_1, \ldots, X_{n-1} = i_{n-1}, X_n = i_n) = \lambda_{i_0} p_{i_0 i_1} \ldots p_{i_{n-1} i_n}. \tag{9}$$

Dokaz. Dokaz provodimo koristeći definiciju uvjetne vjerojatnosti i Markovljevo svojstvo.

$$\begin{aligned}
&P(X_0 = i_0, X_1 = i_1, \ldots, X_{n-1} = i_{n-1}, X_n = i_n) \\
&= P(X_n = i_n | X_{n-1} = i_{n-1}, X_{n-2} = i_{n-2}, \ldots, X_0 = i_0) P(X_{n-1} = i_{n-1}, \ldots, X_0 = i_0) \\
&= P(X_n = i_n | X_{n-1} = i_{n-1}) P(X_{n-1} = i_{n-1}, \ldots, X_0 = i_0) \\
&= P(X_n = i_n | X_{n-1} = i_{n-1}) P(X_{n-1} = i_{n-1} | X_{n-2} = i_{n-2} \ldots, X_0 = i_0) P(X_{n-2} = i_{n-2} \ldots, X_0 = i_0) \\
&= P(X_n = i_n | X_{n-1} = i_{n-1}) P(X_{n-1} = i_{n-1} | X_{n-2} = i_{n-2}) P(X_{n-2} = i_{n-2} \ldots, X_0 = i_0) \\
&= p_{i_{n-1} i_n} p_{i_{n-2} i_{n-1}} P(X_{n-2} = i_{n-2} \ldots, X_0 = i_0) \\
&= \ldots = p_{i_{n-1} i_n} p_{i_{n-2} i_{n-1}} \ldots p_{i_0 i_1} P(X_0 = i_0) = \lambda_{i_0} p_{i_0 i_1} \ldots p_{i_{n-1} i_n}.
\end{aligned}$$

□

Za određivanje distribucije slučajne varijable $X_n, n \in \mathbb{N}$, potrebna nam je matrica Π^n, što je vidljivo iz sljedećeg:

$$\begin{aligned}
P(X_n = i) &= P(X_n = i, X_{n-1} \in S, \ldots, X_0 \in S) \\
&= \sum_{i_{n-1} \in S} \ldots \sum_{i_0 \in S} \lambda_{i_0} p_{i_0 i_1} \ldots p_{i_{n-1} i} = (\lambda \pi^n)_i, i \in S.
\end{aligned}$$

Matricu Π^n nazivamo matrica n-koračnih prijelaznih vjerojatnosti, a njezine elemente $p_{ij}^{(n)}$, vjerojatnosti prijelaza Markovljevog lanca iz stanja i u stanje j u n koraka, n-koračnim prijelaznim vjerojatnostima.

Za skup $B \subset S$ definiramo **prvo vrijeme pogotka Markovljevog lanca u skup** B na sljedeći način:

$$T_B = min\{n \in \mathbb{N}_0 : X_n \in B\}.$$

Definicija 3.2. Za stanja $i, j \in S$ kažemo da je j **dostižno** iz i u oznaci $i \to j$, ako je $P(T_j < \infty | X_0 = i) > 0$.

Dakle, j je **dostižno** iz i ako lanac starta iz i te s pozitivnom vjerojatnošću u konačnom vremenu prvi puta posjećuje stanje j.

Definicija 3.3. Stanja i i j iz skupa S **komuniciraju** ako je $i \to j$ i $j \to i$, što označavamo s $i \leftrightarrow j$.

Relacija komuniciranja relacija je ekvivalencije na skupu $S \times S$, stoga generira particiju skupa S na klase komuniciranja, pri čemu istoj klasi pripadaju stanja koja međusobno komuniciraju.

Definicija 3.4. Markovljev lanac **ireducibilan** je ako je cijeli skup stanja S jedna klasa komuniciranja, tj. ako sva stanja međusobno komuniciraju.

Stanja mogu biti povratna i prolazna, a da bismo ih definirali treba nam sljedeća definicija. Pretpostavka je kako je za $i \in S$ $P(X_0 = i) > 0$, tj. $\lambda_i > 0$.

Definicija 3.5. Vrijeme prvog povratka Markovljevog lanca u stanje $i \in S$ je $T_i^{(1)} = min\{n \in \mathbb{N} : X_n = i\}$.

Definicija 3.6. Stanje $i \in S$ **povratno** je ako je $P(T_i^{(1)} < \infty | X_0 = i) = 1$, a **prolazno** ako je $P(T_i^{(1)} < \infty | X_0 = i) < 1$, tj. $P(T_i^{(1)} = \infty | X_0 = i) > 0$.

Dakle, nakon starta, u povratno se stanje lanac prvi puta vraća u konačnom vremenu s vjerojatnošću 1, tj. vraća se u njega beskonačno mnogo puta. U prolazno se stanje lanac prvi puta vraća u konačnom vremenu s vjerojatnošću manjom od 1. Povratnost i prolaznost svojstva su klase komuniciranja. Ako je stanje $i \in S$ povratno (prolazno), a $i \leftrightarrow j, j \in S$, tada je i j povratno (prolazno). Dokaz se može pronaći u [2](6. poglavlje, Propozicija 6.6).

Spomenimo sada jednu posebnu klasu slučajnih procesa, a to su strogo stacionarni procesi.

Definicija 3.7. Slučajni proces $(X_t, t \in T)$ (pri čemu je $T \subseteq \mathbb{R}$ skup indeksa) je **stacionaran u užem smislu (strogo stacionaran)** ako su distribucije slučajnih vektora $(X_{t_1}, \ldots, X_{t_n})$ i $(X_{t_1+h}, \ldots, X_{t_n+h})$ jednake za proizvoljne indekse $t_1, \ldots, t_n \in T$ i proizvoljan h takav da su $(t_1 + h), \ldots, (t_n + h) \in T$.

Definicija kaže da su konačnodimenzionalne distribucije procesa invarijantne na vremenske pomake. Specijalno, sve su jednodimenzionalne distribucije jednake. To znači da je $P(X_t \in B) = P(X_{t+h} \in B), \forall B \subseteq S, t \in T$ i za svaki h za koji su te dvije vjerojatnosti dobro definirane. Strogo stacionarne procese kraće ćemo nazivati stacionarnim procesima.

Definicija 3.8. Neka je $(X_n, n \in \mathbb{N}_0)$ Markovljev lanac s prebrojivim skupom stanja S i matricom 1-koračnih prijelaznih vjerojatnosti Π. Vjerojatnosna distribucija $\pi = (\pi_i, i \in S)$ na S je **stacionarna ili invarijantna distribucija** tog Markovljevog lanca ako vrijedi $\pi = \pi\Pi$, tj. po komponentama $\pi_j = \sum_{k \in S} \pi_k p_{kj}, \forall j \in S$.

Primjedba 3.1. Iz gornje definicije slijedi: $\pi = \pi\Pi = (\pi\Pi)\Pi = \pi\Pi^2 = \ldots = \pi\Pi^n, \forall n \in \mathbb{N}_0$.

Teorem 3.2. *Neka je $X = (X_n, n \in \mathbb{N}_0)$ (π, Π)-Markovljev lanac, pri čemu je π stacionarna distribucija. Tada je X stacionaran proces.*

Dokaz. Dokaz se može pronaći u [2] (Poglavlje 7, Teorem 7.4). □

Definicija 3.9. Stanje $i \in S$ je **pozitivno povratno** ako je $E[T_i^{(1)} | X_0 = i] < \infty$.

Teorem 3.3. *Neka je X ireducibilan Markovljen lanac s matricom 1-koračnih prijelaznih vjerojatnosti Π. Sljedeće su tvrdnje ekvivalentne:*

(a) svako je stanje pozitivno povratno;

(b) postoji pozitivno povratno stanje;

(c) X ima stacionarnu distribuciju π.

Dokaz se može pronaći u [2] (Poglavlje 7, Teorem 7.14).

Definicija 3.10. Neka je $X = (X_n, n \in \mathbb{N}_0)$ Markovljev lanac sa skupom stanja S i matricom 1-koračnih prijelaznih vjerojatnosti Π. Vjerojatnosna distribucija $\pi = (\pi_i, i \in S)$ naziva se **graničnom distribucijom** tog Markovljevog lanca (odnosno matrice Π) ako za sve $i, j \in S$ vrijedi: $\lim_{n\to\infty} p_{ij}^{(n)} = \pi_j$.

Propozicija 3.1. *Neka je π granična distribucija Markovljevog lanca X. Tada je ona ujedno i stacionarna distribucija.*

Dokaz se može pronaći u [2] (Poglavlje 8, Propozicija 8.3).

Definicija 3.11. Neka je X Markovljev lanac s matricom 1-koračnih prijelaznih vjerojatnosti Π. Za stanje $i \in S$ s $d(i)$ označavamo najvećeg zajedničkog djelitelja skupa $\{n \in \mathbb{N} : p_{ii}^{(n)} > 0\}$, gdje je $d(i) = 1$ ako je taj skup prazan. Kažemo da je stanje i **aperiodično** ako je $d(i) = 1$. U suprotnom kažemo da je **periodično**, a broj $d(i)$ naziva se period stanja i.

Periodičnost je svojstvo klase komuniciranja. Prema [12](Poglavlje 5.1 Ergodicity), za stanja koja su aperiodična i pozitivno povratna kažemo da su **ergodska**, a Markovljev je lanac **ergodski** ako je ireducibilan i sva su njegova stanja ergodska. Prema teoremu 3.3 kod ireducibilnog lanca postojanje jednog pozitivno povratnog stanja implicira pozitivnu povratnost svih stanja. Dakle, pozitivna povratnost svojstvo je klase komuniciranja, kao i periodičnost. Iz toga slijedi da je za ireducibilan lanac dovoljno postojanje jednog aperiodičnog stanja i jednog pozitivno povratnog te će sva biti takva.
Spomenimo još i teorem koji se može shvatiti kao generalizacija jakog zakona velikog brojeva na Markovljeve lance.

Teorem 3.4 (Ergodski teorem)**.** *Neka je Markovljev lanac $X = (X_n, n \in \mathbb{N}_0)$ ireducibilan i pozitivno povratan te π njegova jedinstvena stacionarna distribucija. Neka je f nenegativna ili ograničena realna funkcija definirana na S. Tada vrijedi:*

$$P\left(\lim_{n\to\infty} \frac{1}{n} \sum_{k=0}^{n-1} f(X_k) = \sum_{j\in S} f(j)\pi_j\right) = 1$$

Dokaz se može naći u [2] (Poglavlje 9, Teorem 9.3). Dolazimo do teorema važnog za konstrukciju Markovljevih lanaca u algoritmima koji slijede u narednim poglavljima.

Teorem 3.5. *Neka je λ proizvoljna vjerojatnosna distribucija na skupu S. Pretpostavimo da je $X = (X_n, n \in \mathbb{N}_0)$ (λ, Π)-Markovljev lanac koji je ireducibilan i aperiodičan te ima stacionarnu distribuciju π. Tada je:*

$$\lim_{n\to\infty} P(X_n = j) = \pi_j, \forall j \in S.$$

Specijalno, $\lim_{n\to\infty} p_{ij}^{(n)} = \pi_j, \forall i, j \in S.$

Prema teoremu 3.3, za ireducibilan lanac postojanje stacionarne distribucije implicira pozitivnu povratnost svih njegovih stanja te je lanac iz prethodnog teorema ergodski. Slijedi da je za ergodske lance stacionarna distribucija jednaka graničnoj. To je važna tvrdnja koja će nam trebati u nastavku.

Više o Makrovljevim lancima može se pronaći u [2].

