UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA DELL'INFORMAZIONE

"Metodi Monte Carlo: un'analisi teorica e applicativa"

Relatore: Prof. Giancarlo Calvagno

Laureando: Alessandro Giuffrè

ANNO ACCADEMICO 2021 – 2022

Data di laurea 22/09/2022

Sommario

Il metodo Monte Carlo è fondamentale per la risoluzione di moltissimi problemi, sia in ambito classicamente matematico sia in altre discipline scientifiche. In questo scritto, dopo una breve introduzione che passa in rassegna i dettagli storici della sua nascita, viene definito il funzionamento del metodo MC, partendo dai teoremi su cui si basa. Si introducono anche dei metodi di calcolo dell'errore, dato che il metodo opera su base probabilistica. Si introduce poi un aspetto importante e necessario del metodo, ovvero i generatori di numeri casuali e si accenna al funzionamento di alcuni di questi. Infine si propongono tre esempi di uso del metodo: il calcolo di π, il calcolo di integrali uni e multi dimensionali e infine l'applicazione del metodo al problema del ripiegamento delle proteine.

Indice

1 Introduzione

I metodi Monte Carlo (MC) sono un insieme di processi per la risoluzione di problemi tramite la generazione di numeri casuali e iterazioni multiple. Nascono come metodi risolutivi alternativi per problemi impossibili da risolvere in modo analitico. A livello fondamentale i metodi MC mirano ad invertire il solito paradigma dei problemi statistici, ovvero l'uso di quantità casuali per stimare quantità deterministiche invece della stima di quantità aleatorie in modo deterministico. I classici esempi dell'uso di questa tecnica sono usati nel campo della matematica per svolgere integrali e per l'ottimizzazione di sistemi, tuttavia sono diventatati essenziali anche in una varietà di ambiti, tra cui la finanza, la biologia, la climatologia e persino l'ambito legislativo.[12][17]

1.1 Cenni storici

Uno dei primi problemi per cui fu proposto un metodo MC, di cui si hanno prove documentate, è il calcolo di π usando il framework del problema dell'ago di Buffon risalente al 18-esimo secolo. Il problema consisteva nel calcolare, dato un pavimento composto da barre parallele dello stesso spessore, la probabilità che un ago cadesse sopra la striscia tra due barre. Nella risoluzione si è notata una dipendenza del risultato da π, quindi modificando il problema e generando cadute casuali di aghi si poteva giungere al valore di π. Enrico Fermi fu il primo a concretizzare l'idea di risoluzione tramite generazione casuale di numeri, usandola, secondo le testimonianze di Emilio Segrè, nei suoi studi sulla "neutron diffusion", ma sebbene fosse di grande utilità nella risoluzione di molti problemi, non pubblicò mai sull'argomento, né gli diede un nome. Nel 1946 Stanislaw Ulam, lavorando nei laboratori di Los Alamos, propose di risolvere il problema della "neutron diffusion" nel nucleo di un'arma nucleare tramite esperimenti aleatori. Con l'aiuto di John von Neumann iniziarono ad elaborare i calcoli necessari. Essendo un'operazione segreta, per la quale era necessario un nome in codice per il lavoro compiuto, un collega di Neumann e Ulam propose il nome Monte Carlo, un casinò a Monaco dove lo zio di Ulam giocava i soldi chiesti in prestito ai familiari. Nel 1948 i due riuscirono a programmare il computer ENIAC per compiere la prima versione automatizzata di un metodo MC, risolvendo così il problema che si erano posti. Dal 1950 questi metodi iniziarono a diffondersi nel mondo della fisica, attraverso applicazioni in una larga gamma di discipline. Già negli anni '60 era stato approfondito uno studio di metodi MC più sofisticati sulla base della teoria di campo medio, ovvero l'analisi di modelli stocastici ad elevati gradi di libertà approssimandoli tramite modelli più semplici con meno gradi di libertà ottenuti tramite la media di quelli originali, per evolvere, nel 1984, con la creazione dei metodi MC quantistici (interpretabili come una approssimazione MC di campo medio di particella degli integrali di linea Feynman-Kac). Infine, nel 1993, Gordon et al. hanno pubblicato il loro lavoro sulla prima applicazione di un algoritmo di ricampionamento in inferenza statistica Bayesiana, chiamandolo "bootstrap filter",

creando così il MC sequenziale o filtro di particelle. Lo studio di questi metodi è tuttora attivo e trova utilizzo e risultati in diversi ambiti teorici ed applicativi.[12][18]

2 Elementi fondamentali del metodo

2.1 Legge dei grandi numeri

Il principio di probabilità su cui si basa il metodo MC è la legge dei grandi numeri (LLN). La LLN studia la convergenza in probabilità delle medie aritmetiche di sequenze di variabili aleatorie. Qui riporto la definizione di convergenza in probabilità e descrivo due formulazioni della LLN.

