Monte Carlo integracija korištenjem različitih generatora slučajnih brojeva u programskom jeziku R

Knežević, Stela

Sveučilište Josipa Jurja Strossmayera u Osijeku

Fakultet primijenjene matematike i informatike

Sveučilišni prijediplomski studij Matematika i računarstvo

Monte Carlo integracija korištenjem različitih generatora slučajnih brojeva u programskom jeziku R

Završni rad

Mentor:

doc. dr. sc. Ivan Papić

Student:

Stela Knežević

Osijek, 2024.

Sadržaj

Uvod

Monte Carlo metode predstavljaju snažan alat za rješavanje složenih problema područjima kao što su matematika, statistika, fizika, financije te u brojnim drugim znanstvenim disciplinama. Osnovna ideja ovih metoda leži u korištenju slučajnog uzorka za procjenu matematičkih veličina koje je teško ili nemoguće izračunati analitički. Jedna od ključnih primjena Monte Carlo metoda je procjena integrala, posebno u višedimenzionalnim slučajevima gdje klasične numeričke metode gube na učinkovitosti.

Monte Carlo integracija koristi nasumično generirane točke unutar područja integracije kako bi aproksimirala vrijednost integrala funkcije. Glavna prednost ove metode je njena fleksibilnost, a isto tako i sposobnost rješavanja integrala u situacijama kada su granice beskonačne ili funkcija ima složen oblik. Korištenjem različitih tehnika uzorkovanja i generatora slučajnih brojeva moguće je značajno poboljšati točnost i stabilnost procjene.

U ovom radu analizirat će se i usporediti različite metode Monte Carlo integracije, uključujući klasičnu, stratificiranu, antitetičku i *hit − or − miss* integraciju, kao i sofisticiranije pristupe poput *importance sampling* metode i *random walk* integracije. Također, bit će obuhvaćeni različiti generatori slučajnih brojeva koji se koriste za generiranje uzoraka, kao što su linearni kongruencijalni generator (LCG), *Mersenne twister*, Haltonov sekvencijalni generator i drugi.

Cilj ovog rada je prikazati primjenu Monte Carlo integracije korištenjem različitih generatora slučajnih brojeva kroz praktičnu implementaciju u programskom jeziku R, uz korištenje Shiny sučelja za interaktivnu analizu i usporedbu metoda. Rad će naglasiti prednosti i nedostatke svake metode te pokazati kako izbor generatora slučajnih brojeva može utjecati na konačne rezultate integracije.

1 | Monte Carlo metode

1.1 Osnovni principi Monte Carlo metoda

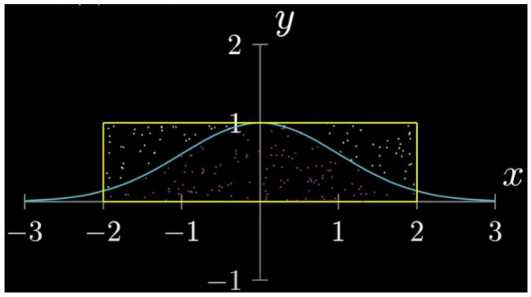
Monte Carlo metode predstavljaju skup numeričkih algoritama koji koriste slučajne uzorke za rješavanje problema koji su često deterministički po svojoj prirodi. Njihova primjena započela je u sredinom 20. stoljeća, prvenstveno u području fizike, ali su se ubrzo proširile i na druga područja poput statistike, financija i računalne biologije. Ključna značajka Monte Carlo metoda je korištenje slučajnih ili pseudo-slučajnih brojeva za simulaciju i analizu složenih sustava.

U srži Monte Carlo metoda nalazi se ideja da se ponavljanjem procesa uzorkovanja i računanjem prosjeka tih uzoraka može procijeniti željena veličina. Zakon velikih brojeva garantira da će, uz dovoljan broj generiranih (pseudo)slučajnih brojeva, aproksimacija konvergirati prema pravoj vrijednosti, čineći Monte Carlo metode pouzdanim alatom za numeričke procjene.

1.2 Primjena Monte Carlo metoda u integraciji

Integracija je proces izračunavanja površine ispod krivulje funkcije i jedan je od temeljnih problema u matematici. Dok je izračunavanje integrala u jednoj dimenziji relativno jednostavno uz klasične metode poput trapeznog ili Simpsonovog pravila, višedimenzionalni integrali često predstavljaju znatno veći izazov. Monte Carlo integracija nudi pristup koji može zaobići probleme tradicionalnih metoda korištenjem slučajnosti.

Monte Carlo metode aproksimiraju integral generiranjem slučajnog uzorka unutar definiranog intervala integracije i korištenjem tako generiranog uzorka za aproksimaciju prosječne vrijednosti integrala. Ovaj pristup omogućava procjenu integrala čak i u slučajevima kada su granice beskonačne ili funkcija ima složen oblik, što je teško ili nemoguće postići tradicionalnim numeričkim metodama.

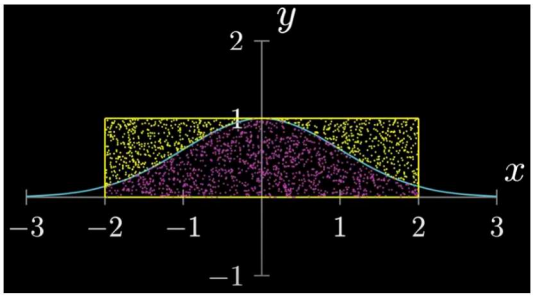


Slika 1.1: Monte Carlo metoda na primjeru aproksimacije vrijednosti funkcije $f(x) = e^{-\frac{x}{2}}$ sa $n = 1000$ točaka (Smith E., (*ResearchGate*))

Na Slici 1.1 vizualno je dočarano kako funkcioniraju Monte Carlo metode. Kao primjer, računali smo područje integracije funkcije

$$f(x) = e^{-\frac{x}{2}}$$

na intervalu [-2, 2].



Slika 1.2: Monte Carlo netoda na primjeru aproksimacije funkcije $f(x) = e^{-\frac{x}{2}}$ sa $n = 2000$ točaka (Smith E., (*ResearchGate*))

Na Slici 1.2. prikazano je poboljšanje u aproksimaciji povećanjem slučajnih točaka n. Rezultat se povećanjem broja n sve više približava stvarnoj vrijednosti zadanog integrala.

2 | Osnovni pojmovi iz vjerojatnosti i statistike

Za razumijevanje Monte Carlo integracije i ulogu generatora slučajnih brojeva, potrebno je poznavati ključne pojmove iz vjerojatnosti i statistike. Ovi pojmovi čine temelj teorije koja se koristi za procjenu matematičkih veličina putem slučajnih uzoraka.

Ovo poglavlje obuhvaća osnovne pojmove iz vjerojatnosti i statistike koji su temeljni za razumijevanje Monte Carlo metoda i njihovih primjena u procjeni integrala.

2.1 Slučajna varijabla

Slučajna varijabla je funkcija koja svakom ishodu slučajnog eksperimenta pridružuje brojčanu vrijednost. Može biti diskretna (ako poprima samo diskretne vrijednosti, poput bacanja kocke) ili neprekidna (ako može poprimiti bilo koju vrijednost iz nekog intervala, poput visine ljudi). U Monte Carlo metodama, slučajne varijable generiraju se pomoću generatora slučajnih brojeva, što omogućava procjenu očekivanih vrijednosti integrala.

Diskretna slučajna varijabla : Na primjer, bacanje kockice koja se koristi u društve definira slučajnu varijablu X koja može uzeti vrijednosti od 1 do 6, pri čemu svaka vrijednost ima jednaku vjerojatnost $1/6$.