4 Monte Carlo Markovljevi lanci (MCMC)

U ovome ćemo poglavlju objasniti princip metode Monte Carlo Markovljevih lanaca (skraćeno MCMC) i dati primjere za lakše razumijevanje. MCMC je metoda generiranja slučajnih uzoraka iz dane distribucije. Na početku se zadaje distribucija π koju nazivamo ciljna distribucija i skup stanja S. Ova metoda zatim generira Markovljev lanac čija stacionarna distribucija dobro aproksimira distribuciju π. Skup stanja S pretražuje se tako što se gledaju vjerojatnosti prelaska iz nekog stanja u novo stanje i po određenom se kriteriju odabire ono stanje koje je vjerojatnije te dodaje u lanac. Lanac koji se simulira je ergodski te je njegova stacionarna distribucija jednaka graničnoj. Dakle, što više simulacija provedemo, bit ćemo bliži ciljnoj distribuciji. Pošto je lanac ergodski, on je ireducibilan, a to nam svojstvo kaže da sva stanja međusobno komuniciraju. Dakle, bez obzira na početno stanje, lanac će posjetiti neko stanje j iz skupa stanja S u nekom trenutku s vjerojatnošću $\pi(j)$, $\pi(j) > 0$, odnosno nije osjetljiv na odabir početne distribucije. To je važno jer ne moramo brinuti o početnoj distribuciji iz koje će naš lanac krenuti.

4.1 Metropolis-Hastings algoritam

Metropolis-Hastings algoritam naziv je za klasu MCMC algoritama. Obični Metropolis-Hastings algoritam najopćenitiji je MCMC algoritam. Njegov je začetnik američki fizičar i matematičar Nicholas Metropolis [1], a poboljšao ga je kanadski matematičar Wilfred Keith Hastings[2]. Zadatak je ovog algoritma simulirati uzorke iz distribucije zadane funkcijom gustoće f koju nazivamo ciljna distribucija. Kako bismo to napravili, konstruiramo ergodski Markovljev lanac čija stacionarna distribucija dobro aproksimira distribuciju s funkcijom gustoće f kojoj se približavamo povećavanjem veličine uzorka. Uvjetna vjerojatnost $q(x, y) = q(y|x)$ označava vjerojatnost prelaska iz stanja $x \in S$ u stanje $y \in S$. Pomoću uvjetne vjerojatnosti $q(x, y)$, algoritam predlaže stanja koja se ili odbijaju ili prihvaćaju te dodaju u lanac. Nakon navođenja algoritma, postupak će biti detaljnije objašnjen. Dovoljno je poznavati f i q do na normalizirajuću konstantnu, što čini ovaj algoritam jednostavnim za korištenje. Na početku krećemo iz stanja $x_0 \in S$ za koje je $f(x_0) > 0$. Zatim pratimo algoritam 1.

U prvom koraku generiramo novo stanje na osnovu prijelazne vjerojatnosti $q(y|x)$. Kako bismo donijeli odluku o prelasku u novo stanje, moramo definirati kriterij koji ćemo slijediti. Pretpostavimo da imamo novčić nepravilnog oblika, pri čemu je vjerojatnost okretanja glave

[1] Živio je između 1915.-1999. Radio je kao asistent, zatim kao profesor na sveučilištu u Chicagu i kao znanstvenik te viši suradnik u nacionalnom laboratoriju u Los Alamosu. Dobitnik je nagrade "Computer Pioneer Award" od strane Instituta inženjera elektrotehnike i elektronike te je nagrađivan za svoje izvanredne znanstvene radove.

[2] Živio je između 1930. i 2016. godine. Studirao je matematiku na sveučilištu u Torontu. Radio je na sveučilištu Canterbury, na Novom Zelandu, zatim u "Nokia Bell Labs" laboratoriju u New Yerseyu te se nakon toga vraća u Toronto, gdje je 5 godina radio na sveučilištu kao izvanredni profesor. Njegovo posljednje radno mjesto bilo je mjesto izvanrednog profesora na sveučilištu u Viktoriji.

Algoritam 1 Metropolis-Hastings algoritam

0: Neka je $X_n = x_n$.
1: Generiraj y_n iz $Y_n \sim q(y|X_n)$.
2: Generiraj u iz $U \sim U(0,1)$.
3: Izračunaj $\rho(x_n, y_n)$, gdje je

$$\rho(x_n, y_n) = min\left\{\frac{f(y_n)}{f(x_n)}\frac{q(x_n|y_n)}{q(y_n|x_n)}, 1\right\}.$$

4: Definiraj $x_{n+1} = \begin{cases} y_n, & \text{ako } u \leq \rho(x_n, y_n) \\ x_n, & \text{inače} \end{cases}$.
5: Ponavljaj za $n = n + 1$.

u koji dolazi iz uniformne distribucije na nosaču $[0, 1]$. Zatim gledamo umnožak omjera:

$$\frac{f(y)}{f(x)}\frac{q(x|y)}{q(y|x)}. \tag{10}$$

Prvi faktor uspoređuje vjerojatnosti dva stanja, a drugi vjerojatnost prelaska iz predloženog u trenutno stanje i obrnuto, iz trenutnog u predloženo stanje. Ako je gornji izraz veći od 1, to znači da je novo stanje vjerojatnije od trenutnog. Tada je $\rho(x_n, y_n) = 1$, što znači da u koraku 4 algoritma 1 prelazimo u novo stanje jer je $u \in [0, 1]$. Ako je manji od 1, bacamo novčić i okretanjem glave prijeći ćemo u novo stanje, a okretanjem pisma ostajemo u starom. To znači da u svako stanje možemo doći s nekom vjerojatnošću koja je različita od nule. To je jako bitno jer nam omogućava da algoritam ne "zapne" u lokalnim ekstremima. Sada vidimo kako je f i q dovoljno poznavati do na konstante jer nam je za račun potreban samo njihov omjer. Vjerojatnost $\rho(x_n, y_n)$ naziva se vjerojatnost prihvaćanja, a ona nam govori kolika je vjerojatnost prelaska u novo predloženo stanje. Ovaj se algoritam zapravo zasniva na ideji pokušaja i pogrješke. Dakle, u 4. koraku formira se novo stanje u ovisnosti o prethodnom stanju pa je jasno kako je riječ o konstrukciji Markovljevog lanca.
Primjetimo da, ukoliko je q simetrična funkcija, tada u izrazu 10 ostaje samo prvi faktor. Upravo je početni Metropolisov algoritam bio takav, a poslije ga je Hastings poopćio.

Niz je varijabli dobiven ovim algoritmom zavisan jer je svaka varijabla definirana pomoću prethodne i distribucija slučajne varijable X_n za veliki n približno je jednaka distribuciji s gustoćom f. Dakle, u simuliranom će uzorku postojati autokorelacija. To znači da lanac duljine 5000 dobiven ovim algoritmom daje manje informacija od nezavisnog slučajnog uzorka duljine 5000 jer se u lancu javlja redundancija, tj. višak informacija zbog ponavljanja. Nas zanima koliko "manje" informacija nam daje taj lanac, odnosno koliko u našem lancu ima slučajnih varijabli koje nisu međusobno korelirane.
Efektivna veličina uzorka(eng. effective sample size, ESS) broj je koji nam kazuje koliko je nezavisne informacije prisutno u lancu $(X_n, n \in \mathbb{N})$ generiranom gornjim algoritmom.

Računa se na sljedeći način:

$$ESS = \frac{N}{1 + 2\sum_{k=1}^{\infty}\sum_{t=0}^{N}\rho(X_t, X_{t+k})}, \tag{11}$$

pri čemu N predstavlja veličinu simuliranog lanca, tj. broj iteracija, a $\rho(X_t, X_{t+k})$ autokorelaciju lanca s pomakom za k koraka koja se definira na sljedeći način:

$$\rho(X_t, X_{t+k}) = \frac{\gamma(X_t, X_{t+k})}{\sigma_{X_t}\sigma_{X_{t+k}}}.$$

U gornjoj jednadžbi, $\gamma(X_t, X_{t+k})$ predstavlja kovarijancu slučajnih varijabli X_t i X_{t+k}, a σ_{X_t} i $\sigma_{X_{t+k}}$ njihove su standardne devijacije. Kada uspoređujemo više lanaca, bolji je onaj koji ima veću efektivnu veličinu uzorka.

4.2 Nezavisni Metropolis-Hastings algoritam

Kod ovog specijalnog slučaja Metropolis-Hastings algoritma predložena distribucija q ne ovisi o prethodnom stanju u kojem se lanac nalazi. Zbog toga nosi ime "nezavisni". Predloženu distribuciju možemo kraće pisati samo kao $q(y), y \in S$ jer ona ovisi samo o stanju u kojem se lanac trenutno nalazi.

Algoritam 2 Nezavisni Metropolis-Hastings algoritam

0: Neka je $X_n = x_n$.
1: Generiraj y_n iz $Y_n \sim q(y)$.
2: Generiraj u iz $U \sim U([0,1])$.
3: Izračunaj $\rho(x_n, y_n)$, gdje je

$$\rho(x_n, y_n) = min\left\{\frac{f(y_n)}{f(x_n)}\frac{q(x_n)}{q(y_n)}, 1\right\}.$$

4: Definiraj $x_{n+1} = \begin{cases} y_n, & \text{ako } u \leq \rho(x_n, y_n) \\ x_n, & \text{inače} \end{cases}$.
5: Ponavljaj za $n = n + 1$.

Dakle, u ovom algoritmu svakom se iteracijom generira realizacija slučajne varijable iz iste vjerojatnosne distribucije q. Iako su generirani brojevi realizacije nezavisnih slučajnih varijabli, dobiveni uzorak nije nezavisan jer se u svakoj iteraciji promatra omjer vjerojatnosti koje se odnose na prethodno i na novo, predloženo stanje, odnosno vjerojatnost prihvaćanja $\rho(x_n, y_n)$ ovisi o prethodnom stanju x_n.

Za bolje razumijevanje, pogledajmo primjer koji slijedi.

Primjer 4.1. *U ovome ćemo primjeru uz pomoć generiranih vrijednosti iz uniformne distribucije na intervalu* $[0,1]$ *simulirati beta distribuciju s parametrima 3 i 5, tj.* $Be(3,5)$.