2.1.1 Convergenza in probabilità

Definizione 2.1. *La sequenza di variabili aleatorie* $\{X_n\}$ *converge in probabilità alla variabile aleatoria* X *se:*

$$\lim_{n\to\infty} P(|X_n - X| \geq \varepsilon) = 0, \qquad \forall \varepsilon > 0$$

allora si scrive

$$X_n \xrightarrow{P} X$$

2.1.2 LLN Chebyshev 1867

Teorema 2.1. *Sia* $\{X_n\}$ *una sequenza di v.a. indipendenti e identicamente distribuite (iid) Se le v.a. della sequenza hanno secondo momento* $E(X_1^2) < \infty$*, allora, detto* $\mu = E(X_1) < \infty$*,*

$$\overline{X}_n \xrightarrow{P} \mu$$

Dimostrazione. Per la disuguaglianza di Chebyshev

$$P(|\overline{X}_n - \mu| \geq \varepsilon) \leq \frac{E(|\overline{X}_n - \mu|^2)}{\varepsilon^2} = \frac{\mathrm{var}(\overline{X}_n)}{\varepsilon^2} = \frac{\sigma^2}{n\varepsilon^2} \to 0, \quad \forall \varepsilon > 0$$

essendo

$$\mathrm{var}(\overline{X}_n) = \mathrm{var}\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) = \frac{1}{n^2} n\sigma^2 = \frac{\sigma^2}{n} \to 0$$

poiché le variabili sono iid. Ma la prima espressione a cui siamo giunti è la definizione di convergenza in probabilità, da cui la tesi. □

2.1.3 LLN Khinchine 1928

Teorema 2.2. *Sia $\{X_n\}$ una sequenza di va, iid. Se le v.a. della sequenza hanno valore atteso, $\mu = E(X_1) < \infty$, allora*

$$\overline{X}_n \xrightarrow{P} \mu$$

Dimostrazione. Sia $\varphi_{X_1}(\omega)$ la funzione caratteristica (CF) della v.a. X_1. Poiché per ipotesi X_1 ammette valore atteso, $\varphi_{X_1}(\omega) \in C^1(\mathbb{R})$, quindi la formula di Mac-Laurin al primo ordine è

$$\varphi_{X_1}(\omega) = \varphi_{X_1}(0) + \left[\frac{d}{d\omega}\varphi_{X_1}(\omega)\right]_{\omega=0} \omega + o(\omega) = 1 + j\mu\omega + o(\omega)$$

Poiché le v.a. X_n sono iid, la CF di $\overline{X}_n$ vale

$$\varphi_{\overline{X}_n}(\omega) = \left[\varphi_{X_1}\left(\frac{\omega}{n}\right)\right]^n = \left(1 + j\mu\frac{\omega}{n} + o\left(\frac{\omega}{n}\right)\right)^n$$

Il calcolo del limite dà

$$\lim_{n\to\infty} \varphi_{\overline{X}_n}(\omega) = \lim_{n\to\infty} \left(1 + j\mu\frac{\omega}{n} + o\left(\frac{\omega}{n}\right)\right)^n = e^{j\omega\mu}$$

Essendo $\varphi(\omega) = e^{j\omega\mu}$ la CF della variabile aleatoria costante $X = \mu$, si ha che, per il teorema della continuità, $\overline{X}_n$ converge in distribuzione alla costante μ e quindi converge a μ anche in probabilità. □

Più semplicemente possiamo dire che il valore della media di una quantità molto grande di valori prodotti da una successione di v.a. iid è arbitrariamente vicino al valore atteso della prima variabile.[8]

2.2 Il metodo Monte Carlo

Ora possiamo introdurre il funzionamento vero e proprio del metodo MC. Al livello più basico consiste nel descrivere il problema che si vuole risolvere come valore atteso di una variabile aleatoria, per poi usare la LLN.[8] Ovvero

Consideriamo una v.a. X e una funzione f misurabile allora

$$E[f(X)] \approx \frac{1}{n}\sum_{i=1}^{n} f(X_i) \qquad \text{per } n \to \infty$$

Dove X_i sono realizzazioni indipendenti della variabile aleatoria X.

2.3 Calcolo dell'errore

Per il calcolo dell'errore è necessario poter calcolare la probabilità

$$P(|\overline{X}_n - \mu| \leq \varepsilon)$$

dove $\overline{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$, $\mu = E(X_1)$ e ε è un valore positivo arbitrariamente piccolo, tutto definito su una sequenza di v.a. iid. $\{X_n\}_{n\geq 1}$. Essenzialmente vogliamo calcolare quanto probabile sia che la media campionaria delle prime n v.a. si discosti dalla media della prima v.a. più di un certo valore ε. Questo calcolo risulta molto difficile a prima vista, poiché le v.a. $\overline{X}_n$ non sono identicamente distribuite né indipendenti. Tuttavia grazie al teorema limite centrale (CLT) si può evitare il problema per valori di n elevati. Prima di introdurre il teorema definisco la convergenza in distribuzione e il teorema di Lévy.