Neprekidna slučajna varijabla : Za neprekidne varijable, poput vremena do kvara stroja, koristimo gustoću vjerojatnosti da bismo opisali distribuciju vjerojatnosti unutar određenog intervala. Klasičan primjer neprekidne varijable je normalna varijabla, koja ima kontinuiranu distribuciju i koristi se u brojnim znanstvenim disciplinama.

Matematički, za svaku slučajnu varijablu X definiramo funkciju distribucije vjerojatnosti (CDF),

$$F(X) = P(X \leq x),$$

koja daje vjerojatnost da će X uzeti vrijednost manju ili jednaku od x.

2.2 Gustoća vjerojatnosti

Za neprekidne slučajne varijable, koncept vjerojatnosti opisan je funkcijom gustoće vjerojatnosti (PDF – Probability Density Function). Gustoća vjerojatnosti $f(x)$ nije sama po sebi vjerojatnost, već funkcija koja, kada se integrira preko određenog intervala, daje vjerojatnost da će slučajna varijabla pasti unutar tog intervala. Matematički, vjerojatnost da varijabla X uzme vrijednost unutar intervala $[a, b]$ definira se kao:

$$P(a \leq X \leq b) = \int_a^b f(x)\,dx,$$

gdje je $f(x)$ funkcija gustoće. Funkcija gustoće uvijek zadovoljava uvjet :

$$\int_{-\infty}^{\infty} f(x)\,dx = 1,$$

što osigurava da se ukupna vjerojatnost svih mogućih ishoda mora zbrojiti do 1. Funkcija gustoće često se koristi u Monte Carlo metodama kada se uzorci generiraju iz specifičnih distribucija, kao što su normalna, eksponencijalna ili uniformna distribucija.

2.3 Matematičko očekivanje

Očekivanje ili matematičko očekivanje (eng. expectation) slučajne varijable X, označeno s $E(X)$, opisuje dugoročni prosjek svih mogućih vrijednosti koje varijabla može uzeti, ponderirano njihovim vjerojatnostima. Drugim riječima, očekivanje predstavlja središnju tendenciju distribucije varijabli i vrlo je važno u procjenama Monte Carlo metoda.

Za diskretne varijable X, očekivanje se računa kao suma svih mogućih vrijednosti x_i pomnoženih s njihovim vjerojatnostima p_i :

$$E(X) = \sum_i x_i p_i.$$

Za neprekidne varijable, očekivanje se definira putem integrala :

$$E(X) = \int_{-\infty}^{\infty} x f(x)\,dx,$$

gdje je $f(x)$ funkcija gustoće vjerojatnosti. U Monte Carlo metodama očekivanje igra ključnu ulogu jer se vrijednost integrala procjenjuje kao prosječna vrijednost funkcije na slučajno odabranim uzorcima. Primjerice, ako želimo aproksimirati integral funkcije $f(x)$ na intervalu $[a, b]$, matematičko očekivanje pruža osnovu za računski proces.

2.4 Varijanca

Varijanca je mjera raspršenosti vrijednosti slučajne varijable oko njenog očekivanja. Ona kvantificira koliko su moguće vrijednosti varijable udaljene od očekivane vrijednosti. Što je varijanca veća, to su veće razlike između očekivane vrijednosti i stvarnih ishoda. Za slučajnu varijablu X, varijanca je definirana kao:

$$\mathrm{Var}(X) = E\left((X - E(X))^2\right),$$

odnosno očekivano kvadratno odstupanje slučajne varijable od njenog očekivanja. Varijanca je važna u Monte Carlo metodama jer procjene s manjom varijancom obično rezultiraju preciznijim aproksimacijama integrala. U kontekstu Monte Carlo simulacija, smanjenje varijance uzoraka jedan je od glavnih ciljeva, a to se često postiže tehnikama poput stratificiranog uzorkovanja ili važnosnog uzorkovanja (*importance sampling*).

2.5 Uzorak i slučajni uzorak

Uzorak je skup podataka dobiven iz neke populacije, dok je slučajni uzorak specifičan tip uzorka gdje su podaci odabrani nasumično, kako bi svaki član populacije imao jednaku šansu biti odabran. U kontekstu Monte Carlo metoda, slučajni uzorak odnosi se na niz slučajnih točaka generiranih unutar intervala integracije pomoću generatora slučajnih brojeva.

Kvaliteta slučajnog uzorka od ključne je važnosti za preciznost Monte Carlo procjena. Na primjer, uniformni generator će ravnomjerno raspodijeliti uzorke po cijelom intervalu, dok će generator s nekom drugom distribucijom uzrokovati veća odstupanja u procjenama integrala. Tehnike poput stratificiranog uzorkovanja omogućuju bolje pokrivanje prostora integracije, smanjujući varijancu rezultata.

3 | Monte Carlo integracija

Monte Carlo integracija je posebna primjena Monte Carlo metoda za aproksimaciju određenih integrala, posebno u višedimenzionalnim slučajevima gdje tradicionalne numeričke metode postaju neefikasne ili nepraktične. Temeljna ideja Monte Carlo integracije je da se koristi prosječna vrijednost funkcije evaluirana na slučajno odabranim točkama unutar intervala integracije kako bi se dobila aproksimacija integrala.

3.1 Osnovni princip Monte Carlo integracije

Pretpostavimo da želimo izračunati određeni integral funkcije $f(x)$ na intervalu $[a, b]$:

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

Monte Carlo metoda za aproksimaciju ovog integrala temelji se na generiranju velikog broja realizacija iz uniformne distribucije nad intervalom $[a, b]$. Osnovna ideja je generirati n slučajnih točaka x_i koje su ravnomjerno raspodijeljene unutar intervala $[a, b]$. Svaka točka x_i je nasumično odabrana prema uniformnoj distribuciji, što znači da svaka vrijednost unutar intervala ima jednaku vjerojatnost da bude izabrana.

Aproksimacija integrala pomoću Monte Carlo metode tada se može zapisati kao:

$$I \approx \frac{b-a}{n} \sum_{i=1}^{n} f(x_i).$$

Ovaj izraz predstavlja prosjek funkcijskih vrijednosti x_i ponderiran s duljinom intervala $[a, b]$. Ključni dio ove metode je uniformno uzorkovanje, jer omogućava nepristrano procjenjivanje integrala bez potrebe za detaljnim poznavanjem oblika funkcije $f(x)$. Ovaj pristup se može proširiti i poboljšati na različite načine, uključujući korištenje različitih tehnika uzorkovanja koje mogu smanjiti varijancu i povećati točnost procjene.

3.2 Monte Carlo integracija za proizvoljnu funkciju gustoće

Monte Carlo metode mogu se proširiti na slučajeve kada funkcija gustoće nije uniformna, već proizvoljna. Ako želimo aproksimirati integral funkcije $g(x)$ preko intervala $[a, b]$, možemo koristiti sljedeći pristup:

Na početku generiramo realizaciju $X_1, X_2, \ldots, X_n$, gdje je $(X_1, X_2, \ldots, X_n)$ jednostavni slučajni uzorak iz distribucije $X \sim p(x)$, a $(x_1, x_2, \ldots, x_n)$ predstavlja pripadni uzorak, odnosno nezavisne realizacije dobivene iz te distribucije. Zatim, uporabom važnosnog uzorkovanja (importance sampling), aproksimiramo integral:

$$\int_a^b g(x)\,dx = \int_a^b \frac{g(x)}{p(x)} p(x)\,dx \approx \frac{1}{n}\sum_{i=1}^{n} \frac{g(x_i)}{p(x_i)},$$

gdje x_i označuje realizaciju slučajnog uzorka generiranu prema gustoći $p(x)$. Važno je naglasiti da izbor funkcije gustoće $p(x)$ mora odgovarati granicama integracije, tj. nosač funkcije gustoće $p(x)$ mora biti upravo interval $[a, b]$. Drugim riječima, funkcija gustoće mora biti jednaka nuli izvan granica integracije, što osigurava da se generirani slučajni brojevi x_i mogu realizirati samo unutar intervala $[a, b]$.