Podsjetimo se kako je beta distribucija definirana na intervalu $[0,1]$ *i parametrizirana s dva pozitivna parametra oblika,* α *i* β*. Njezina je funkcija gustoće:*

$$f(x) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)} I_{[0,1]}(x), \qquad B(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)},$$

pri čemu je Γ *Gamma funkcija. Ovdje jasno vidimo da će nam za izračunavanje vjerojatnosti prihvaćanja* $\rho(x_n, y_n)$ *biti dovoljno računati*

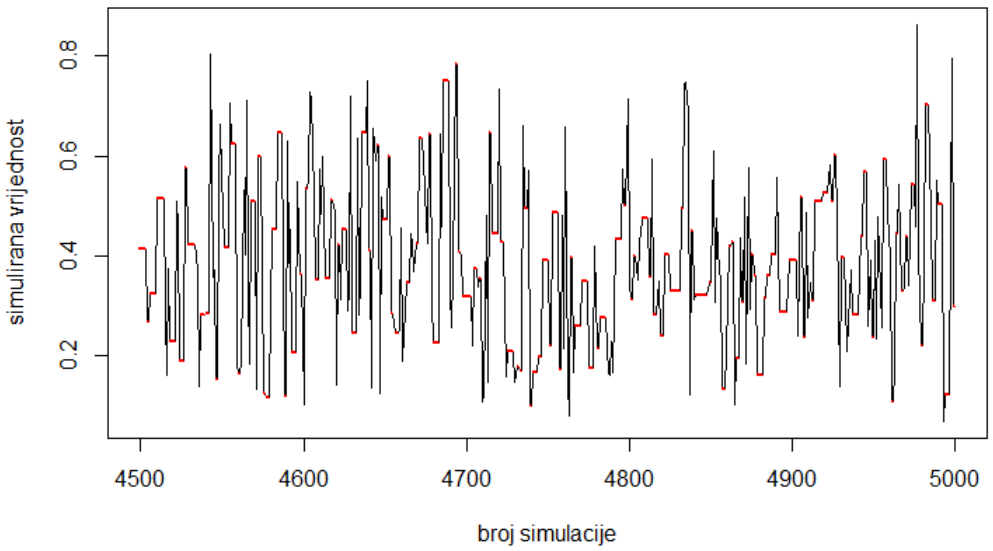
$$\tilde{f}(x) = x^{\alpha-1}(1-x)^{\beta-1}$$

jer će se konstanta $B(\alpha,\beta)$ *skratiti pošto se u kvocijentu* $\dfrac{f(y_n)}{f(x_n)}$ *pojavljuje i u brojniku i nazivniku. Dakle, ciljna distribucija je* $Be(3,5)$*, a predložena* $U[0,1]$*. Za predloženu smo distribuciju mogli uzeti bilo koju distribuciju definiranu na skupu koji je nadskup segmenta* $[0,1]$*. Podsjetimo se još kako izgleda gustoća predložene uniformne distribucije:*

$$q(x) = I_{[0,1]}(x)$$

Predložena je distribucija simetrična i ne ovisi o prethodnoj vrijednosti, tj. $q(x_{n+1})$ *ne ovisi o* x_n*, za* $n \in \mathbb{N}$*. Zbog njezine se simetrije izraz za* $\rho(x_n, y_n)$ *pojednostavljuje pa* $\rho(x_n, y_n)$ *možemo računati na sljedeći način:*

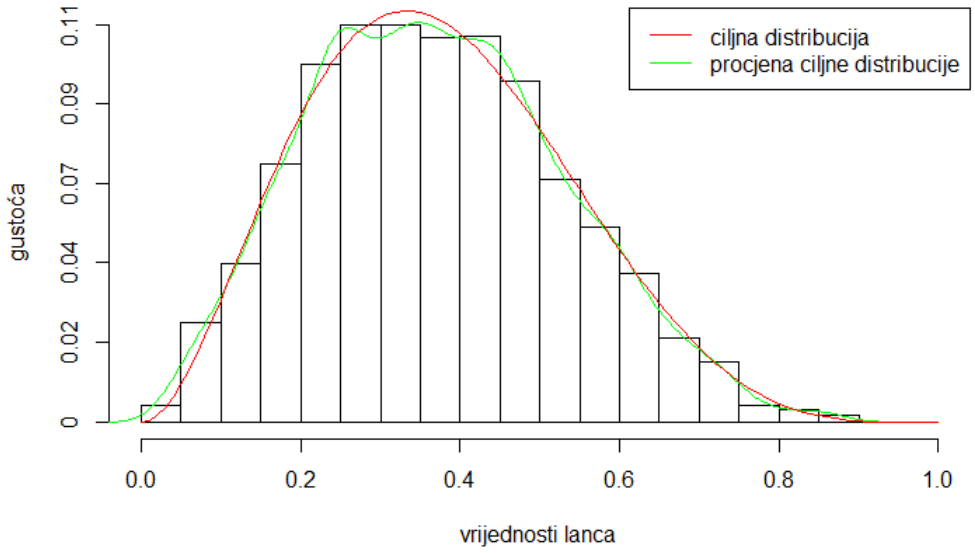
$$\rho(x_n, y_n) = \frac{\tilde{f}(y_n)}{\tilde{f}(x_n)}.$$



Slika 5: Simulirani niz od 4500. do 5000. iteracije

Gornji graf prikazuje posljednjih 500 simuliranih vrijednosti. Obično se prvi dio lanca ne prikazuje jer se smatra "burn in" periodom, tj. dijelom lanca u kojem se traži put do mjesta

gdje je ditribucija lanca bliža njegovoj stacionarnoj distribuciji. Nakon određenog broja iteracija lanac se približava svojoj stacionarnoj distribuciji. Linije označene crvenom bojom prikazuju mjesta na kojima se vrijednost kroz simulacije nije mijenjala. To su mjesta na kojima u 4. koraku gornjeg algoritma nije prihvaćena nova predložena vrijednost generirana u 1. koraku. Broj prihvaćenih koraka (eng. acceptance rate) u cijelom lancu iznosi 2616. To je broj iteracija u kojima je algoritam prihvatio predloženu vrijednost i stavio ju u lanac. Za nezavisni Metropolis-Hastings algoritam bolji je onaj lanac koji prihvaća više koraka. U idealnom bi slučaju kod ovog algoritma bili prihvaćeni svi koraci.



Slika 6: Histogram simuliranih podataka

Na gornjem grafu prikazan je histogram simuliranih vrijednosti lanca i pripadna (zaglađena) empirijska gustoća (zeleno), kao i funkcija gustoće beta distribucije $Be(3,5)$. Ako pogledamo cijeli lanac kao uzorak, daje se naslutiti kako pripadni histogram dobro aproksimira ciljnu beta ditribuciju. Provjerimo to Kolmogorov-Smirnovljevim testom za testiranje jednakosti dvaju distribucija. Dobivena je p-vrijednost 0.3036 te na razini značajnosti 0.05 ne odbacujemo nul-hipotezu o jednakosti distribucija, tj. nemamo dokaza na temelju kojih bismo mogli tvrditi da su te dvije distribucije različite. Također, ako pogledamo očekivanje i varijancu ovih dviju distribucija vidimo da su te numeričke karakteristike približno jednake:

	Predložena distribucija	Ciljna distribucija
Očekivanje	0.3751171	0.375
Varijanca	0.02606829	0.02604167

Tablica 2: Prikaz parametara

4.3 Metropolis-Hastings algoritam sa slučajnom šetnjom

Ako želimo skup stanja pretraživati lokalno tako da se iz trenutnog stanja pomičemo u neko njemu susjedno blisko stanje, tada ćemo prilikom generiranja predloženog stanja u obzir uzeti i trenutno. Novo će stanje biti zadano kao:

$$Y_n = X_{n-1} + \epsilon_n.$$

Distribuciju q slučajne varijable ϵ_n možemo zadati kao $\mathcal{N}(0, \sigma^2)$. Tada će distribucija predloženog stanja biti $Y_n \sim \mathcal{N}(X_{n-1}, \sigma^2)$. Dakle, Y_n ovisi o prethodnom stanju lanca i o standardnoj devijaciji koju zadajemo. Markovljev lanac generiran na ovaj način spada u kategoriju slučajnih šetnji. U ovom algoritmu zadajemo distribuciju q slučajne varijable ϵ_n te preko nje računamo distribuciju slučajne varijable Y_n. Tada je zapravo $q(y|x) = q(y - x)$ jer je $\epsilon_n = Y_n - X_{n-1}$. Za distribuciju q slučajne varijable ϵ_n najčešće uzimamo simetrične distribucije poput normalne ili studentove. Tada je $q(y|x) = q(y - x) = q(x - y) = q(x|y)$.

Algoritam 3 Metropolis-Hastings algoritam sa slučajnom šetnjom

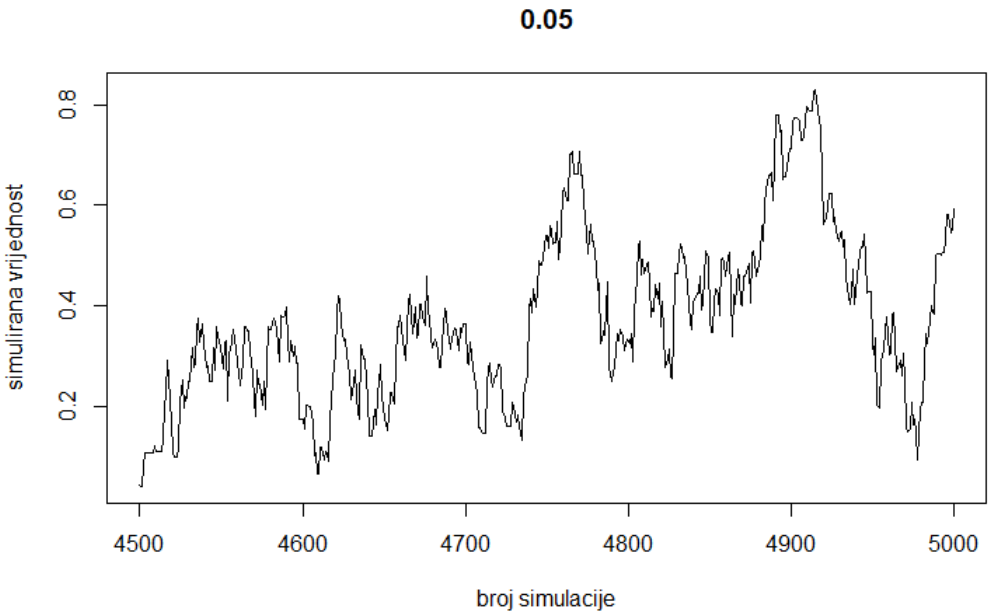
0: Neka je $X_n = x_n$.
1: Generiraj y_n iz $Y_n \sim q(y)$.
2: Generiraj u iz $U \sim U([0, 1])$.
3: Izračunaj $\rho(x_n, y_n)$, gdje je

$$\rho(x_n, y_n) = min\left\{\frac{f(y_n)}{f(x_n)}, 1\right\}.$$

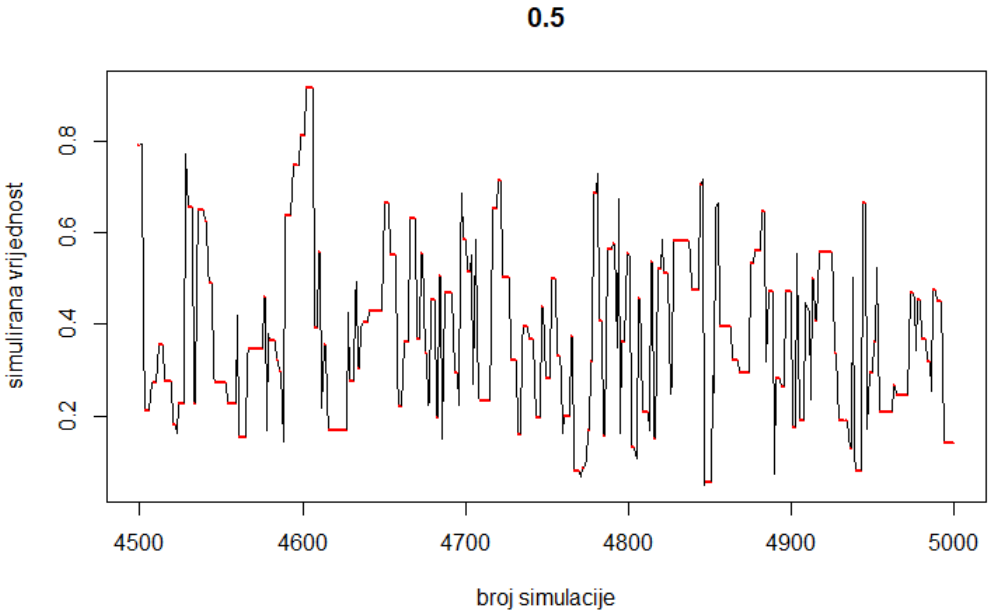
4: Definiraj $x_{n+1} = \begin{cases} y_n, & \text{ako } u \leq \rho(x_n, y_n) \\ x_n, & \text{inače} \end{cases}$.
5: Ponavljaj za n=n+1.