2.3.1 Convergenza in distribuzione

Definizione 2.2. *La sequenza di v.a. $\{X_n\}_{n\geq 1}$ di funzioni di distribuzione (FdD) $\{F_n(x)\}_{n\geq 1}$, converge in distribuzione se esiste una FdD $F(x)$ tale che*

$$\lim_{n\to\infty} F_n(x) = F(x) \qquad \textit{per ogni } x \textit{ dove } F(x) \textit{ è continua}$$

allora si scrive

$$X_n \xrightarrow{D} F(x)$$

2.3.2 Teorema di Lèvy

Teorema 2.3. *Sia $\{X_n\}_{n\geq 1}$ una sequenza di v.a. di CF $\{\varphi_n(\omega)\}$. Si supponga che*

$$\lim_{n\to\infty} \varphi_n(\omega) = \varphi(\omega) \qquad \textit{per ogni } \omega \in \mathbb{R}$$

per qualche funzione $\varphi(\omega)$. Sotto queste condizioni sono equivalenti le seguenti proposizioni

1. *$\varphi(\omega)$ è continua nell'origine;*
2. *$\varphi(\omega)$ è una CF;*
3. *$X_n \xrightarrow{D} F(x)$, per qualche $F(x)$ la cui CF è $\varphi(\omega)$.*

2.3.3 Teorema limite centrale

La seguente è la formulazione di Lindeberg-Lévy.

Teorema 2.4. *Sia* $\{X_n\}_{n\geq 1}$ *una sequenza di v.a. iid. Se* $E(X_1^2) < \infty$, *e* $\sigma^2 = var(X_1) > 0$, *allora detto* $\mu = E(X_1)$ *e* $\mathcal{N}(0,1)$ *una gaussiana standardizzata*

$$\frac{\sqrt{n}}{\sigma}(\overline{X}_n - \mu) \xrightarrow{D} \mathcal{N}(0,1)$$

Dimostrazione. Sia $W_n = \frac{\sqrt{n}}{\sigma}(\overline{X}_n - \mu)$. Rappresentiamo W_n come somma di v.a. iid standardizzate

$$W_n = \frac{\sqrt{n}}{\sigma}(\overline{X}_n - \mu) = \frac{\sqrt{n}}{\sigma}\left(\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right) = \frac{1}{\sqrt{n}}\sum_{i=1}^{n}\frac{X_i - \mu}{\sigma} = \frac{1}{\sqrt{n}}\sum_{i=1}^{n} Z_i$$

dove $Z_i = \frac{X_i - \mu}{\sigma}$. Per costruzione $E(Z_i) = 0$ e $\text{var}(Z_i) = 1$. Sia $\varphi(\omega)$ la CF delle Z_i e poiché le v.a. ammettono varianza $\varphi(\omega)$ è derivabile con derivata seconda continua. L'approssimazione di Mac-Laurin al secondo ordine di $\varphi(\omega)$ è

$$\varphi(\omega) = \varphi(0) + \varphi'(0)\omega + \varphi''(0)\frac{\omega^2}{2} + o(\omega^2)$$

Considerando che $\varphi(0) = 1$, $\quad \frac{1}{j}\varphi'(0) = E(Z_i) = 0$, $\quad \frac{1}{j^2}\varphi''(0) = E(Z_i^2) = \text{var}(Z_i) = 1$ si ottiene $\varphi'(0) = 0$ e $\varphi''(0) = -1$. Infine sostituendo

$$\varphi(\omega) = 1 - \frac{\omega^2}{2} + o(\omega^2)$$

Le variabili Z_i sono iid quindi la CF di $S_n = \sum_{i=1}^{n} Z_i$ è

$$\varphi_{S_n}(\omega) = \varphi^n(\omega) = \left(1 - \frac{\omega^2}{2} + o(\omega^2)\right)$$

e quindi la CF di $W_n = \frac{1}{\sqrt{n}}S_n$ è

$$\begin{aligned}\varphi_{W_n}(\omega) = \varphi_{S_n}\left(\frac{\omega}{\sqrt{n}}\right) &= \left(1 - \frac{\left(\frac{\omega}{\sqrt{n}}\right)^2}{2} + o\left(\left(\frac{\omega}{\sqrt{n}}\right)^2\right)\right) \\ &= \left(1 - \frac{\frac{\omega^2}{2}}{n} + o\left(\frac{\omega^2}{n}\right)\right)^n\end{aligned}$$

Calcolando il limite si giunge a

$$\lim_{n\to\infty} \varphi_{W_n}(\omega) = \lim_{n\to\infty} \left(1 - \frac{\frac{\omega^2}{2}}{n} + o\left(\frac{\omega^2}{n}\right)\right)^n = e^{-\frac{\omega^2}{2}}$$

Poiché il limite è la CF di una v.a. $\mathcal{N}(0,1)$, possiamo concludere la dimostrazione invocando il teorema di Lévy.