Ovaj pristup omogućuje fokusiranje uzorkovanja na regije gdje je doprinos funkcije $g(x)$ najveći, čime se može značajno poboljšati efikasnost metode. Ispravan izbor funkcije gustoće $p(x)$ ključno je za smanjenje varijance procjene i povećanje točnosti aproksimacije.

3.3 Jaki zakon velikih brojeva

Jaki zakon velikih brojeva (JZVB) je ključni teorem u teoriji vjerojatnosti koji opisuje konvergenciju aritmetičke sredine nezavisnih i identično distribuiranih slučajnih varijabli prema njihovom očekivanju. Formalno, teorem glasi:

TEOREM (Jaki zakon velikih brojeva) : Neka su $X_1, X_2, \ldots, X_n$ nezavisne i identično distribuirane slučajne varijable s konačnim očekivanjem $E[X]$. Tada, za svako $\epsilon > 0$:

$$P\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - E[X]\right| \geq \epsilon\right) \to 0 \quad \text{kada} \quad n \to \infty..$$

Ovaj teorem implicira da će aritmetička sredina uzoraka konvergirati prema očekivanoj vrijednosti kako broj uzoraka n postaje sve veći.

Dokaz : Dokaz ovog teorema može se naći u standardnim udžbenicima teorije vjerojatnosti, kao što su [4] ili [11].

Monte Carlo aproksimacija temelji se na jakom zakonu velikih brojeva. Njena formulacija omogućuje procjenu očekivanja funkcije $g(X)$ pomoću Monte Carlo

metoda, gdje se jaki zakon velikih brojeva koristi za konvergenciju aritmetičke sredine uzoraka prema očekivanoj vrijednosti.

Neka je $X_1, X_2, \ldots, X_n$ jednostavni slučajni uzorak (j.s.u.) iz distribucije X $\sim$p(x) , a $(x_1, x_2, \ldots, x_n)$ su pripadne nezavisne realizacije iz dane distribucije. Definirajmo slučajne varijable

$$Yi = \frac{g(X_i)}{p(X_i)}$$

, gdje je $X_i \sim p(x)$ gustoća prema kojoj se uzorak generira.

Tada je :

$$\mathbb{E}[Y_i] = \mathbb{E}\left[\frac{g(X)}{p(X)}\right] = \int_a^b \frac{g(x)}{p(x)} p(x)\, dx = \int_a^b g(x)\, dx,$$

gdje su $Y_1, Y_2, \ldots, Y_n$ nezavisne i jednako distribuirane slučajne varijable.

Jaki zakon velikih brojeva kaže da aritmetička sredina ovih varijabli gotovo sigurno (g.s.) konvergira prema njihovom očekivanju kad broj uzoraka n ide u beskonačnost:

$$\frac{1}{n}\sum_{i=1}^{n} Y_i = \frac{1}{n}\sum_{i=1}^{n} \frac{g(X_i)}{p(X_i)} \to \mathbb{E}\left[\frac{g(X)}{p(X)}\right] = \int_a^b g(x)\, dx \quad \text{g.s. kad} \quad n \to \infty.$$

U kontekstu Monte Carlo integracije, ovo znači da prosjek

$$\frac{1}{n}\sum_{i=1}^{n} \frac{p(X_i)}{g(X_i)}$$

konvergira prema stvarnoj vrijednosti integrala funkcije $g(x)$ kada se dimenzija uzorka n povećava. Ovaj rezultat pokazuje da se Monte Carlo metoda može pouzdano koristiti za aproksimaciju integrala, a točnost procjene raste s povećanjem dimenzije uzorka, što je osnova za primjenu ove metode u praksi.

3.4 Specijalni slučaj : Uniformna distribucija

U poglavlju 2.2 opisuje se Monte Carlo integracija s proizvoljnom funkcijom gustoće $p(x)$. Kada se funkcija gustoće odabere tako da bude uniformna, situacija postaje specijalni slučaj opisanih metoda. Uniformna distribucija može se definirati unutar intervala $[a, b]$ kao:

$$p(x) = \begin{cases} \frac{1}{b-a} & \text{, ako } x \in [a, b] \\ 0 & \text{, inače} \end{cases}$$

Uzimanje uzoraka iz uniformne distribucije Kada biramo $p(x)$ kao uniformnu distribuciju, realizacija uzorka $X_1, X_2, \ldots, X_n$ generira se jednostavno. Biraju se uniformni slučajni brojevi unutar intervala $[a, b]$. Ovaj izbor pomaže u pojednostavljenju integracije jer je gustoća konstantna unutar intervala.

Monte Carlo integracija s uniformnom gustoćom

Kada koristimo uniformnu funkciju gustoće $p(x)$ u okviru formule iz poglavlja 2.2, integral

$$\int_a^b g(x)\,dx$$

aproksimira se na sljedeći način :

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

gdje je x_i nasumičan uzorak generiran prema

$$p(x) = \frac{1}{b-a}.$$

Primjena jakog zakona velikih brojeva S obzirom na to da su

$$Y_i = g(X_i)\cdot(b-a)$$

(jer je $p(x)$ konstantno), možemo sada primijeniti jaki zakon velikih brojeva :

$$\frac{1}{n}\sum_{i=1}^{n} Y_i = \frac{1}{n}\sum_{i=1}^{n} g(X_i)\cdot(b-a) \to E[g(X)](b-a) \quad \text{kada } n\to\infty = \int_a^b g(x)\,dx.$$

Zatvaranje Kako bi se povezali rezultati, možemo reći :

1. Kada je $p(x)$ uniformna, Monte Carlo integracija se pojednostavljuje, a varijanca procjene može biti niža.

2. Jaki zakon velikih brojeva jamči da će aritmetička sredina

$$\frac{1}{n}\sum_{i=1}^{n} g(X_i)$$

konvergirati prema stvarnom integralu

$$\int_a^b g(X)\,dx$$

kada se dimenzija uzorka povećava.

Ova veza između uniformne gustoće i općih principa Monte Carlo integracije ukazuje na to da je uniformna distribucija posebni, ali važan slučaj koji omogućuje jednostavnije i efikasnije aproksimacije integrala.

4 | Implementirane Monte Carlo metode

4.1 Klasična Monte Carlo integracija

Klasična Monte Carlo integracija je najosnovnija primjena Monte Carlo metoda za aproksimaciju određenih integrala. Ova metoda koristi uniformnu distribuciju za generiranje nasumičnih točaka unutar zadanog intervala integracije. Osnovna ideja je da se prosječna vrijednost funkcije na slučajno odabranim točkama koristi za procjenu vrijednosti integrala.