Kod vjerojatnosti prihvaćanja gubi se drugi faktor jer je distribucija q simetrična. Prijeđimo sada na primjer.

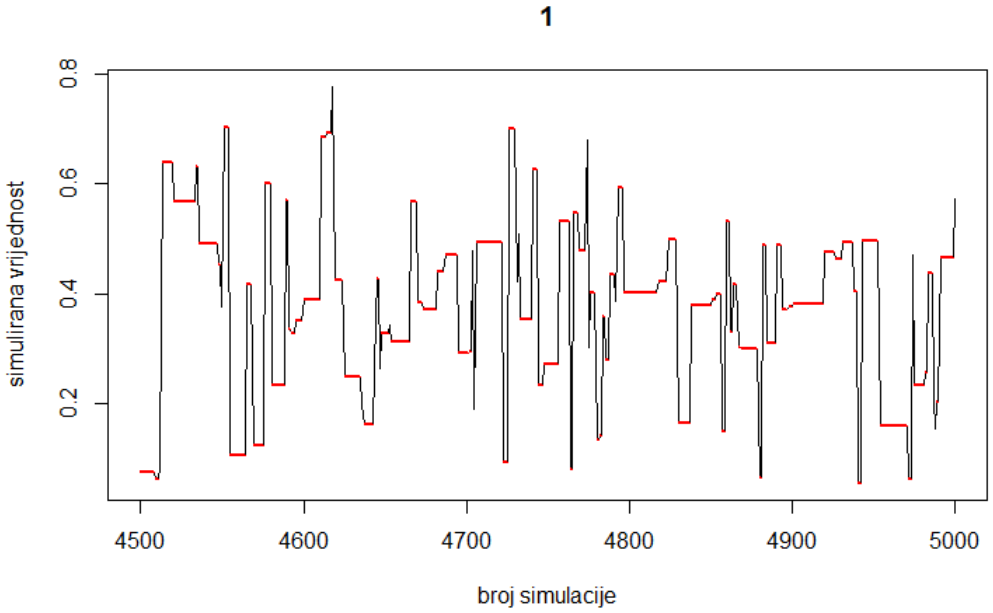
Primjer 4.2. *U ovome ćemo primjeru simulirati beta distribuciju s parametrima 3 i 5, baš kao u primjeru 4.1, ali ćemo ovdje koristiti Metropolis-Hastings algoritam sa slučajnom šetnjom. Distribucija varijable ϵ_n bit će normalna s očekivanjem 0 i standardnom devijacijom σ. Kao što smo gore već objasnili, to znači da će slučajna varijabla Y_n imati normalnu distribuciju s očekivanjem koje je jednako prethodnoj vrijednosti lanca, tj. X_{n-1} i standardnom devijacijom σ. U nastavku ćemo se baviti usporedbom simulacija za različite standardne devijacije. Razmatrat ćemo slučajeve kada je σ jednaka 0.05, 0.5, 1 i 1.5. Svaku smo simulaciju započeli iz točke 0.5. Prvo ćemo prikazati zadnjih 500 simulacija, zatim histogram svih simuliranih podataka, a za kraj pomoću vrijednosti očekivanja i varijance, broja prihvaćenih koraka i efektivne veličine uzorka prokomentirati za koji odabir parametra σ dobivamo najbolje rezultate.*



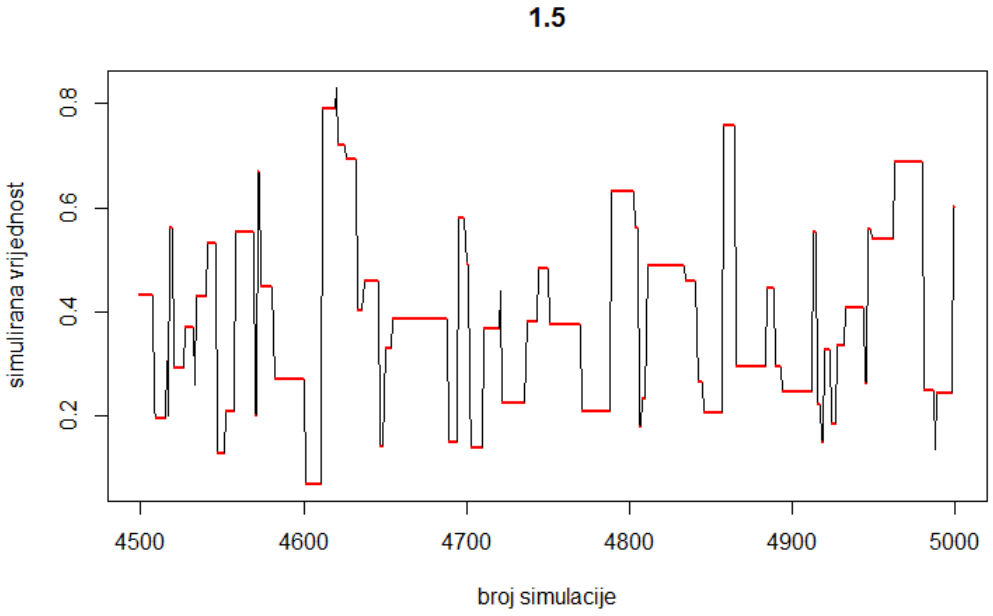
Slika 7: Zadnjih 500 simulacija za $\sigma = 0.05$



Slika 8: Zadnjih 500 simulacija za $\sigma = 0.5$



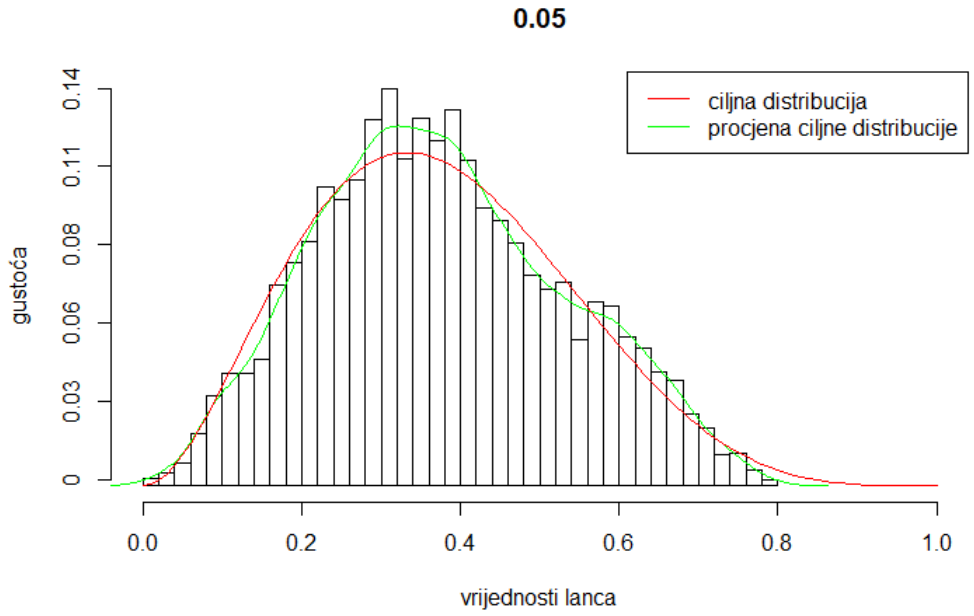
Slika 9: Zadnjih 500 simulacija za $\sigma = 1$



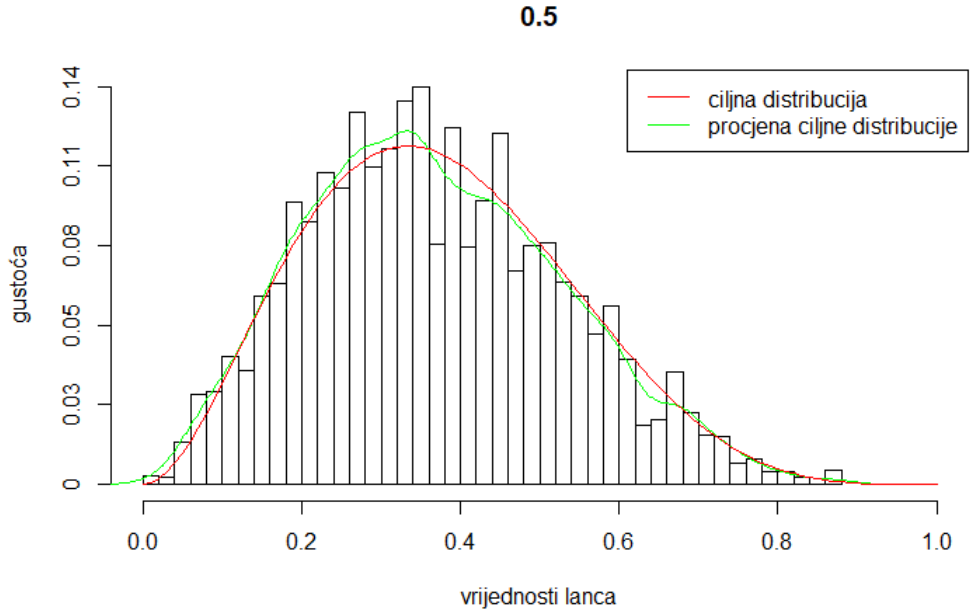
Slika 10: Zadnjih 500 simulacija za $\sigma = 1.5$

Iz gornjih prikaza vidimo da se za male vrijednosti standardnih devijacija pomičemo jako sporo, što nije baš poželjno jer će biti potreban velik broj simulacija kako bismo distribuciju lanca doveli bliže stacionarnoj. Pri izboru većih vrijednosti standardnih devijacija vidimo da lanac jako "skače", što također nije poželjno jer ako u nekom trenutku dođemo u mjesto gdje je distribucija lanca blizu stacionarne, ali to nije stacionarna, lako možemo iz njega "iskočiti" u neko mjesto gdje je distribucija daleko od stacionarne distribucije. Dakle, treba

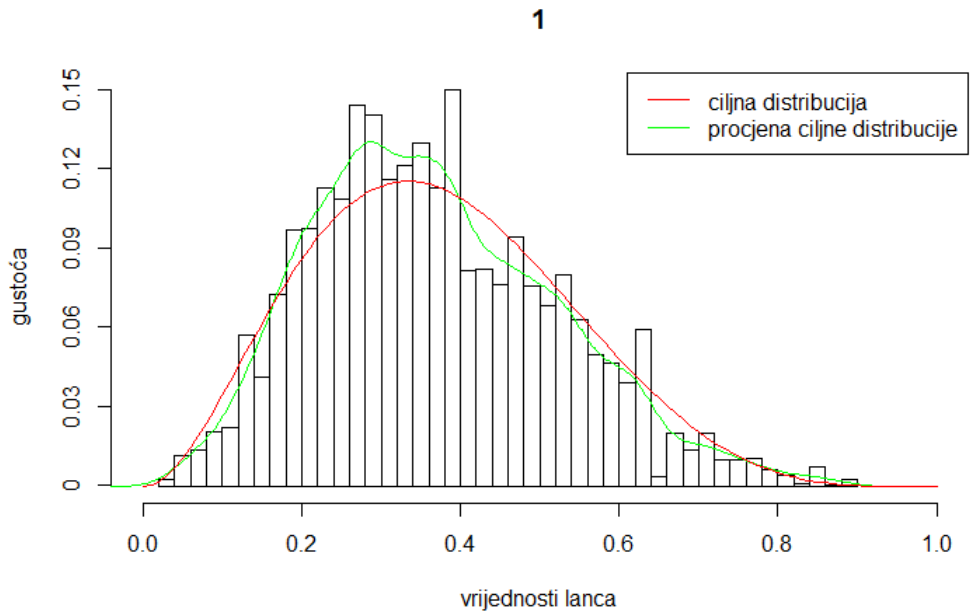
optimizirati brzinu pomicanja.