□

L'importanza di questo teorema deriva dall'assenza di condizioni sulle specifiche v.a. di partenza. Ciò significa che la valutazione dell'equazione per il calcolo dell'errore introdotta prima si riduce a cercare i valori della funzione $\Phi(x)$ in una tabella [8], infatti

$$\begin{aligned} P(|\overline{X}_n - \mu| \leq \varepsilon) &= P(-\varepsilon \leq \overline{X}_n - \mu \leq \varepsilon) = P\left(-\varepsilon \leq \frac{1}{n}S_n - \mu \leq \varepsilon\right) \\ &= P(-n\varepsilon \leq S_n - n\mu \leq n\varepsilon) = P\left(-\frac{n\varepsilon}{\sigma\sqrt{n}} \leq \frac{S_n - n\mu}{\sigma\sqrt{n}} \leq \frac{n\varepsilon}{\sigma\sqrt{n}}\right) \\ &= P\left(-\frac{n\varepsilon}{\sigma\sqrt{n}} \leq Z \leq \frac{n\varepsilon}{\sigma\sqrt{n}}\right) = 2\mathbf{\Phi}\left(\frac{n\varepsilon}{\sigma\sqrt{n}}\right) - 1 \end{aligned}$$

2.3.4 Teorema Berry-Esseen

Il teorema di Berry-Esseen è un teorema simile al CLT che quantifica lo scostamento di una somma di v.a. da una distribuzione normale. Prima di enunciare e dimostrare il teorema, riporto dei risultati necessari.

Definizione 2.3 (Distribuzione di Polya)**.** *La distribuzione di Polya ha densità:*

$$f_P(x) = \frac{1 - \cos x}{\pi x^2}$$

e CF:

$$\varphi_P(t) = (1 - |t|)^+$$

dove $(\cdot)^+$ *indica la parte positiva della funzione. [5][7]*

Teorema 2.5. *Sia* $\varphi(t) = \int e^{itx}\mu(dx)$ *dove* μ *è una misura di probabilità. Se* $\int |\varphi(t)|dt < \infty$ *allora* μ *ha densità limitata e continua pari a*

$$f(y) = \frac{1}{2\pi}\int e^{-ity}\varphi(t)dt$$

Questo teorema è una conseguenza della formula di inversione:

$$\lim_{T\to\infty} \frac{1}{2\pi}\int_{-T}^{T} \frac{e^{-ita} - e^{-itb}}{it}\varphi(t)dt = \mu(a,b) + \frac{1}{2}\mu(\{a,b\})$$

dove φ è la CF rispetto alla misura di probabilità μ.[7]

Lemma 1 (Riemann-Lebesgue)**.** *Data $f \in L(\mathbb{R})^1$ allora per la trasformata di Fourier di f, F, vale*

$$F(z) = \int_{\mathbb{R}} f(x)e^{-izx}dx \to 0 \text{ per } |z| \to \infty$$

Dimostrazione. L'idea dietro a questa dimostrazione consiste nel trovare una minoranza tra la funzione $f(x)$ e una funzione che sappiamo sia arbitrariamente piccola, in modo tale da poter poi mostrare che la funzione $f(x)$ sia anch'essa arbitrariamente piccola. Partiamo supponendo $f(x) = \chi_{(a,b)}(x)$, cioè la funzione indicatrice nell'intervallo (a, b), allora

$$\int f(x)e^{i\lambda x}dx = \int_a^b e^{i\lambda x}dx = \frac{e^{i\lambda b} - e^{i\lambda a}}{i\lambda} \to 0 \text{ per } |\lambda| \to \infty$$

Per l'additività dei limiti, lo stesso vale per una funzione gradino arbitraria, cioè per ogni funzione del tipo:

$$f(x) = \sum_{i=1}^{N} c_i\chi_{(a_i,b_i)}, \quad c_i \in \mathbb{R}, \quad a_i \leq b_i \in \mathbb{R}$$

Abbiamo che

$$\lim_{|\lambda|\to\infty} \int f(x)e^{i\lambda x}dx = 0$$

Infine sia $f \in L^1$ arbitraria. Sia $\varepsilon \in \mathbb{R} > 0$ fissato. Per le proprietà delle funzioni gradino in L^1 esiste una funzione gradino g tale che

$$\int |f(x) - g(x)|dx < \varepsilon$$

poiché g è una funzione gradino e per quello che abbiamo dimostrato prima deve esistere un $N \in \mathbb{N}$ tale che per ogni $|\lambda| > N$

$$\left| \int g(x)e^{i\lambda x}dx \right| < \varepsilon$$

Per additività degli integrali

$$\int f(x)e^{i\lambda x}dx = \int (f(x) - g(x))e^{i\lambda x}dx + \int g(x)e^{i\lambda x}dx$$