Matematička formulacija ove metode opisana je u poglavlju 3.1. gdje se opisuje osnovni princip Monte Carlo integracije.

```
MonteCarloIntegracija <- function(f, a, b, n, generator, seed) {
  if (is.infinite(a) || is.infinite(b)) {
    # Za beskonačne intervale koristimo normalnu distribuciju
    u <- generator(seed, n)
    x <- qnorm(u)
    integral_approx <- mean(f(x))
  } else {
    u <- generator(seed, n)
    x <- a + (b - a) * u
    integral_approx <- mean(f(x)) * (b - a)
  }

  return(integral_approx)
}
```

Kod 4.1. implementacija - klasična Monte Carlo metoda

Kod koji je prikazan na slici 4.1. implementira Monte Carlo metodu za aproksimaciju integrala u R-u i nudi fleksibilnost za rad s različitim vrstama intervala, uključujući beskonačne. Funkcija "MonteCarloIntegracija" uzima nekoliko ulaznih parametara: funkciju f koju želimo integrirati, donju i gornju granicu intervala integracije a i b, broj slučajnih točaka n koje će se koristiti za simulaciju, funk-

ciju "generator" koja generira slučajne brojeve, i sjeme "seed" za osiguranje reproducibilnosti rezultata.

Kod provjerava jesu li granice intervala beskonačne. Ako je barem jedna granica beskonačna, koristi normalnu distribuciju za generiranje slučajnih točaka. U tom slučaju, funkcija "generator" generira slučajne brojeve koje se zatim koriste za dobivanje kvantila normalne distribucije putem funkcije "qnorm". Aproksimacija integrala se dobiva računanjem prosjeka funkcije f evaluirane na tim kvantilima.

Ako su granice intervala konačne, funkcija generira slučajne brojeve između 0 i 1 i skalira ih na interval $[a, b]$. Aproksimacija integrala se tada dobiva kao prosjek funkcije f evaluirane na tim skaliranim točkama, pomnožen s duljinom intervala $(b - a)$. Ova metoda omogućava preciznu aproksimaciju integrala i za beskonačne i za konačne intervale, pružajući fleksibilnost za različite vrste funkcija i intervala.

4.2 Stratificirana Monte Carlo integracija

Stratificirana Monte Carlo integracija smanjuje varijancu rezultata dijeljenjem intervala integracije na manje podintervale, poznate kao stratumi. Unutar svakog stratuma generiraju se slučajni uzorci, što osigurava bolju pokrivenost cijelog intervala i smanjuje osjetljivost na lokalne varijacije funkcije.

Pretpostavimo da interval $[a, b]$ dijelimo na k podintervala. U svakom podintervalu generiramo $\frac{n}{k}$ uzoraka i izračunavamo prosječnu vrijednost funkcije unutar svakog stratuma. Ukupna aproksimacija integrala tada se dobiva sumiranjem doprinosa svih stratuma.

Funkcija "StratificiranaMonteCarlo" u R-u implementira stratificiranu Monte Carlo metodu za aproksimaciju integrala, koja poboljšava preciznost procjene integrala koristeći stratifikaciju kako bi se smanjila varijanca u rezultatu. Funkcija prihvaća nekoliko parametara: funkciju f koju želimo integrirati, donju i gornju granicu intervala integracije a i b, ukupan broj slučajnih točaka n, funkciju "generator" za generiranje slučajnih brojeva, te sjeme "seed" za reproducibilnost.

Kod prvo određuje broj stratuma k kao korijen iz n, s minimalnom vrijednošću 1, kako bi se izbjegli problemi s premalim brojem stratuma. Zatim se izračunava broj točaka po stratumima. Funkcija zatim generira slučajne brojeve pomoću generator funkcije i dodjeljuje stratume svakoj točki. U slučaju beskonačnih intervala, koristi se normalna distribucija za generiranje točaka, a integral se aproksimira kao prosjek funkcije f na tim točkama.

Za konačne intervale, slučajni brojevi se skaliraju na interval $[a, b]$ uzimajući u obzir stratume, a integral se aproksimira kao prosjek funkcije f na tim skaliranim točkama, pomnožen s duljinom intervala. Ova metoda omogućava ravnomjernije pokrivanje intervala integracije i smanjenje varijance procjene, čime se poboljšava točnost Monte Carlo integracije.

```
StratificiranaMonteCarlo <- function(f, a, b, n, generator, seed) {
  if (is.infinite(a) || is.infinite(b)) {
    k <- max(1, floor(sqrt(n)))
    n_per_stratum <- floor(n / k)
    u <- generator(seed, n)
    stratumi <- rep(1:k, each = n_per_stratum)
    x <- qnorm(u)
    integral_approx <- mean(f(x))
  } else {
    k <- max(1, floor(sqrt(n)))
    n_per_stratum <- floor(n / k)
    u <- generator(seed, n)
    stratumi <- rep(1:k, each = n_per_stratum)
    x <- a + (stratumi - 1 + u) * (b - a) / k
    integral_approx <- (b - a) * mean(f(x))
  }

  return(integral_approx)
}
```

Kod 4.2. implementacija - stratificirana Monte Carlo metoda

4.3 Antitetička Monte Carlo integracija

Antitetička Monte Carlo metoda koristi parove slučajnih varijabli kako bi smanjila varijancu procjene integrala. Umjesto korištenja samo slučajnog uzorka u_i, metoda također koristi i njegovu antitetičku vrijednost 1 - u_i , čime se smanjuju odstupanja procjena od stvarne vrijednosti.

U ovoj metodi, generiraju se uzorci u_i i 1 - u_i. Funkcija se evaluira u oba skupa uzoraka, a aproksimacija integrala računa se kao prosjek svih izračunatih vrijednosti.

Funkcija prihvaća nekoliko parametara: funkciju f koju treba integrirati, donju i gornju granicu intervala integracije a i b, ukupni broj slučajnih točaka n, funkciju "generator" koja generira slučajne brojeve, i sjeme "seed" za reproducibilnost. Kod najprije provjerava jesu li granice intervala beskonačne. Ako je to slučaj, koristi normalnu distribuciju za generiranje slučajnih brojeva.

Funkcija generator generira $\frac{n}{2}$ slučajnih brojeva, koji se koriste za izračun kvantila normalne distribucije. Za svaki slučajni broj u, izračunavaju se komplementarni brojevi $1 - u$. Ovi brojevi koriste se za generiranje dvaju skupova točaka: $x1$ i $x2$. Aproksimacija integrala se zatim dobiva kao prosjek funkcije f evaluirane na obje skupine točaka.

Ako su granice intervala konačne, metoda je slična, ali koristi linearnu transformaciju za skaliranje slučajnih brojeva na interval $[a, b]$. Funkcija "generator"

```
AntitetickaMonteCarlo <- function(f, a, b, n, generator, seed) {
  if (is.infinite(a) || is.infinite(b)) {
    u <- generator(seed, n/2)
    x1 <- qnorm(u)
    x2 <- qnorm(1 - u)
    integral_approx <- mean(f(c(x1, x2)))
  } else {
    u <- generator(seed, n/2)
    x1 <- a + (b - a) * u
    x2 <- a + (b - a) * (1 - u)
    integral_approx <- (b - a) * mean(f(c(x1, x2)))
  }

  return(integral_approx)
}
```

Kod 4.3. implementacija - antitetička Monte Carlo metoda

generira $\frac{n}{2}$ slučajnih brojeva između 0 i 1, koji se koriste za dobivanje točaka $x1$ i $x2$ u oba kraja intervala. Aproksimacija integrala se dobiva kao prosjek funkcije f evaluirane na obje skupine točaka, pomnožen s duljinom intervala $(b - a)$.

Ova metoda koristi antitetičke parove koji su komplementarni, kako bi smanjila varijancu u procjeni integrala, jer su $x1$ i $x2$ u svakom paru povezani i suprotstavljeni, čime se smanjuje ukupna varijanca u procjeni integrala i poboljšava preciznost.