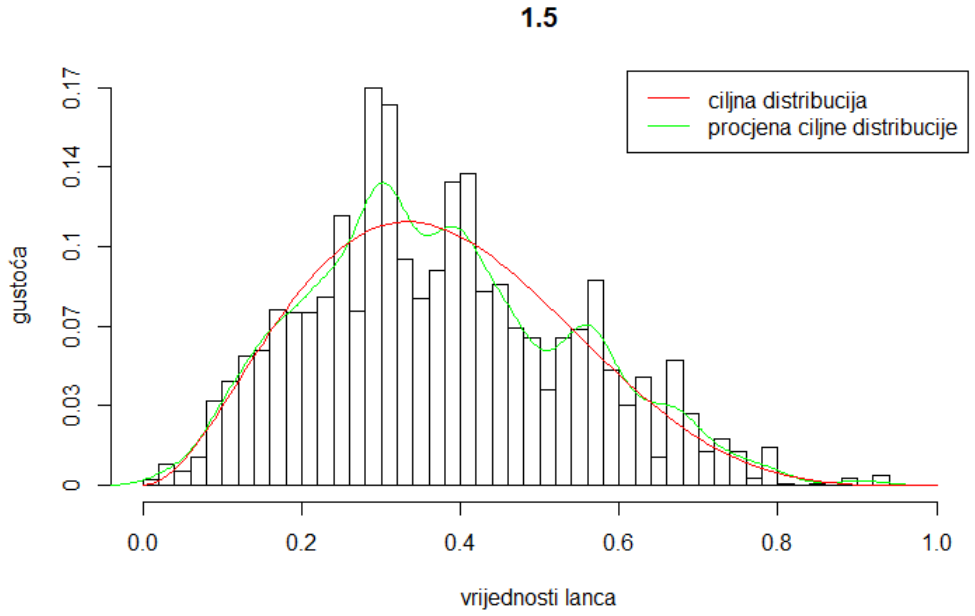
Slika 11: Histogram simuliranih podataka za $\sigma = 0.05$



Slika 12: Histogram simuliranih podataka za $\sigma = 0.5$



Slika 13: Histogram simuliranih podataka za $\sigma = 1$



Slika 14: Histogram simuliranih podataka za $\sigma = 1.5$

Iz gornjih bi se histograma bi se reći da se ciljna i predložena distribucija najbolje podudaraju za $\sigma = 0.5$. *Pogledajmo jesu li očekivanje i varijanca simuliranih uzoraka bliske teorijskom očekivanju i varijanci.*

	Ciljana distribucija	$\mathcal{N}(X_{n-1}, 0.05^2)$	$\mathcal{N}(X_{n-1}, 0.5^2)$	$\mathcal{N}(X_{n-1}, 1^2)$	$\mathcal{N}(X_{n-1}, 1.5^2)$
Očekivanje	0.375	0.3810518	0.3741636	0.3751455	0.3708218
Varijanca	0.02604167	0.02786754	0.02719336	0.02738467	0.03228286

Tablica 3: Prikaz parametara

Iz gornje tablice vidimo da su očekivanje i varijanca najbliži teorijskim vrijednostima za izbor 0.5 *i* 1 *za parametar* σ.

	$\mathcal{N}(X_{n-1}, 0.05^2)$	$\mathcal{N}(X_{n-1}, 0.5^2)$	$\mathcal{N}(X_{n-1}, 1^2)$	$\mathcal{N}(X_{n-1}, 1.5^2)$
Prihvaćeni koraci	4577	1899	1020	740
ESS	110.7792	1054.758	668.1201	456.6609

Tablica 4: Prikaz mjera za usporedbu kvalitete simulacija

Već smo vidjeli da za male vrijednosti parametra σ *postižemo male pomake, a to znači da su vrijednosti* $f(x_n)$ *i* $f(y_n)$ *jako blizu pa je njihov kvocijent približno jedan, što znači da je vjerojatnost prihvaćanja približno jedan. Dakle, za male ćemo vrijednosti standardnih devijacija prihvatiti jako puno koraka. Ovdje smo od 5000 koraka prihvatili 4577 za* $\sigma = 0.05$. *Skup stanja pretražuje se sporo, što nije dobro. Efektivna je veličina uzorka mala jer u lancu ima dosta redundantnih informacija.*
Obrnuto, za velike vrijednosti standardnih devijacija brojevi x_n *i* y_n *jako su udaljeni pa se ponekad dogodi da predloženo stanje nije u skupu* $[0, 1]$, *koji je nosač beta distribucije. Tada ono biva automatski odbijeno i zbog toga je broj prihvaćenih koraka jako mali. To možemo vidjeti i na slici 10 gdje konstantni crveno obojeni dijelovi grafičkog prikaza predstavljaju mjesta na kojima predložena vrijednost nije prihvaćena. To, također, znači da i ovaj lanac sadrži puno redundantnih informacija. Zbog toga je i ovdje efektivna veličina uzorka relativno mala.*
Vidimo da su najbolji kandidati za σ *vrijednosti 0.5 i 1, iako bismo za 0.5 mogli reći da je bolji kandidat jer je za taj lanac ESS najveća.*

Zaključujemo da kod Metropolis-Hastings algoritma sa slučajnom šetnjom broj prihvaćenih koraka ne mora biti velik, kao što je to bio slučaj sa prethodnim algoritmom.

5 Primjer dekriptiranja poruke

U ovome ćemo se poglavlju baviti dekriptiranjem poruke koristeći prethodno navedeni Metropolis-Hastings algoritam. Može se reći da je kriptografija znanstvena disciplina koja se bavi proučavanjem metoda za slanje poruka u takvom obliku da ih onaj kome su namijenjene može pročitati, ali ostali ne mogu. Osobu koja šalje poruku nazivamo *pošiljatelj*, a onu koja ju prima *primatelj*. Pošiljatelj piše poruku koju zovemo *otvoreni tekst* te ju transformira koristeći dogovoreni *ključ* koji je poznat i primatelju. Taj se postupak naziva *šifriranje* ili *kriptiranje*, a dobivena poruka *šifrat* ili *kriptogram*. Zatim pošiljatelj šalje šifrat putem nekog nesigurnog komunikacijskog kanala prema primatelju. Tu poruku može presresti protivnik, ali bez poznavanja ključa ne može otkriti njezin sadržaj.

Definicija 5.1. Kriptosustav je uređena petorka $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ za koju vrijedi:

(a) $\mathcal{P}$ je konačan skup svih mogućih elemenata otvorenog teksta

(b) $\mathcal{C}$ je konačan skup svih mogućih elemenata šifrata

(c) $\mathcal{K}$ je konačan skup svih mogućih ključeva

(d) $\mathcal{E}$ je skup svih funkcija šifriranja

(e) $\mathcal{D}$ je skup svih funkcija dešifriranja

(f) $\forall K \in \mathcal{K}$ postoji funkcija šifriranja $e_K \in \mathcal{E}$ i odgovarajuća funkcija dešifriranja $d_K \in \mathcal{D}$. Pritom su $e_k : \mathcal{P} \to \mathcal{C}$ i $d_k : \mathcal{C} \to \mathcal{P}$ funkcije sa svojstvom da je $d_K(e_K(x)) = x$, za svaki otvoreni tekst $x \in \mathcal{P}$.

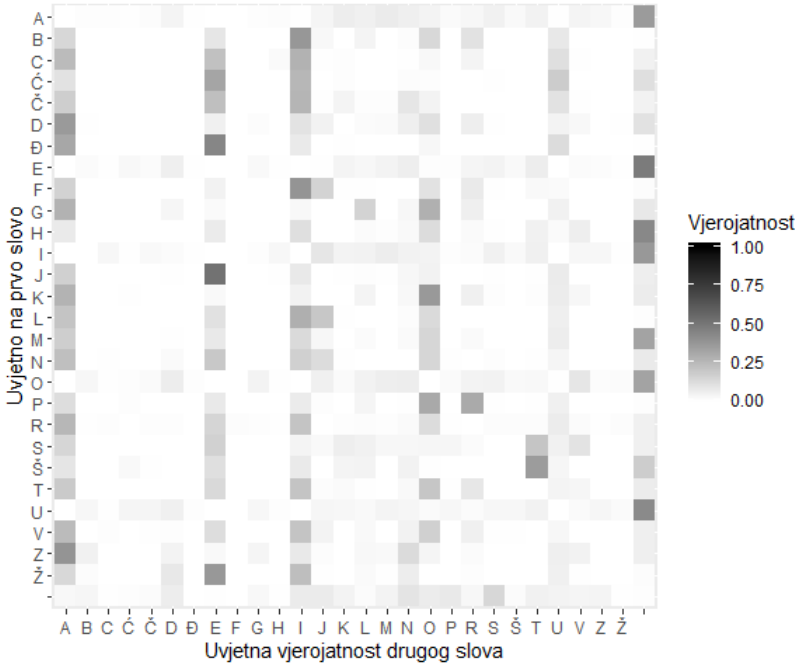
Primjetimo da, ukoliko je $\mathcal{P} = \mathcal{C}$, funkcije šifriranja i dešifriranja su permutacije, a to će biti tako u primjeru koji slijedi. Obzirom na tip operacija koje se koriste pri šifriranju, razlikujemo:

(a) supstitucijske šifre: svaki se element otvorenog teksta zamjenjuje nekim drugim elementom prema unaprijed određenoj transformaciji

(b) transpozicijske šifre: elementi se otvorenog teksta međusobno permutiraju.

U ovom će primjeru poruka biti šifrirana supstitucijskom šifrom. Za početak trebat ćemo matricu prijelaznih vjerojatnosti Markovljevog lanca koji želimo konstruirati. Nju ćemo dobiti prebrojavanjem svih bigrama iz nekog referentnog teksta koji je dovoljno dugačak i za koji smatramo da konzistentno predstavlja frekvenciju bigrama u hrvatskom jeziku. Bigram je slijed od dva susjedna slova. Na primjer, u riječi *ŠKOLA* bigrami su: *ŠK, KO, OL* i *LA*. Također, bitno je da je tekst pisan hrvatskim književnim jezikom, a ne nekim određenim narječjem. Za referentni tekst odabrali smo djelo ruskog književnika Fjodora Mihajloviča Dostojevskog *Zločin i kazna* prevedeno na hrvatski jezik jer ono zadovoljava prethodne uvjete. Iz tog smo teksta za svako slovo bilježili frekvenciju pojavljivanja slova

koje slijedi iza njega. Promatrali smo hrvatsku abecedu bez slova "dž", "lj" i "nj" jer su ona ujedno i bigrami pa nam stvaraju problem u ovoj analizi. Dakle, iz naše abecede sada preostaje 27 slova. Kada bismo neku poruku šifriranu supstitucijskom šifrom pokušali ručno dešifrirati, tj. tehnikom grube sile, to bi bilo jako teško jer postoji $27 \cdot \ldots \cdot (27-n+1)$ takvih načina šifriranja, pri čemu je $n \in \mathbb{N}$ broj različitih slova u šifratu. Stoga se okrećemo nekim efikasnijim metodama.