Per la disuguaglianza triangolare e poiché $e^{i\lambda x}dx < 1$

$$\left|\int f(x)e^{i\lambda x}dx\right| \leq \int |f(x)-g(x)|dx + \left|\int g(x)e^{i\lambda x}dx\right| < 2\varepsilon$$

Per ogni $|\lambda| > N$. Il limite destro è dovuto ai limiti discussi sopra. Poiché ε è arbitrario

$$\lim_{|\lambda|\to\infty} \int f(x)e^{i\lambda x}dx = 0$$

per ogni $f \in L^1$ [28] □

Lemma 2.

$$\left|e^{ix} - \sum_{m=0}^{n} \frac{(ix)^m}{m!}\right| \leq \min\left(\frac{|x|^{n+1}}{(n+1)!}, \frac{2|x|^n}{n!}\right)$$

Dimostrazione. Integriamo la funzione $f(x) = (x-s)^n e^{is}$ in ds per parti

$$\int_0^x (x-s)^n e^{is}ds = \frac{x^{n+1}}{n+1} + \frac{i}{n+1}\int_0^x (x-s)^{n+1}e^{is}ds$$

Per $n = 0$ si ha

$$\int_0^x e^{is}ds = x + i\int_0^x (x-s)e^{is}ds$$

Integrando l'esponenziale e riordinando si ottiene

$$e^{ix} = 1 + ix + i^2\int_0^x (x-s)e^{is}ds$$

Per $n = 1$ si ha

$$\int_0^x (x-s)e^{is}ds = \frac{x^2}{2} + \frac{i}{2}\int_0^x (x-s)^2 e^{is}ds$$

Si trova il valore dell'integrale a sinistra dall'equazione precedente e sostituendo si ottiene

$$e^{ix} = 1 + ix + \frac{i^2x^2}{2} + \frac{i^3}{2}\int_0^x (x-s)^2 e^{is}ds$$

Iterando il processo si ottiene

$$e^{ix} - \sum_{m=0}^{n} \frac{(ix)^m}{m!} = \frac{i^{n+1}}{n!}\int_0^x (x-s)^n e^{is}ds \tag{1}$$

Ora stimando la quantità a destra per x piccoli, quindi poiché $|e^{is}| \leq 1 \quad \forall s$

$$\left|\frac{i^{n+1}}{n!}\int_0^x (x-s)^n e^{is}ds\right| \leq \left|\frac{i^{n+1}}{n!}\int_0^x (x-s)^n ds\right| = \left|\frac{i^{n+1}}{n!}\frac{x^{n+1}}{n+1}\right| \leq \frac{|x|^{n+1}}{(n+1)!} \tag{2}$$

Ora per x grandi, si parte integrando per parti il seguente integrale

$$\frac{i}{n}\int_0^x (x-s)^n e^{is}ds = -\frac{x^n}{n} + \int_0^x (x-s)^{n-1}e^{is}ds$$

Siccome $\frac{x^n}{n} = \int_0^x (x-s)^{n-1}ds$ si può scrivere

$$\frac{i^{n+1}}{n!}\int_0^x (x-s)^n e^{is}ds = \frac{i^n}{(n-1)!}\int_0^x (x-s)^n (e^{is}-1)ds$$

e visto che $|e^{ix}-1| \leq 2$ si ottiene

$$\left|\frac{i^{n+1}}{n!}\int_0^x (x-s)^n e^{is}ds\right| \leq \left|\frac{2}{(n-1)!}\int_0^x (x-s)^{n-1}ds\right| \leq \frac{2|x|^n}{n!} \tag{3}$$

Combinando le relazioni (1), (2) e (3) si dimostra il lemma. [7]

□

Ora possiamo procedere al teorema di Berry-Esseen.

Teorema 2.6 (Berry-Esseen)**.** *Siano $X_1, X_2, ..., X_n$ v.a. iid con $E[X_i] = 0, E[X_i^2] = \sigma^2$, e $E[|X_i|^3] = \rho < \infty$. Se $F_n(x)$ è la FdD di $S_n = \frac{X_1+...+X_n}{\sigma\sqrt{n}}$ e $\Phi(x)$ è la FdD di una gaussiana standard, allora*

$$|F_n(x) - \Phi(x)| \leq \frac{3\rho}{\sigma^3\sqrt{n}}$$

Dimostrazione. La dimostrazione consiste nel dimostrare due lemmi per poi passare al risultato del teorema vero e proprio.