4.4 Hit-or-Miss Monte Carlo integracija

$Hit - or - Miss$ metoda intuitivno pristupa problemu integracije generiranjem slučajnih točaka unutar pravokutnika koji obuhvaća funkciju i broji koliko tih točaka pada ispod funkcije. Ovaj pristup koristi jednostavno "pogodi ili promaši" logiku za aproksimaciju integrala.

Određuje se pravokutnik koji potpuno obuhvaća funkciju $f(x)$ na intervalu $[a, b]$. Generira se n slučajnih točaka unutar tog pravokutnika, a broj "pogodaka" ispod funkcije koristi se za procjenu integrala.

Ova metoda generira slučajne točke unutar pravokutnog područja kako bi procijenila površinu ispod funkcijske krivulje. Ako su granice intervala beskonačne, funkcija generira slučajne brojeve pomoću normalne distribucije i koristi procjenu maksimuma funkcije za definiranje pravokutnog područja.

Uspoređuje slučajne brojeve y s vrijednostima funkcije $f(x)$ kako bi izračunala broj "hitova" unutar područja ispod funkcije. Aproksimacija integrala dobiva se

```
HitOrMissMonteCarlo <- function(f, a, b, n, generator, seed) {
  if (is.infinite(a) || is.infinite(b)) {
    u_x <- generator(seed, n)
    u_y <- generator(seed + 1, n)
    x <- qnorm(u_x)
    f_max <- max(f(x))  # Procjena maksimuma funkcije
    y <- f_max * u_y
    hits <- sum(y <= f(x))
    area <- f_max
    integral_approx <- area * (hits / n)
  } else {
    u_x <- generator(seed, n)
    u_y <- generator(seed + 1, n)
    x <- a + (b - a) * u_x
    f_max <- max(f(x))  # Pretpostavka da imamo procjenu maksimuma funkcije
    y <- f_max * u_y
    hits <- sum(y <= f(x))
    area <- (b - a) * f_max
    integral_approx <- area * (hits / n)
  }

  return(integral_approx)
}
```

Kod 4.4. implementacija - $hit - or - miss$ Monte Carlo metoda

kao omjer tih "hitova" i ukupnog broja točaka, pomnožen s površinom pravokutnika.

Za konačne intervale, metoda se prilagođava tako što skalira slučajne brojeve na interval $[a, b]$. Funkcija koristi sličan pristup, ali s pravokutnim područjem koje je sada $(b - a) * f_{max}$. U oba slučaja, metoda omogućava preciznu procjenu integrala, što je korisno za funkcije koje su složene za analitičko izračunavanje.

4.5 Importance Sampling Monte Carlo integracija

Importance Sampling sofisticirana je metoda koja koristi funkciju gustoće $p(x)$ sličnu obliku funkcije $f(x)$ kako bi smanjila varijancu procjene. Ovaj pristup omogućuje fokusiranje uzorkovanja na područja gdje funkcija ima veći doprinos, čime se značajno poboljšava učinkovitost metode.

Integral funkcije $g(x)$ aproksimira se pomoću važnosnog uzorkovanja na sljedeći način:

$$\int_a^b g(x)\,dx \approx \frac{1}{n}\sum_{i=1}^{n} p(x_i)g(x_i)$$

gdje su x_i uzorci generirani prema gustoći $p(x)$.

```
ImportanceSamplingMonteCarlo <- function(f, a, b, n, generator, seed) {
  if (is.infinite(a) || is.infinite(b)) {
    u <- generator(seed, n)
    x <- qnorm(u)
    g <- dnorm(x)
    w <- 1 / g
    integral_approx <- mean(f(x) * w)
  } else {
    u <- generator(seed, n)
    x <- a + (b - a) * u
    g <- dnorm(x, mean = (a + b) / 2, sd = (b - a) / 6)
    w <- 1 / g
    integral_approx <- mean(f(x) * w)
  }

  return(integral_approx)
}
```

Kod 4.5. implementacija - *importance sampling* Monte Carlo metoda

Funkcija "ImportanceSamplingMonteCarlo" implementira Monte Carlo integraciju koristeći tehniku uzorkovanja važnosti (importance sampling) kako bi poboljšala točnost procjene integrala. Ova tehnika uzima u obzir različite distribucije za generiranje slučajnih brojeva kako bi se smanjila varijanca procjene, fokusirajući se na područja gdje funkcija najviše doprinosi vrijednosti integrala. Funkcija prihvaća parametre f (funkciju koju treba integrirati), a i b (granice intervala integracije), n (broj točaka), generator (funkciju za generiranje slučajnih brojeva) i seed (sjeme za reprodukciju rezultata).

Za beskonačne intervale, generiraju se točke pomoću kvantila normalne distribucije (qnorm), a funkcija gustoće vjerojatnosti g je standardna normalna distribucija (dnorm). Težine w se izračunavaju kao recipročna vrijednost gustoće kako bi se prilagodila procjena integrala prema uzorku. Za konačne intervale, generirane točke x se skaliraju na interval $[a, b]$, a distribucija g je normalna s očekivanjem u sredini intervala i standardnom devijacijom koja pokriva širinu intervala.

Integracija se provodi množenjem funkcije $f(x)$ s odgovarajućim težinama w, a konačna procjena se dobiva kao prosječna vrijednost ponderiranih funkcijskih vrijednosti. Ovaj pristup omogućava bolju procjenu integrala, posebno kada funkcija ima dominantna područja koja značajno utječu na ukupni rezultat.

4.6 Random Walk Monte Carlo integracija

Random Walk Monte Carlo integracija je metoda koja koristi nasumičan hod za generiranje točaka unutar intervala integracije. U ovoj metodi, točke se generiraju

kao nasumični koraci počevši od početne točke unutar intervala, pri čemu svaki korak uzima u obzir granice intervala. Metoda je korisna za integrale gdje je funkcija vrlo promjenjiva ili ima lokalne varijacije, jer simulira prirodne procese poput difuzije.

Random Walk Monte Carlo metoda koristi sekvencu $X_1, X_2, \ldots, X_n$, gdje je $X_{i+1} = X_i + \epsilon_i$, a ϵ_i predstavlja slučajan korak generiran prema nekoj distribuciji (npr. normalnoj distribuciji). Cilj je uzorkovati interval koristeći nasumične korake koji omogućuju bolje istraživanje prostora integracije.

Neka je funkcija $f(x)$ definirana na intervalu $[a, b]$, tada se aproksimacija integrala računa kao:

$$I \approx \frac{b-a}{n} \sum_{i=1}^{n} f(X_i)$$

gdje su X_i točke generirane nasumičnim hodom.

```
RandomWalkMonteCarlo <- function(f, a, b, n, generator, seed) {
  if (is.infinite(a) || is.infinite(b)) {
    u <- generator(seed, n)
    x <- numeric(n)
    x[1] <- qnorm(u[1])
    for (i in 2:n) {
      x[i] <- x[i - 1] + rnorm(1)
    }
    integral_approx <- mean(f(x))
  } else {
    u <- generator(seed, n)
    x <- numeric(n)
    x[1] <- a + (b - a) * u[1]
    for (i in 2:n) {
      x[i] <- x[i - 1] + (b - a) * (u[i] - 0.5)
      if (x[i] < a) x[i] <- a
      if (x[i] > b) x[i] <- b
    }
    integral_approx <- (b - a) * mean(f(x))
  }

  return(integral_approx)
}
```

Kod 4.6. implementacija - *random walk* Monte Carlo metoda

Ovaj kod implementira Monte Carlo integraciju uz korištenje metode slučajnog hoda (random walk) kako bi procijenila integral funkcije f. Funkcija uzima interval integracije definiran s granicama a i b, broj uzoraka n, generator slučajnih brojeva "generator", i sjeme "seed" za reproducibilnost. Za beskonačne intervale,

generira se početna točka x_1 pomoću standardne normalne distribucije (qnorm), a zatim se svaka sljedeća točka x_i računa dodavanjem slučajnog pomaka dobivenog iz normalne distribucije prethodnoj točki, čime se simulira slučajni hod.