Slika 15: Shematski prikaz matrice prijelaznih vjerojatnosti

Na gornjoj slici vidimo shematski prikaz matrice prijelaznih vjerojatnosti. Nju smo dobili iz matrice frekvencija dodavši svakom elementu broj 1 (kako ne bismo imali vjerojatnosti jednake nuli) i podijelivši svaki element sa sumom elemenata iz tog retka. Intenzitet crne boje u pojedinom kvadratiću govori o vjerojatnosti da je sljedeće slovo ono koje je upisano u tom retku, uz uvjet da se "nalazimo" na slovu koje je upisano za taj stupac. Tako, na primjer, tamni kvadratić, koji se nalazi u retku sa slovom "J" i u stupcu sa slovom "E", odražava kolika je vjerojatnost da iza slova "E" slijedi slovo "J". Posljednji redak i stupac matrice predstavljaju razmak. Tako smo dobili bolji uvid u to koje će slovo vjerojatnije biti na kraju riječi, a koje na početku. Vidimo da je slovo "S" najvjerojatnije na početku riječi, dok je na kraju riječi najvjerojatnije neki od samoglasnika. Također, vidimo da su samoglasnici česti sljedbenici većine slova. I na kraju, iz matrice slijedi da je najfrekventniji bigram "JE", a to je poznata činjenica. Po svemu ovome, izgleda da je referentni tekst dobro odabran.

Nakon prebrojavanja bigrama, konstruirali smo funkciju pomoću koje vršimo dešifriranje. Njoj predajemo ključ i šifrat, a ona vraća tekst dešifriran prema tom ključu. Također, trebat će nam i funkcija koja računa vjerodostojnost danog teksta, odnosno koliko je vjerojatno da

je dani tekst traženi otvoreni tekst, tj. koliko on ima smisla. Vjerodostojnost nekog teksta računamo kao umnožak prijelaznih vjerojatnosti svih bigrama koji se nalaze u njemu. Kako bismo izbjegli računanje umnoška, logaritmirat ćemo sve vrijednosti iz matrice prijelaznih vjerojatnosti i zbrajati ih. To nam daje ekvivalentan rezultat.

Algoritam 4 Metropolis-Hastings algoritam za dešifriranje

0: Neka je X_0 neki ključ (mapiranje).
Za i od 1 do N ponavljaj:
1: Predloži novi ključ X_i^* uz pretpostavljenu distribuciju.
2: Dešifriraj pomoću predloženog ključa i izračunaj "log-likelihood" novog teksta $f(X_i^*)$
3: Generiraj u iz $U \sim U([0,1])$.
4: Izračunaj $\rho(X_i, X_i^*)$, gdje je

$$\rho(X_i, X_i^*) = min\left\{e^{f(X_i^*)-f(X_i)}, 1\right\}.$$

5: Definiraj $X_{i+1} = \begin{cases} X_i^*, & \text{ako } u \leq \rho(X_i, X_i^*) \\ X_i, & \text{inače.} \end{cases}$.

Krenimo sada sa provedbom Metropolis-Hastings algoritma. Dani otvoreni tekst je:

"DANAS JE LIJEP DAN PETAK JE PREZENTIRAT ĆU VAM SVOJ RAD I NADAM SE DA ĆE VAM SE SVIDJETI JAKO VOLIM MATEMATIKU I OD SAMOG POČETKA STUDIRANJA NISAM SE MOGLA ZAMISLITI NIGDJE DRUGDJE MATEMATIKA JE JAKO ZANIMLJIVA I PRIMJENJIVA ONA JE OSNOVA MNOGIM DRUGIM GRANAMA ZNANOSTI VJERUJEM DA MNOGI TO ZNAJU".

Logaritam vjerodostojnosti ovog teksta, odnosno suma logaritama prijelaznih vjerojatnosti svih bigrama koji se nalaze u njemu, iznosi -668.9767. Sada nasumično premutiramo slova naše abecede (kao što smo već rekli, to je abeceda bez slova "dž", "lj" i "nj") i dobivamo jedno mapiranje po kojemu šifriramo otvoreni tekst. Dobiveni šifrat glasi:

"RŠIŠJ ZT CKZTČ RŠI ČTĐŠH ZT ČUTGTIĐKUŠĐ ĆP MŠN JMFZ UŠR K IŠRŠN JT RŠ ĆT MŠN JT JMKRZTĐK ZŠHF MFCKN NŠĐTNŠĐKHP K FR JŠNFV ČFBTĐHŠ JĐPRKUŠIZŠ IKJŠN JT NFVCŠ GŠNKJCKĐK IKVRZT RUPVRZT NŠĐTNŠĐKHŠ ZT ZŠHF GŠIKNCZKMŠ K ČUKNZTIZKMŠ FIŠ ZT FJIFMŠ NIFVKN RUPVKN VUŠIŠNŠ GIŠIFJĐK MZTUPZTN RŠ NIFVK ĐF GIŠZP".

Dakle, svako je slovo zamijenjeno nekim drugim slovom. Kao početnu vrijednost u Markovljevom lancu uzimamo neko nasumično mapiranje i njime dešifriramo tekst. Početna je vrijednost:

"ČZDZS ŽK REŽKC ČZD CKGZT ŽK CHKJKDGEHZG FV AZU SALŽ HZČ E DZČZU SK ČZ FK AZU SK SAEČŽKGE ŽZTL ALREU UZGKUZGETV E LČ SZULP CLMKGTZ SGVČEHZDŽZ DESZU SK ULPRZ JZUESREGE DEPČŽK ČHVPČŽK

UZGKUZGETZ ŽK ŽZTL JZDEURŽEAZ E CHEUŽKDŽEAZ LDZ ŽK LSDLAZ
UDLPEU ČHVPEU PHZDZUZ JDZDLSGE AŽKHVŽKU ČZ UDLPE GL JDZŽV"

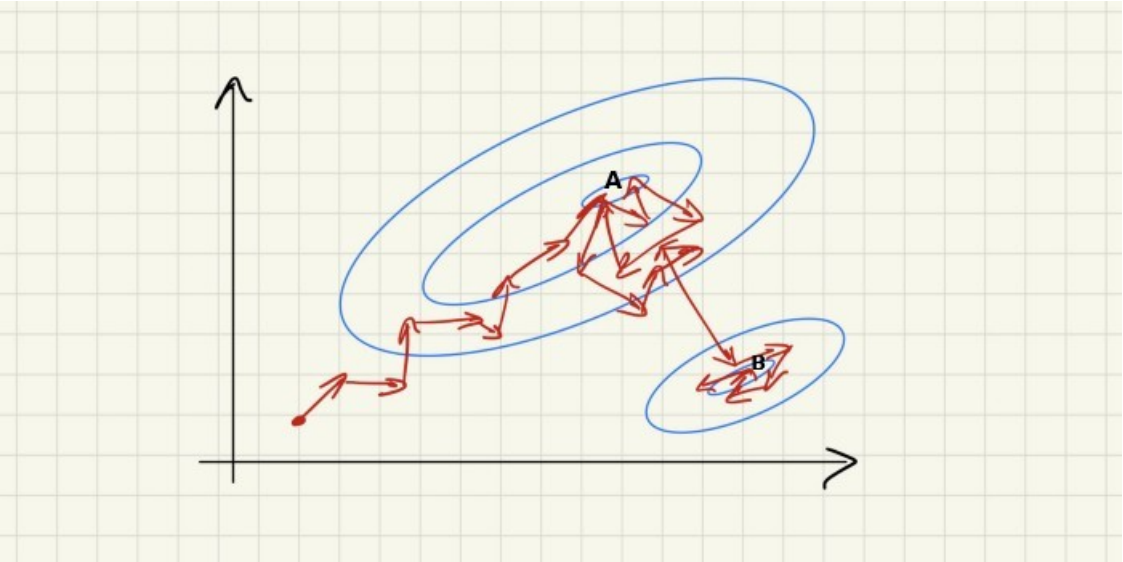
Zatim provodimo algoritam. U svakoj iteraciji algoritma na slučajan se način odabiru dva slova koja se u mapiranju zamjenjuju i time se dobiva novo mapiranje. Po njemu dešifriramo tekst i računamo vjerodostojnost dobivenog teksta. Tu vjerodostojnost podijelimo sa vjerodostojnošću teksta iz prethodne iteracije, odnosno oduzmemo logaritme vjerodostojnosti i razliku transformiramo eksponencijalnom funkcijom. Ukoliko je dobiveni kvocijent veći od jedan, pomičemo se u novo stanje, tj. prihvaćamo novo mapiranje i tekst dodajemo u lanac. U suprotnom, odabire se broj iz uniformne distribucije na intervalu $[0, 1]$. Ukoliko je odabrani broj manji od dobivenog kvocijenta, prihvaća se novo mapiranje sa vjerojatnošću koja je jednaka tom kvocijentu. Dakle, algoritam teži pronalasku onog rješenja za koje je vjerodostojnost najveća. Iteracije ponavljamo dok ne dođemo do rješenja ili približnog rješenja iz kojeg je moguće shvatiti kako glasi otvoreni tekst. Pogledajmo sada tijek 500 iteracija pri čemu je svaka 50. ispisana:

50 TAZAJ RU GNRUS TAZ SUOAŠ RU SVUBUZONVAO IL DAK JDMR VAT N ZATAK JU TA IU DAK JU JDNTRUON RAŠM DMGNK KAOUKAONŠL N MT JAKME SMPUOŠA JOLTNVAZRA ZNJAK JU KMEGA BAKNJGNON ZNETRU TVLETRU KAOUKAONŠA RU RAŠM BAZNKGRNDA N SVNKRUZRNDA MZA RU MJZMDA KZMENK TVLENK EVAZAKA BZAZMJON DRUVLRUK TA KZMEN OM BZARL
100 TADAZ NE GONES TAD SEBAR NE SKEŽEDBOKAB LI VAM ZVJN KAT O DATAM ZE TA LE VAM ZE ZVOTNEBO NARJ VJGOM MABEMABORI O JT ZAMJU SJĆEBRA ZBITOKADNA DOZAM ZE MJUGA ŽAMOZGOBO DOUTNE TKIUTNE MABEMABORA NE NARJ ŽADOMGNOVA O SKOMNEDNOVA JDA NE JZDJVA MDJUOM TKIUOM UKADAMA ŽDADJZBO VNEKINEM TA MDJUO BJ ŽDANI
150 JARAZ NE GONES JAR SEDAL NE SKEPERDOKAD ŠU VAM ZVIN KAJ O RAJAM ZE JA ŠE VAM ZE ZVOJNEDO NALI VIGOM MADEMADOLU O IJ ZAMIT SIĆEDLA ZDUJOKARNA ROZAM ZE MITGA PAMOZGODO ROTJNE JKUTJNE MADEMADOLA NE NALI PAROMGNOVA O SKOMNERNOVA IRA NE IZRIVA MRITOM JKUTOM TKARAMA PRARIZDO VNEKUNEM JA MRITO DI PRANU
200 NOROZ JE LAJES NOR SEDOB JE STEPERDATOD ĆU VOM ZVIJ TON A RONOM ZE NO ĆE VOM ZE ZVANJEDA JOBI VILAM MODEMODABU A IN ZOMIK SICEDBO ZDUNATORJO RAZOM ZE MIKLO POMAZLADA RAKNJE NTUKNJE MODEMODABO JE JOBI PORAMLJAVO A STAMJERJAVO IRO JE IZRIVO MRIKAM NTUKAM KTOROMO PRORIZDA VJETUJEM NO MRIKA DI PROJU
250 NOROZ JE LAJES NOR SEDOG JE STEPERDATOD BU VOM ZVIJ TON A RONOM ZE NO BE VOM ZE ZVANJEDA JOGI VILAM MODEMODAGU A IN ZOMIK SICEDGO ZDUNATORJO RAZOM ZE MIKLO POMAZLADA RAKNJE NTUKNJE MODEMODAGO JE JOGI PORAMLJAVO A STAMJERJAVO IRO JE IZRIVO MRIKAM NTUKAM KTOROMO PRORIZDA VJETUJEM NO MRIKA DI PROJU
300 NOROZ JE LAJES NOR SEDOG JE STEPERDATOD BU VOM ZVIJ TON A RONOM ZE NO BE VOM ZE ZVANJEDA JOGI VILAM MODEMODAGU A IN ZOMIK SIHEDGO ZDUNATORJO RAZOM ZE MIKLO POMAZLADA RAKNJE NTUKNJE MODEMODAGO JE JOGI PORAMLJAVO A STAMJERJAVO IRO JE IZRIVO MRIKAM NTUKAM KTOROMO PRORIZDA VJETUJEM NO MRIKA DI PROJU
350 SANAZ JE LOJEP SAN PEDAĆ JE PRETENDORAD BU VAM ZVIJ RAS O NASAM ZE SA BE VAM ZE ZVOSJEDO JAČI VILOM MADEMADOČU O IS ZAMIK PIĆEDČA ZDUSORANJA NOZAM ZE MIKLA TAMOZLODO NOKSJE SRUKSJE MADEMADOČA JE JAČI TANOMLJOVA O PROMJENJOVA INA JE IZNIVA MNIKOM SRUKOM KRANAMA TNANIZDO VJERUJEM SA MNIKO DI TNAJU