Lemma 3. *Siano F e G FdD con $G'(x) \leq \lambda < \infty$. Sia F_P la FdD della distribuzione di Polya (Definizione 2.3). Sia $\Delta(x) = F(x) - G(x), \eta = \sup|\Delta(x)|, \Delta_P = \Delta * F_P$[1], e $\eta_P = \sup|\Delta_P(x)|$. Allora*

$$\eta_P \geq \frac{\eta}{2} - \frac{12\lambda}{\pi L}$$

Dimostrazione. Δ tende a 0 a $\pm\infty$, G è continua e F è una FdD quindi esiste un x_0 tale che $\Delta(x_0) = \eta$ oppure $\Delta(x_0-) = -\eta$. Assumiamo la prima. poiché $G' \leq \lambda$ e F è non decrescente per le proprietà delle FdD si ha

[1]Operazione di convoluzione definita come $v * w(z) = \int_{\mathbb{R}} v(z-y)w(y)dy$

$$\Delta(x_0 + s) \geq \eta - \lambda s$$

Sia $\delta = \frac{\eta}{2\lambda}$ e $t = x_0 + \delta$, si ottiene

$$\Delta(t - x) \geq \begin{cases} \frac{\eta}{2} + \lambda x & \text{se } |x| \leq \delta \\ -\eta & \text{altrimenti} \end{cases}$$

Per stimare la convoluzione, consideriamo

$$2\int_{\delta}^{\infty} f_P(x)dx \leq 2\int_{\delta}^{\infty} \frac{2}{\pi L x^2}dx = \frac{4}{\pi L \delta}$$

dove f_P è la densità di Polya. Osservando l'intervallo $(-\delta, \delta)$ e il suo complementare si nota che f_P è pari quindi

$$\int_{-\delta}^{\delta} x f_P(x)dx = 0$$

e infine abbiamo

$$\eta_P \geq \Delta_P(t) \geq \frac{\eta}{2}\left(1 - \frac{4}{\pi L \delta}\right) - \eta\frac{4}{\pi L \delta} = \frac{\eta}{2} - \frac{6\eta}{\pi L \delta} = \frac{\eta}{2} - \frac{12\lambda}{\pi L}$$

che dimostra il lemma. □

Questo risultato ci serve per dimostrare che non vi è troppa differenza tra prima e dopo la convoluzione.

Lemma 4. *Siano K_1 e K_2 densità di v.a. con media 0 e le cui CF κ_1 e κ_2 sono integrabili, allora*

$$K_1(x) - K_2(x) = \frac{1}{2\pi}\int -e^{-itx}\frac{\kappa_1(t) - \kappa_2(t)}{it}dt$$

Dimostrazione. poiché i κ_l sono integrabili, per il Teorema 2.5, la densità $k_l(x)$ vale

$$k_l(y) = \frac{1}{2\pi}\int e^{-ity}\kappa_l(t)dt$$

Possiamo sottrarre l'espressione per $l = 2$ quella con $l = 1$, integrare da a a x e porre $\Delta K = K_1 - K_2$ ottenendo

$$k_1(y) - k_2(y) = \frac{1}{2\pi} \int e^{-ity}(\kappa_1(t) - \kappa_2(t))dt$$
$$\int_a^x (k_1(y) - k_2(y))dy = \frac{1}{2\pi} \int_a^x \int e^{-ity}(\kappa_1(t) - \kappa_2(t))dtdy$$
$$K_1(x) - K_1(a) - K_2(x) + K_2(a) = \frac{1}{2\pi} \int \left(\int_a^x e^{-ity} dy \right) (\kappa_1(t) - \kappa_2(t))dt$$
$$\Delta K(x) - \Delta K(a) = \frac{1}{2\pi} \int (e^{-ita} - e^{-itx}) \frac{\kappa_1(t) - \kappa_2(t)}{it} dt$$

dove Fubini è ammesso poiché le κ_l sono integrabili e stiamo considerando un intervallo limitato di y. Facendo tendere a ad ∞ e usando il lemma di Riemann-Lebesgue (Lemma 1) si dimostra il risultato. □

Dimostrando questo lemma abbiamo ottenuto un legame tra la differenza di due FdD e la differenza delle loro CF, quindi se possiamo minorare le CF possiamo minorare la differenza tra va, cioè l'obbiettivo della dimostrazione.