Ako su granice intervala konačne, početna točka x_1 se skalira na interval $[a, b]$, a svaka sljedeća točka se generira dodavanjem pomaka proporcionalnog širini intervala uzimajući u obzir slučajni faktor u_i - 0.5, pri čemu se vrijednosti ograničavaju na interval $[a, b]$.

Na kraju, integral se aproksimira računanjem prosjeka funkcije f evaluirane na generiranim točkama x te se taj prosjek množi s duljinom intervala $(b - a)$ u slučaju konačnih granica. Ova metoda koristi nasumična kretanja točaka unutar intervala kako bi se što bolje pokrilo područje ispod funkcijske krivulje i omogućila precizna aproksimacija integrala, naročito u slučajevima složenih ili nelinearnih funkcija.

5 | Generatori slučajnih brojeva

Generatori slučajnih brojeva temeljni su alati u primjeni Monte Carlo metoda jer omogućuju stvaranje slučajnog ili pseudoslučajnog uzorka potrebnog za procjenu integrala i drugih numeričkih problema. Iako su konceptualno jednostavni, kvalitetni generatori slučajnih brojeva ključni su za točnost, stabilnost i učinkovitost Monte Carlo metoda. Korištenjem generatora, moguće je simulirati vrijednosti iz različitih distribucija slučajnih varijabli koje odgovaraju specifičnim zahtjevima metode.

Generatori slučajnih brojeva moraju zadovoljavati nekoliko ključnih svojstava: ravnomjernu raspodjelu, dugu periodičnost, brzinu generiranja i izbjegavanje predvidljivih uzoraka. Loši generatori mogu dovesti do značajnih pogrešaka u Monte Carlo procjenama, dok dobri generatori omogućuju preciznije i pouzdanije rezultate.

U ovom poglavlju predstavljeni su različiti generatori slučajnih brojeva, uključujući klasične metode kao što su Linearni Kongruencijalni Generator (LCG) i Multiplikativni Kongruencijalni Generator (MCG), te napredniji pristupi poput Mersenne Twistera i Haltonovih sekvencijalnih generatora. Detaljno su opisane njihove matematičke osnove, prednosti i nedostaci te njihova primjena u Monte Carlo metodama. Korištenje odgovarajućeg generatora može značajno poboljšati učinkovitost i točnost Monte Carlo integracije, što je važno za kompleksne i višedimenzionalne probleme.

5.1 Linearni kongruencijalni Generator (LCG)

Linearni kongruencijalni Generator (LCG) je jedan od najstarijih i najjednostavnijih algoritama za generiranje pseudo-slučajnih brojeva. Ovaj generator koristi rekurzivnu formulu koja uključuje multiplikativnu i aditivnu komponentu te operaciju modulo. Na taj način generira se niz brojeva koji se normaliziraju u interval $[0, 1]$. Zbog svoje jednostavnosti, LCG se koristi u brojnim aplikacijama, no može imati probleme s periodičnošću i kvalitetom generiranih uzoraka, što utječe na rezultate Monte Carlo metoda.

LCG generira niz brojeva koristeći rekurzivnu formulu :

$$X_{n+1} = (a \cdot X_n + c)(mod)m$$

gdje su a, c i m konstante koje definiraju generator. Uobičajeno je da se vrijednosti normaliziraju na interval $[0, 1]$ dijeljenjem s m.

```
LCG <- function(seed, n, a = 1664525, c = 1013904223, m = 2^32) {
  random_numbers <- numeric(n)
  random_numbers[1] <- seed

  for (i in 2:n) {
    random_numbers[i] <- (a * random_numbers[i - 1] + c) %% m
  }

  random_numbers <- random_numbers / m
  return(random_numbers)
}
```

Kod 5.1. implementacija - LCG generator slučajnih brojeva

5.2 Multiplikativni kongruencijalni generator (MCG)

Multiplikativni kongruencijalni Generator (MCG) je varijanta LCG-a koja koristi samo multiplikativnu komponentu, bez aditivne konstante. Ovaj pojednostavljeni pristup omogućava brže generiranje brojeva, ali dijeli slične probleme s kvalitetom generiranih sekvenci kao i LCG.

Formula za generiranje brojeva je :

$$X_{n+1} = (a \cdot X_n)(mod)m$$

MCG generira uzorke brže, ali s većom sklonošću ka obrascima, što može negativno utjecati na stabilnost Monte Carlo metoda.

```
MCG <- function(seed, n, a = 1664525, m = 2^32) {
  random_numbers <- numeric(n)
  random_numbers[1] <- seed

  for (i in 2:n) {
    random_numbers[i] <- (a * random_numbers[i - 1]) %% m
  }

  random_numbers <- random_numbers / m
  return(random_numbers)
}
```

Kod 5.2. implementacija - MCG generator slučajnih brojeva

5.3 Haltonov sekvencijalni generator

Haltonov generator koristi nizove s malim odstupanjem, što omogućuje ravnomjerniju raspodjelu točaka u prostoru, čime se postiže bolja pokrivenost i točnost procjena u kvazi-Monte Carlo metodama.

Sekvencijalni generatori poput Haltonovog koriste osnovne brojeve za generiranje sekvenci koje ravnomjernije pokrivaju interval u usporedbi s klasičnim generatorima slučajnih brojeva.

```
    Halton <- function(n, base = 2) {
  seq <- numeric(n)
  for (i in 1:n) {
    seq[i] <- 0
    f <- 1 / base
    j <- i
    while (j > 0) {
      seq[i] <- seq[i] + f * (j %% base)
      j <- floor(j / base)
      f <- f / base
    }
  }
  return(seq)
}
```

Kod 5.3. implementacija - Haltonov sekvencijalni generator slučajnih brojeva

5.4 Mersenne twister generator

Mersenne twister je jedan od najpopularnijih generatora slučajnih brojeva zbog svoje brzine i visoke kvalitete generiranih brojeva. Ima vrlo dug period i generira brojeve s izvrsnim statističkim svojstvima. Njegova implementacija u R-u omogućuje lako korištenje u raznim aplikacijama Monte Carlo metoda.

Ugrađen je u mnogim programskim jezicima što omogućuje njegovo lako korištenje. Daje visoku kvalitetu i ravnomjernu raspodjelu generiranih brojeva.

```
    MersenneTwister <- function(seed, n) {
  set.seed(seed, kind = "Mersenne-Twister")
  return(runif(n))
}
```

Kod 5.4. implementacija - *Mersenne twister* generator slučajnih brojeva

5.5 Uniformni generator

Uniformni generator koristi standardne funkcije za generiranje ravnomjerno raspoređenih brojeva u intervalu $[0, 1]$. Ovaj generator je jednostavan, brz i često korišten kao osnovni generator u mnogim Monte Carlo metodama.