400 TANAZ JE LOJEP TAN PEDAB JE PRESENDORAD ČU VAM ZVIJ RAT O NATAM ZE TA ČE VAM ZE ZVOTJEDO JABI VILOM MADEMADOBU O IT ZAMIK PIHEDBA ZDUTORANJA NOZAM ZE MIKLA SAMOZLODO NOKTJE TRUKTJE MADEMADOBA JE JABI SANOMLJOVA O PROMJENJOVA INA JE IZNIVA MNIKOM TRUKOM KRANAMA SNANIZDO VJERUJEM TA MNIKO DI SNAJU
450 DANAS JE LOJEP DAN PETAK JE PREZENTORAT CU VAM SVIJ RAD O NADAM SE DA CE VAM SE SVODJETO JAKI VILOM MATEMATOKU O ID SAMIG PIČETKA STUDORANJA NOSAM SE MIGLA ZAMOSLOTO NOGDJE DRUGDJE MATEMATOKA JE JAKI ZANOMLJOVA O PROMJENJOVA INA JE ISNIVA MNIGOM DRUGOM GRANAMA ZNANISTO VJERUJEM DA MNIGO TI ZNAJU
500 DANAS JE LIJEP DAN PETAK JE PREZENTIRAT BU VAM SVOJ RAD I NADAM SE DA BE VAM SE SVIDJETI JAKO VOLIM MATEMATIKU I OD SAMOG POŽETKA STUDIRANJA NISAM SE MOGLA ZAMISLITI NIGDJE DRUGDJE MATEMATIKA JE JAKO ZANIMLJIVA I PRIMJENJIVA ONA JE OSNOVA MNOGIM DRUGIM GRANAMA ZNANOSTI VJERUJEM DA MNOGI TO ZNAJU

Tablica 5: Iteracije algoritma

Iako vidimo da nisu sva slova dobro dešifrirana, iz dobivenog se teksta može iščitati otvoreni tekst. Logaritam vjerodostojnosti dobivenog otvorenog teksta iznosi -668.1247, što je veće od logaritma vjerodostojnosti pravog otvorenog teksta i iz tog je razloga algoritam odabrao ovo kao najbolji mogući otvoreni tekst.



Slika 16: Ilustracija algoritma, preuzeto sa [11]

Na prvu bismo rekli kako je ovo primjer jednog pohlepnog algoritma. Pohlepni algoritmi mogu "zaglaviti" u lokalnim ekstremima i nisu optimalni. Dobra je stvar kod MCMC metode što ona ipak s malom vjerojatnošću prihvaća i ona stanja s manjom vjerodostojnošću.
Slika 16 prikazuje ilustraciju algoritma. Područja omeđena plavim elipsama područja su veće vjerojatnosti, pri čemu su slovima A i B označena područja sa najvećim vjerojatnostima. Vidimo kako algoritam kada dođe u područje A ostaje u njemu neko vrijeme, ali ipak može prijeći u B. Upravo to se ponekad događalo i s ovim algoritmom. Naime, događalo se da nakon izvršavanja algoritma, posljednjih 100 iteracija daju tekst koji nije smislen, ali je

njegov logaritam vjerodostojnosti bio blizu -668. Dakle, algoritam je "zaglavio" u području gdje je logaritam vjerodostojnosti velik, a to nije traženo rješenje, i potrebno je puno koraka da se "izvuče". U tim situacijama može se ili iterirati dalje, što je zahtjevno jer treba puno koraka do rješenja, ili ponovno pokrenuti algoritam ispočetka.
Dakle, ako algoritam "upadne" u neko područje čija stanja imaju veliku vjerodostojnost, a ne daju nam traženo rješenje, ipak postoji vjerojatnost da on "izađe" iz tog područja i pronađe rješenje. Ponekad će za to biti potreban veći broj koraka, ali konvergenciju prema rješenju jamči nam ergodski teorem.

Dodatci

Korištena R skripta za Primjer 2.1:

```
library(plotrix)
#na pocetku nacrtamo kvadrat i upisani krug:
plot(c(0, 1), c(0,1), type = "n", asp=1, xlab="", ylab="")
rect( 0, 0, 1, 1, col="powderblue")
draw.circle( 0.5, 0.5, .5, col="moccasin")
n<-100000 #broj tocaka
x<-runif(n,0,1)
y<-runif(n,0,1)
points(x,y, pch=46, col="red")
#Monte Carlo simulacija:
m<-1000 #broj iteracija simulacija
aproksimacija_pi<-numeric(m)
funkcija_g<-numeric(m)
for(j in 1:m)
{
  x<-runif(n,0,1)
  y<-runif(n,0,1)

  udaljenost<-numeric(n)
  for (i in c(1:n)) {
    udaljenost[i]<-sqrt((0.5-x[i])^2+(0.5-y[i])^2)   #euklidska
  }
  udaljenost
  funkcija_g[j]<-sum(udaljenost<=0.5)
  funkcija_g[j]/n   #aproksimacija povrsine kruga
  aproksimacija_pi[j]<-funkcija_g[j]/(n*0.25)   #aproksimacija  pi
}
mean(aproksimacija_pi)
var(aproksimacija_pi)
pi
#za tablicu
funkcija_g
aproksimacija_pi
#crtanje 3d histograma i 2d prikaza:
library(plot3D)
xcut<-cut(x, 100)
ycut<-cut(y,100)
hist3D(z=table(xcut,ycut), xlab="x-koordinata", ylab="y-koordinata",
       zlab="frekvencija", main="")
image2D(z=table(xcut,ycut), border="black", xlab="x-koordinata",
        ylab="y-koordinata")
```

Korištena R skripta za Primjer 4.1:

```
a=3; b=5  # pocetne vrijednosti
N=5000    #broj simulacija
X=rep(runif(1),N)  #inicijalizacija lanca
#X
prihvacanje<-0  #tu cemo upisati broj prihvacenih koraka
for (i in 2:N){
  Y=runif(1)                                    #q je uniformna
  rho=dbeta(Y,a,b)/dbeta(X[i-1],a,b)            #f je beta distribucija
  X[i]=X[i-1] + (Y-X[i-1])*(runif(1)<rho)   #odluka o prihvacanju
  if(X[i]!=X[i-1])
  {
    prihvacanje=prihvacanje+1       #prihvaceni koraci
  }
}
#X
prihvacanje
#efektivna velicina uzorka:
library(coda)
effectiveSize(X)
#graf:
plot(c(4500:5000), X[4500:5000], type = "l", xlab="broj simulacije",
     ylab="simulirana vrijednost")
for(i in c(4500:5000)){
  if(X[i]==X[i-1])
  {
    #zacrveni gdje se vrijednost ne mijenja:
    lines(c(i, i-1), c(X[i],X[i-1]),col="red", lwd=2)
  }
}
#histogram:
h<-hist(X, xlim=range(0,1), freq = FALSE, main="",
        xlab="vrijednosti lanca", ylab="gusto a", yaxt="n")
hmin<-min(h$density)
hmax<-max(h$density)
axis(2, seq(from=hmin, to=hmax,length.out = 6),
     round(seq(hmin/20,hmax/20,length.out =  6),2))
lines(density(X), col="green")
curve(dbeta(x,a,b), add=T, col="red")
legend("topright", legend=c("ciljna distribucija",
        "procjena ciljne distribucije"),
       col=c("red", "green"), lty=1)
#analiza:
jitter(X)-X        #jitter dodaje  um  u uzorak
ks.test(jitter(X),rbeta(5000,a,b))
mean(X)   #ocekivanje simuliranog uzorka
a/(a+b)   #ocekivanje beta distribucije
var(X)                  #varijanca simuliranog uzorka
a*b/((a+b)^2*(a+b+1))   #varijanca beta distribucije
```

Korištena R skripta za Primjer 4.2:

```
a=3; b=5  # pocetne vrijednosti
N=5000    #broj simulacija
#standardne devijacije(za svaku pokrecemo novu simulaciju):
stdev=0.05
stdev=0.5
stdev=1
stdev=1.5
X=rep(0.5,N)   #inicijalizacija lanca, krecemo iz 0.5
#X
prihvacanje=0  #tu cemo upisati broj prihvacenih koraka
for (i in 2:N){
  Y=rnorm(1,X[i-1],stdev)                    #q je normalna
  rho=dbeta(Y,a,b)/dbeta(X[i-1],a,b)         #f je beta distribucija
  X[i]=X[i-1] + (Y-X[i-1])*(runif(1)<rho)    #odluka o prihvacanju
  if(X[i]!=X[i-1])
  {
    prihvacanje=prihvacanje+1        #prihvaceni koraci
  }
  else{}
}
#X
prihvacanje
#efektivna velicina uzorka:
library(coda)
effectiveSize(X)
#graf:
plot(c(4500:5000), X[4500:5000], type = "l", xlab="broj simulacije",
     ylab="simulirana vrijednost", main=stdev)
for(i in c(4500:5000)){
  if(X[i]==X[i-1])
  {
    #zacrveni gdje se vrijednost ne mijenja:
    lines(c(i, i-1), c(X[i],X[i-1]),col="red", lwd=2)
  }
}
#histogram:
h<-hist(X, xlim=range(0,1), freq = FALSE, main="",
        xlab="vrijednosti lanca", ylab="gusto a", yaxt="n",
        breaks = 50)
hmin<-min(h$density)
hmax<-max(h$density)
axis(2, seq(from=hmin, to=hmax,length.out = 6),
     round(seq(hmin/20,hmax/20,length.out =  6),2))
lines(density(X), col="green")
curve(dbeta(x,a,b), add=T, col="red")
legend("topright", legend=c("ciljna distribucija",
                            "procjena ciljne distribucije"),
       col=c("red", "green"), lty=1)
title(stdev)
#analiza:
```

```
mean(X)   #ocekivanje simuliranog uzorka
a/(a+b)   #ocekivanje beta distribucije
var(X)                  #varijanca simuliranog uzorka
a*b/((a+b)^2*(a+b+1))   #varijanca beta distribucije
```