Siano φ_F e φ_G le CF di F e G. Consideriamo $F_P = F * F_P$ e $G_P = G * F_P$ e applichiamo ad esse il Lemma 4

$$\begin{aligned} |F_P(x) - G_P(x)| &= \frac{1}{2\pi} \int |e^{-itx}| \frac{|\varphi_F(t)\varphi_P(t) - \varphi_G(t)\varphi_P(t)|}{|it|} dt \\ &\leq \frac{1}{2\pi} \int |\varphi_F(t)\varphi_P(t) - \varphi_G(t)\varphi_P(t)| \frac{dt}{|t|} \\ &\leq \frac{1}{2\pi} \int_{-L}^{L} |\varphi_F(t) - \varphi_G(t)| \frac{dt}{|t|} \end{aligned}$$

poiché la CF di una convoluzione è il prodotto delle CF, $|e^{-itx}| \leq 1$ e $|\varphi_P(t)| \leq 1$. Siamo arrivati ad una minoranza tra la differenza di v.a. e un integrale, cioè una quantità finita. Applicando il Lemma 3 si ottiene

$$|F(x) - G(x)| \leq \frac{1}{\pi} \int_{-L}^{L} |\varphi_F(\theta) - \varphi_G(\theta)| \frac{d\theta}{|\theta|} + \frac{24\lambda}{\pi L}$$

dove $\lambda = \sup_x G'(x)$. Ponendo $F = F_n$ e $G = \Phi$, ovvero le funzioni che siamo interessati a studiare, si ha

$$|F_n(x) - \Phi(x)| \leq \frac{1}{\pi} \int_{-L}^{L} \left| \varphi^n \left(\frac{\theta}{\sqrt{n}} \right) - \Psi(\theta) \right| \frac{d\theta}{|\theta|} + \frac{24\lambda}{\pi L} \tag{4}$$

dove $\Psi(\theta)$ è la CF di $\mathcal{N}(0,1)$. Il resto della dimostrazione consiste in passaggi algebrici per trovare una stima per la parte destra della disuguaglianza. Consideriamo $n \geq 10$. Per fare ciò consideriamo che

$$\sup_x G'(x) = G'(0) = (2\pi)^{-\frac{1}{2}} < \frac{2}{5} \tag{5}$$

Per il primo termine invece, notiamo che se $|\alpha|, |\beta| \leq \gamma$ allora

$$|\alpha^n - \beta^n| \leq \sum_{m=0}^{n-1} |\alpha^{n-m}\beta^m - \alpha^{n-m-1}\beta^{m+1}| \leq n|\alpha - \beta|\gamma^{n-1} \tag{6}$$

Ora usando il Lemma 2 fino ad $n = 2$, ponendo $x = tX$ e applicando il valore atteso si ottiene

$$\left|\varphi(t) - 1 + \frac{t^2}{2}\right| \leq \frac{\rho|t|^3}{6} \tag{7}$$

poiché $E[e^{itX}] = \varphi(t), E[X] = 0, E[X^2] = 1$ e $E[X^3] = \rho$. Inoltre $\frac{\rho|t|^3}{6} \leq |t|^2$. Quindi se $t^2 \leq 2$

$$|\varphi(t)| \leq 1 - \frac{t^2}{2} + \frac{\rho|t|^3}{6} \tag{8}$$

Sia $L = \frac{4\sqrt{n}}{3\rho}$. Se $|\theta| \leq L$ allora per (8) e il fatto che $\frac{\rho|\theta|}{\sqrt{n}} \leq \frac{4}{3}$ si ha

$$\begin{aligned}\left|\varphi\left(\frac{\theta}{\sqrt{n}}\right)\right| &\leq 1 - \frac{\theta^2}{2n} + \frac{\rho|\theta|^3}{6n^{\frac{3}{2}}} \\ &\leq 1 - \frac{\theta^2}{2n} + \frac{4}{3}\frac{\theta^2}{6n} = 1 - \frac{5\theta^2}{18n} \\ &\leq e^{-\frac{5\theta^2}{18n}}\end{aligned}$$

poiché $1 - x \leq e^{-x}$. Ora applichiamo (6) con:

$$\alpha = \varphi\left(\frac{\theta}{\sqrt{n}}\right) \qquad \beta = e^{-\frac{\theta^2}{2n}} \qquad \gamma = e^{-\frac{5\theta^2}{18n}}$$

poiché $n \geq 10$

$$\gamma^{n-1} \leq e^{-\frac{\theta^2}{4}} \tag{9}$$

visto che $\gamma^{n-1} \downarrow$ se $n \uparrow$. Per l'altra parte di (6) scriviamo

$$n|\alpha - \beta| \leq n\left|\varphi\left(\frac{\theta}{\sqrt{n}}\right) - 1 + \frac{\theta^2}{2n}\right| + n\left|1 - \frac{\theta^2}{2n} - e^{-\frac{\theta^2}{2n}}\right|$$

per stimare un limite del primo membro a destra, notiamo che (7) implica

$$n\left|\varphi\left(\frac{\theta}{\sqrt{n}}\right)-1+\frac{\theta^2}{2n}\right|\leq\frac{\rho|\theta|^3}{6\sqrt{n}}$$

Per il secondo termine invece notiamo che per $0<x<1$ abbiamo una serie alternante con termini decrescenti quindi

$$|e^{-x}-(1-x)|=\left|-\frac{x^2}{2!}+\frac{x^3}{3!}-...\right|\leq\frac{x^2}{2}$$