Također je ugrađen u R-u te je standardiziran i široko dostupan u mnogim programskim jezicima.

```
  UniformGenerator <- function(seed, n) {
set.seed(seed)
return(runif(n))
}
```

Kod 5.5. implementacija - uniformni generator slučajnih brojeva

5.6 Eksponencijalni generator

Eksponencijalni generator koristi eksponencijalnu distribuciju za generiranje slučajnih brojeva. Eksponencijalna distribucija je neprekidna raspodjela koja opisuje vrijeme između događaja u Poissonovom procesu, odnosno u situacijama gdje se događaji događaju neprekidno i nezavisno uz konstantnu stopu. Ovaj generator često se koristi u simulacijama koje modeliraju trajanje ili čekanje, kao što su kvarovi sustava, vrijeme do sloma ili intervali između dolazaka u prometnim tokovima.

Kao i prethodna dva generatora, ugrađen je u mnogim programskim jezicima pa tako i u R-u.

```
  ExponentialGenerator <- function(seed, n) {
set.seed(seed)
return(rexp(n))
}
```

Kod 5.6. implementacija - eksponencijalni generator slučajnih brojeva

5.7 Normalni generator

Normalni generator koristi normalnu distribuciju (poznatu i kao Gaussovu distribuciju) za generiranje slučajnih brojeva. Normalna distribucija je jedna od najvažnijih i najčešće korištenih neprekidnih raspodjela u statistici zbog svojih jedinstvenih svojstava, poput simetričnosti i oblika zvona. Koristi se za modeliranje fenomena koji su rezultat zbrajanja brojnih nezavisnih faktora, što se često susreće u prirodnim i društvenim znanostima.

Ugrađen je u programskom jeziku R, što omogućuje njegovo lako korištenje.

```
  NormalGenerator <- function(seed, n) {
set.seed(seed)
return(rnorm(n))
}
```

Kod 5.7. implementacija - normalni generator slučajnih brojeva

6 | Implementacija Monte Carlo metode

Cjelokupna implementacija svih opisanih metoda i generatora slučajnih brojeva te njihova integracija unutar Shiny sučelja dostupna je na poveznici : Monte Carlo metode - implementacija u R - u

Implementacija različitih Monte Carlo integracija i generatora slučajnih brojeva ključna je za praktičnu primjenu teorijskih znanja. Korištenje programskog jezika R, uz Shiny sučelje, omogućuje korisnicima jednostavnu interakciju i analizu različitih metoda Monte Carlo integracije te njihovih performansi u stvarnom vremenu. U ovom poglavlju opisane su implementacije svih korištenih metoda i generatora te način na koji su povezani kroz Shiny sučelje.

Cjelokupni sustav implementiran je tako da omogućuje korisnicima brzo prebacivanje između različitih metoda i generatora, prikaz rezultata i usporedbu Monte Carlo integracije s klasičnim numeričkim metodama. Ovaj pristup olakšava eksperimentiranje i analizu performansi, što je ključno za primjenu Monte Carlo metoda u stvarnim problemima.

6.1 Implementacija Monte Carlo metoda i generatora slučajnih brojeva

Svaka Monte Carlo metoda implementirana je kao zasebna funkcija unutar programskog jezika R, omogućujući fleksibilnost pri izboru metoda za različite tipove problema. Implementacija je strukturirana tako da korisnici mogu jednostavno mijenjati parametre, odabirati različite metode i uspoređivati rezultate. Ove metode su integrirane unutar cjelokupnog sustava na način koji omogućava korisnicima brz pristup i testiranje.

Generatori slučajnih brojeva ključni su dio svake Monte Carlo metode jer direktno utječu na raspodjelu uzoraka i konačne rezultate. Svaki generator implementiran je kao posebna funkcija koja se može pozvati unutar bilo koje metode integracije. Time se korisnicima omogućuje jednostavan odabir generatora koji najbolje odgovara specifičnoj metodi.

6.2 Shiny sučelje za interaktivnu analizu

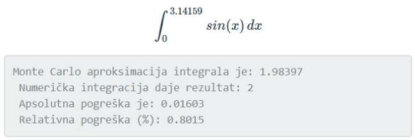
Shiny sučelje je interaktivna web aplikacija razvijena unutar R programskog jezika koja omogućuje dinamičku vizualizaciju podataka i interaktivno korisničko iskustvo. Shiny povezuje snagu R-a za statističku analizu s modernim web tehnologijama, pružajući korisnicima alat za brzo testiranje, analiziranje i vizualizaciju rezultata u stvarnom vremenu bez potrebe za složenim programiranjem ili posebnim web razvojnim vještinama.

U kontekstu cjelokupne implementacije, Shiny sučelje omogućuje korisnicima jednostavno testiranje različitih Monte Carlo metoda i generatora slučajnih brojeva za integraciju. Kroz sučelje, korisnici mogu unositi matematičke funkcije za integraciju, definirati granice integracije, odabrati *seed* iz kojeg će se realizirati slučajni uzorak te eksperimentirati s različitim metodama i generatorima. Sučelje je intuitivno i korisnicima pruža mogućnost brzih promjena parametara, omogućujući im da odmah vide kako te promjene utječu na konačne rezultate aproksimacije.

Slika 6.1: Shiny sučelje za Monte Carlo integraciju - Tab "Unos"

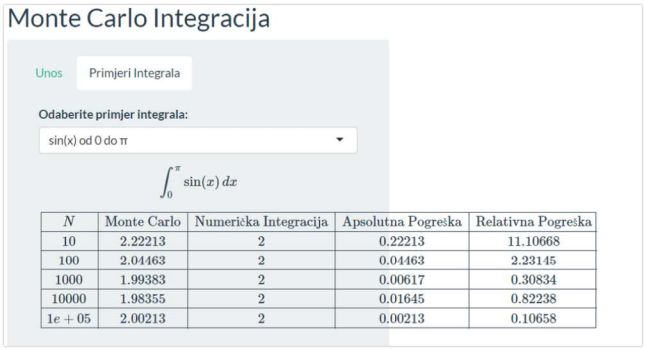
Jedna od ključnih prednosti Shiny sučelja je vizualizacija rezultata. Aplikacija ne samo da prikazuje numeričke vrijednosti aproksimiranih integrala, već i grafički prikazuje funkciju te slučajno generirane točke koje se koriste u Monte Carlo integraciji. Ovakav vizualni prikaz olakšava razumijevanje načina na koji različite metode i generatori utječu na preciznost i stabilnost rezultata. Na primjer, korisnici mogu vidjeti kako se uzorci distribuiraju u stratificiranoj Monte Carlo metodi u usporedbi s klasičnim uniformnim uzorkovanjem ili kako antitetička metoda smanjuje varijabilnost rezultata.

Osim što pruža detaljan uvid u performanse različitih metoda, Shiny aplikacija omogućuje i usporedbu Monte Carlo aproksimacija s numeričkim integracijama, što je korisno za procjenu točnosti i učinkovitosti svake metode. Na taj način, Shiny sučelje služi kao moćan alat za eksperimentiranje, edukaciju i analizu, što



Slika 6.2: Shiny sučelje za Monte Carlo integraciju - rezultat

ga čini idealnim za primjene u obrazovanju, istraživanju i industriji, gdje se Monte Carlo metode koriste za procjenu složenih matematičkih problema. Fleksibilnost i pristupačnost Shiny sučelja omogućuju korisnicima, bez obzira na njihovu razinu stručnosti, da duboko istraže dinamiku Monte Carlo integracija i steknu dublje razumijevanje njihove primjene.