Korištena R skripta za primjer dekriptiranja poruke:

```
getwd()
setwd("C:/Users/Marija␣Kristina/Desktop/DIPLOMSKI␣RAD")
tekst=readLines("zlocinikazna.txt")  #ucitavanje referentnog teksta
tekst=toupper(tekst)
library("stringr")
#prociscavanje referentnog teksta:
tekst<-str_replace_all(tekst, "-␣", "")
tekst<-str_replace_all(tekst, "  ␣", "")
tekst<-str_replace_all(tekst, ",", "")
str_count(tekst, ",")
str_count(tekst, "A")
tekst<-str_replace_all(tekst, fixed("."), "")
str_count(tekst, fixed("."))
tekst<-str_replace_all(tekst, fixed("("), "")
str_count(tekst, fixed("("))
tekst<-str_replace_all(tekst, fixed(")"), "")
str_count(tekst, fixed(")"))
tekst[1]
str_count(tekst, "-\t")
tekst<-str_replace_all(tekst, "-\t", "")
str_count(tekst, "-\t")
#izgradnja matrice prijelaznih vjerojatnosti:
slova<-c("A", "B", "C", "  ", "  ", "D", "  ", "E", "F", "G", "H",
         "I", "J", "K", "L", "M", "N", "O", "P", "R", "S", "  ",
         "T", "U", "V", "Z", "  ")
length(slova)
k<-28 #broj slova u abecedi +1 (ova jedinica je zbog razmaka)
matrica_prijelaza=matrix(0,k,k)
rownames(matrica_prijelaza)=colnames(matrica_prijelaza)=
  c(toupper(slova), "␣")
zadnje=""
substring(tekst[1],0,0)
for (linija in 1:length(tekst)) {
  if (linija %% 100 ==0) {cat("Redak",linija,"\n")}
  for (pos in 1:nchar(tekst[linija])) {
    trenutno=substring(tekst[linija],pos,pos)
    zadnje=substring(tekst[linija],pos-1,pos-1)
    if (trenutno %in% toupper(slova)) {
      matrica_prijelaza[rownames(matrica_prijelaza)==zadnje,
                        colnames(matrica_prijelaza)==trenutno]=
        matrica_prijelaza[rownames(matrica_prijelaza)==zadnje,
                          colnames(matrica_prijelaza)==trenutno]+1
    }
    else {
      if (trenutno=="␣") {
        matrica_prijelaza[rownames(matrica_prijelaza)==zadnje,k]=
          matrica_prijelaza[rownames(matrica_prijelaza)==zadnje,k]+1
        zadnje="␣"
      }
    }
```

```
  }
}
matrica_prijelaza
#analiza za najfrekventniji bigram:
max(matrica_prijelaza[1:27, 1:27])
matrica_prijelaza[1:27, 1:27]==max(matrica_prijelaza[1:27, 1:27])
#JE je najfrekventiniji bigram
which(matrica_prijelaza==max(matrica_prijelaza))
matrica_prij_vj=sweep(matrica_prijelaza+1,1,
                      rowSums(matrica_prijelaza+1),FUN="/")
max(matrica_prij_vj)
which(matrica_prij_vj==max(matrica_prij_vj))
matrica_prij_vj[209]
matrica_prij_vj==max(matrica_prij_vj)
colnames(matrica_prij_vj)[matrica_prij_vj==max(matrica_prij_vj)]
rownames(matrica_prij_vj)[matrica_prij_vj==max(matrica_prij_vj)]
matrica_prij_vj[1:27, 27]
which(matrica_prij_vj[1:27, 27]==max(matrica_prij_vj[1:27, 27]))
#-----------------------------------------------------------------
#shematski prikaz matrice prijelaznih vjerojatnosti:
library(ggplot2)
library(reshape2)
ggplot(melt(matrica_prij_vj),aes(Var2,Var1))+geom_tile(aes(fill=value))
  +scale_fill_gradient(low="white",high="black",limits=c(0,1))+
  labs(x="Uvjetna␣vjerojatnost␣drugog␣slova",
       y="Uvjetno␣na␣prvo␣slovo",fill="Vjerojatnost")+
  scale_y_discrete(limits = rev(levels(melt(matrica_prij_vj)$Var1)))+
  coord_equal()
#-----------------------------------------------------------------
#funkcija za sifriranje i desifriranje kojoj predajemo kljuc i sifrat:
desifriranje <- function(kljuc,sifrat) {
  sifrat=toupper(sifrat)
  otvoreni=sifrat
  for (i in 1:nchar(sifrat)) {
    if (substring(sifrat,i,i) %in% toupper(slova)) {
      substring(otvoreni,i,i)=toupper(slova[kljuc==
                                              substring(sifrat,i,i)])
    }
  }
  otvoreni
}
#funkcija koja racuna logaritam vjerodostojnosti:
log.vj <- function(otvoreni) {
  otvoreni=toupper(otvoreni)
  logvj=0
  for (i in 2:nchar(otvoreni)) {
    trenutno=substring(otvoreni,i,i)
    zadnje=substring(otvoreni,i-1,i-1)
    if (trenutno %in% toupper(slova)) {
      logvj=logvj+log(matrica_prij_vj[rownames(matrica_prijelaza)==
                                        zadnje,
```

```
                                        colnames(matrica_prijelaza)==
                                          trenutno])
    } else {
      if (trenutno==" ") {
        logvj=logvj+
          log(matrica_prij_vj[rownames(matrica_prijelaza)==zadnje,k])
      }
    }
  }
  logvj
}
#-------------------------------------------------------------
mojtekst="danas je lijep dan petak je prezentirat u vam svoj rad i
nadam se da e vam se svidjeti jako volim matematiku i od samog
po etka studiranja nisam se mogla zamisliti nigdje drugdje
matematika je jako zanimljiva i primjenjiva ona je osnova mnogim
drugim granama znanosti vjerujem da mnogi to znaju"
mojtekst=toupper(mojtekst)
(vjerod<-log.vj(mojtekst))
# ifrira  otvoreni tekst:
(sifrat=desifriranje(sample(toupper(slova)),mojtekst))

#pocetne vrijednosti:
kljuc=sample(toupper(slova))
(cur.decode=desifriranje(kljuc,sifrat))
(cur.logvj=log.vj(cur.decode))
max.logvj=cur.logvj
max.decode=cur.decode
#algoritam:
i=1
n=500
vjerodostojnost<-numeric(500)
while (i<=n) {
  proposal=sample(1:(k-1),2) #biram 2 slova za zamjenu
  prop.kljuc=kljuc
  #mijenjamo dva slova u kljuu:
  prop.kljuc[proposal[1]]=kljuc[proposal[2]]
  prop.kljuc[proposal[2]]=kljuc[proposal[1]]

  prop.decode=desifriranje(prop.kljuc,sifrat)
  prop.logvj=log.vj(prop.decode)

  if (runif(1)<exp(prop.logvj-cur.logvj))  #odluka o prihva anju
  {kljuc=prop.kljuc
  cur.decode=prop.decode
  cur.logvj=prop.logvj
  if(cur.logvj>max.logvj) {
    max.logvj=cur.logvj
    max.decode=cur.decode
  }
  if ((i %%  50) == 0){    #printa svaku 50-tu iteraciju
```

```
    cat(i,cur.decode,"\n")
  }
  vjerodostojnost[i]<-cur.logvj
  i=i+1
  }
}
```

Literatura

[1] *Introducing Monte Carlo Methods with R*, Christian P. Robert i George Casella, Springer, 2010

[2] *Markovljevi lanci*, prof.dr.sc. Vondraček, Zoran, 2012./2013.

[3] *Markov chains*, Pierre Bremaud, Springer, 1999.

[4] *MCMC i primjene*, Bartoš, Elio , Diplomski rad, Sveučilište u Zagrebu, Prirodoslovno-matematički fakultet, Matematički odsjek, 2016.

[5] *Monte Carlo simulacije*, Kovačević, Alen , Diplomski rad, Sveučilište J.J.Strossmayera u Osijeku, Odjel za matematiku, 2015.

[6] *On-line Monte Carlo simulacija*, Ivanković, Margita , Završni rad, Sveučilište u Zagrebu, Fakultet strojarstva i brodogradnje, 2017.

[7] *Bayesian Data Analysis*, Gelman, Carlin, Stern, Dunson, Vehtari, Rubin, elektroničko izdanje, 2020.

[8] *Intro to Markov Chain Monte Carlo*, Rebecca C. Steorts, <http://www2.stat.duke.edu/ rcs46/lecturesModernBayes/601-module6-markov/markov-chain-monte-carlo.pdf>

[9] *A Zero-Math Introduction to Markov Chain Monte Carlo Methods*, Ben Shaver, 2017, <https://towardsdatascience.com/a-zero-math-introduction-to-markov-chain-monte-carlo-methods-dcba889e0c50>

[10] *Text decryption using MCMC*, R-bloggers, 2013, <https://www.r-bloggers.com/2013/01/text-decryption-using-mcmc/>

[11] *Introduction to MCMC*, Alexander Bailey, towards data science, 2020, <https://towardsdatascience.com/introduction-to-mcmc-1c8e3ea88cc9>

[12] *Markov chain*, Wikipedia, <https://en.wikipedia.org/wiki/Markov$_chain$ >

Monte Carlo Markov chains

Sažetak

Monte Carlo simulacije naziv su za numeričku metodu rješavanja matematičkih problema na način da se generira velik broj realizacija slučajnih varijabli. Primjenu su pronašle u problemima koji se mogu svesti na aproksimiranje integrala. Metoda Monte Carlo Markovljevih lanca postupak je kojim se simuliraju uzorci iz zadane distribucije. Pri tome se konstruira ergodski Markovljev lanac tako što se u svakoj iteraciji MCMC algoritma odabire neki parametar i ukoliko je on "bolji" od prethodnog dodaje se u lanac. U ovome radu predstavljena je klasa Metropolis-Hastings algoritama te su pobliže opisani i oprimjereni nezavisni Metropolis-Hastings algoritam i Metropolis-Hastings algoritam sa slučajnom šetnjom. Na kraju rada dana je primjena MCMC algoritama na problem dešifriranja poruke šifrirane supstitucijskom šifrom.

Ključne riječi: Monte carlo, Markovljevi lanci, algoritam Metropolis-Hastings algoritam, slučajni brojevi, supstitucijska šifra, dešifriranje

Abstract

Monte Carlo simulations are numerical methods for solving mathematical problems in a way that generates a large number of realization random variables. They have found application in problems that can be reduced to approximating integrals. Monte Carlo Markov chains method is a procedure that simulates samples from default distribution. In doing so, an ergodic Markov chain is constructed by selecting a parameter in each iteration of the MCMC algorithm and adding it to the chain if it is "better" than the previous one. In this paper, a class of Metropolis-Hastings algorithms is presented and the independent Metropolis-Hastings algorithm and the Metropolis-Hastings algorithm with random walk are described and exemplified in more detail. At the end of the paper, the application of MCMC algorithms to the problem of decrypting a message encrypted with a substitution code is given.

Key words: Monte carlo,Markov chains, algoritam, Metropolis-Hastings algoritam, stohastic numbers, substitution code, decoding

Životopis

Rođena sam 13.6.1997. u Osijeku gdje sam pohađala osnovnu školu, a zatim Opću gimnaziju. Tijekom osnovnoškolskog i srednjoškolskog obrazovanja sudjelovala sam na natjecanjima iz matematike, njemačkog i kemije. U srednjoj sam školi položila Deutsches Sprachdiplom (DSD) certifikat za njemački jezik za razinu B1. Imala sam odličan prosjek ocjena i kao maturantica bila sam voditeljica školske volonterske udruge. U listopadu 2015. godine upisujem Sveučilišni preddiplomski studij matematike na Odjelu za matematiku u Osijeku kojeg završavam 2018. Time stječem akademsku titulu Sveučilišne prevostupnice (baccalauree) matematike. U listopadu iste godine upisujem se na Sveučilišni diplomski studij matematike, smjera financijska matematika i statistika, također na Odjelu za matematiku u Osijeku. Tijekom diplomskog sam studija odradila stručne prakse u poduzećima Osijek-Koteks d.d. u Osijeku i u Hrvatskoj agenciji za nadzor financijskih usluga (HANFA) u Zagrebu.