N	Monte Carlo	Numerička Integracija	Apsolutna Pogreška	Relativna Pogreška
10	2.22213	2	0.22213	11.10668
100	2.04463	2	0.04463	2.23145
1000	1.99383	2	0.00617	0.30834
10000	1.98355	2	0.01645	0.82238
$1e+05$	2.00213	2	0.00213	0.10658

Slika 6.3: Shiny sučelje za Monte Carlo integraciju - Tab "Primjeri integrala"

Ako se implementira dodatna funkcionalnost, kao što su naprednije vizualizacije ili prilagodba izgleda sučelja, Shiny može postati još korisniji za specifične istraživačke ili obrazovne svrhe, omogućujući personalizirano iskustvo koje odgovara potrebama korisnika. Korištenjem Shiny sučelja, implemetacija opisanih Monte Carlo metoda i generatora slučajnih brojeva ne samo da ilustrira njihove teorijske koncepte, već ih prenosi u dinamičko okruženje gdje korisnici aktivno sudjeluju u procesu istraživanja i učenja.

7 | Rezultati i analiza

Povezivanje u Shiny sučelju pruža robusnu platformu za eksperimentiranje s implementiranim Monte Carlo metodama i generatorima slučajnih brojeva. Pomoću Shiny sučelja, korisnici mogu jednostavno istražiti različite pristupe integraciji i analizirati rezultate kako bi odabrali optimalnu kombinaciju za specifične potrebe. Također, korisnik je u mogućnosti na primjerima vidjeti kako se pogreška smanjuje povećavanjem dimenzije uzorka n, te kako Monte Carlo metode djeluju na neke specifične integrale.

Rezultati pokazuju da Stratificirana i Antitetička Monte Carlo metoda pružaju preciznije rezultate u usporedbi s Klasičnom metodom zbog smanjenja varijance.

Generatori poput Mersenne Twistera i Haltonovog niza daju bolje rezultate zbog bolje distribucije točaka, dok jednostavni generatori poput LCG-a mogu generirati točke koje nisu dovoljno uniformne.

Za integrale s beskonačnim granicama, najefikasnija se pokazala Importance Sampling metoda.

Shiny sučelje omogućuje korisnicima vizualnu usporedbu rezultata i bolji uvid u efikasnost različitih metoda.

Literatura

[1] Asmussen S., Glynn P. W., *Stochastic Simulation: Algorithms and Analysis*, Springer Science and Business Media, 2007.

[2] Benšić M., Šuvak N., *Primijenjena statistika*, Sveučilište J.J. Strossmayera, Odjel za matematiku, Osijek, 2013.

[3] Benšić M., Šuvak N., *Uvod u vjerojatnost i statistiku*, Sveučilište J.J. Strossmayera, Odjel za matematiku, Osijek, 2013.

[4] Billingsley P., *Probability and Measure (3rd ed.)*, NY: Wiley, New York, 1995.

[5] Fishman G. S., *Monte Carlo: Concepts, Algorithms, and Applications*, Springer Science and Business Media, 2013.

[6] Glasserman P., *Monte Carlo Methods in Financial Engineering*, Springer Science and Business Media, 2003.

[7] Hastings W. K., *Monte Carlo sampling methods using Markov chains and their applications*, Biometrika, 1970. (*57(1), 97-109.*).

[8] Knuth D. E., *The Art of Computer Programming, Volume 2: Seminumerical Algorithms (3rd Edition)*, Addison-Wesley Professional, 1997. (*poglavlja o generatorima slučajnih brojeva*).

[9] Niederreiter H., *Random Number Generation and Quasi-Monte Carlo Methods*, SIAM, 1992.

[10] Robert C. P., Casella G., *Monte Carlo Statistical Methods*, Springer Science and Business Media, 2004.

[11] Ross S., *A First Course in Probability (9th ed.)*, Upper Saddle River, NJ: Pearson, 2014.

[12] Rubinstein R. Y., Kroese D. P. , *Simulation and the Monte Carlo Method*, John Wiley and Sons, 2016.

[13] Smith E., *Monte Carlo integration*, presentation (https://researchgate.net/).

[14] Sobol I. M., *Uniformly distributed sequences with an additional uniform property*, USSR Computational Mathematics and Mathematical Physics, 1976. (*16(5), 236-242.*).

Sažetak

U ovom radu analizirana je primjena Monte Carlo integracije pomoću različitih generatora slučajnih brojeva, s posebnim naglaskom na praktičnu implementaciju metoda u programskom jeziku R i Shiny sučelju. Monte Carlo metode predstavljaju moćan alat za numeričko rješavanje problema koji su često deterministički, ali zbog svoje složenosti zahtijevaju procjenu pomoću slučajnog uzorka. U fokusu ovog rada je Monte Carlo integracija, koja se koristi za aproksimaciju integrala na temelju slučajno generiranih točaka unutar zadanog intervala integracije. U teorijskom dijelu rada obuhvaćeni su različiti pristupi Monte Carlo integraciji, uključujući klasičnu, stratificiranu, antitetičku i $hit - or - miss$ metodu, kao i važnosno uzorkovanje $(importance\ sampling)$ te metoda nasumičnog hoda $(random\ walk)$. Korištenjem različitih generatora slučajnih brojeva, poput linearnog kongruencijalnog generatora (LCG), Mersenne twistera, Haltonovih sekvenci, te eksponencijalnih i normalnih generatora, istražena je preciznost i učinkovitost svake metode. Poseban naglasak stavljen je na implementaciju svih opisanih metoda i generatora unutar okvira programskog jezika R, kao i na njihovu integraciju u interaktivno Shiny sučelje. Sučelje omogućuje korisnicima jednostavan odabir različitih metoda i generatora, prikaz rezultata aproksimacije integrala u stvarnom vremenu te vizualizaciju uzoraka i funkcija. Analizirane su prednosti i nedostaci svake metode te kako odabir generatora slučajnih brojeva može značajno utjecati na konačne rezultate integracije.

Ključne riječi

Monte Carlo integracija, generatori slučajnih brojeva, aproksimacija integrala, R programiranje, Shiny sučelje, implementacija

Monte Carlo integration using different random number generators

Summary

In this thesis, the application of Monte Carlo integration using different random number generators is analyzed, with special emphasis on the practical implementation of methods in the R programming language and the Shiny interface. Monte Carlo methods represent a powerful tool for numerically solving problems that are often deterministic, but due to their complexity require evaluation using a random sample. The focus of this work is Monte Carlo integration, which is used to approximate integrals based on randomly generated points within a given interval of integration. In the theoretical part of the thesis, various approaches to Monte Carlo integration are covered, including the classic, stratified, antithetical and hit-or-miss method, as well as importance sampling and the random walk method. Using different random number generators, such as the linear congruential generator (LCG), Mersenne twister, Halton sequences, and exponential and normal generators, the precision and efficiency of each method was investigated. Particular emphasis is placed on the implementation of all the described methods and generators within the framework of the R programming language, as well as on their integration into the interactive Shiny interface. The interface allows users to easily select different methods and generators, display the results of real-time approximation of integrals, and visualize patterns and functions. The advantages and disadvantages of each method and how the selection of random number generators can significantly affect the final integration results were analyzed.

Keywords

Monte Carlo integration, random number generators, integral estimation, R programming, Shiny interface, implementation

Životopis

Moje ime je Stela Knežević. Rođena sam 17. svibnja 2001. godine u Požegi. Stanujem u mjestu Dolac par kilometara udaljenom od Požege. Nakon završene Osnovne škole Dragutina Lermana u Brestovcu, 2016. godine upisujem Gimnaziju u Požegi. Iste godine osvajam Godišnju nagradu općine Brestovac za izniman uspjeh i dostignuća u osnovnoškolskom obrazovanju. Godine 2020., završavam opću gimnaziju s odličnim uspjehom i upisujem prijediplomski sveučilišni studij Matematike i računarstva u Osijeku na tadašnjem Odjelu za matematiku, a danas Fakultetu primijenjene matematike i informatike.