Prendendo $x=\frac{\theta^2}{2n}$ e considerando che $|\theta|\leq L\leq\sqrt{2n}$, e quindi $\theta^2\leq 2n$ si ottiene

$$n\left|1-\frac{\theta^2}{2n}-e^{-\frac{\theta^2}{2n}}\right|\leq\frac{\theta^4}{8n}$$

combinando i due risultati ottenuti

$$n|\alpha-\beta|\leq\frac{\rho|\theta|^3}{6\sqrt{n}}+\frac{\theta^4}{8n}\tag{10}$$

Mettendo insieme (10), (9) e (6) si giunge a

$$\begin{aligned}\left|\varphi^n\left(\frac{\theta}{\sqrt{n}}\right)-e^{-\frac{\theta^2}{2}}\right|&\leq\left(\frac{\rho|\theta|^3}{6\sqrt{n}}+\frac{\theta^4}{8n}\right)e^{-\frac{\theta^2}{4}}\\ \frac{1}{|\theta|}\left|\varphi^n\left(\frac{\theta}{\sqrt{n}}\right)-e^{-\frac{\theta^2}{2}}\right|&\leq\left(\frac{\rho\theta^2}{6\sqrt{n}}+\frac{|\theta|^3}{8n}\right)e^{-\frac{\theta^2}{4}}\\ &\leq\frac{1}{L}\left(\frac{2\theta^2}{9}+\frac{|\theta|^3}{18}\right)e^{-\frac{\theta^2}{4}}\end{aligned}$$

visto che $\frac{\rho}{\sqrt{n}}=\frac{4}{3L}$ e $\frac{1}{n}=\frac{1}{\sqrt{n}}\frac{1}{\sqrt{n}}\leq\frac{4}{3L}\frac{1}{3}$ dato che $\rho\geq 1$ e $n\geq 10$. Usando quest'ultimo risultato e (5) nell'equazione (4) restituisce

$$\pi L|F_n(x)-\Phi(x)|\leq\int e^{-\frac{\theta^2}{4}}\left(\frac{2\theta^2}{9}+\frac{|\theta|^3}{18}\right)d\theta+9.6$$

Dato che $L=\frac{4\sqrt{n}}{3\rho}$, notiamo che l'ultimo risultato è nella forma

$$|F_n(x)-\Phi(x)|\leq\frac{C\rho}{\sqrt{n}}$$

con

$$C=\frac{3}{4\pi}\left(\int e^{-\frac{\theta^2}{4}}\left(\frac{2\theta^2}{9}+\frac{|\theta|^3}{18}\right)d\theta+9.6\right)$$

Dalla distribuzione normale sappiamo che

$$\int \frac{1}{\sqrt{(2\pi a)}} x^2 e^{-\frac{x^2}{2a}} dx = a$$

scrivendo $x^3 = 2x^2 \frac{x}{2}$ e integrando per parti

$$\begin{aligned} 2\int_0^\infty x^3 e^{-\frac{x^2}{4}} dx &= 2\int_0^\infty 4x e^{-\frac{x^2}{4}} dx \\ &= -16 e^{-\frac{x^2}{4}} \Big|_0^\infty = 16 \end{aligned}$$

che infine ci da

$$|F_n(x) - \Phi(x)| \leq \frac{1}{\pi}\frac{3}{4}\left(\frac{2}{9}2\sqrt{4\pi} + \frac{16}{18} + 9.6\right)\frac{\rho}{\sqrt{n}} < 3\frac{\rho}{\sqrt{n}}$$

Concludendo la dimostrazione. [7]

□

3 Generazione di numeri casuali

Un *random number generator* (RNG) è un dispositivo o programma capace di generare un stringa di numeri o simboli che non può essere predetta con probabilità migliore della scelta casuale. Come abbiamo visto la generazione di numeri casuali è un aspetto importante del metodo MC, come per molteplici applicazioni, tra cui la crittografia, i video giochi e i giochi d'azzardo nei casinò. In questa sezione analizzo i metodi usati al giorno d'oggi per ottenere numeri casuali e pseudo-casuali. [11]

3.1 Hardware random number generator

La generazione di numeri *veramente* casuali avviene esclusivamente tramite l'utilizzo di fenomeni fisici troppo complessi per essere modellati matematicamente. I generatori che implementano questa metodologia vengono denominati *Hardware random number generator* (HRNG) e i più utilizzati si basano su fenomeni microscopici a loro volta suddivisi in due categorie:
fenomeni termici

- Rumore termico prodotto da un resistore.
- Rumore generato dall'effetto Zener.

- Rumore atmosferico.

fenomeni quantistici

- Fotoni che passano attraverso uno specchio semi-trasparente.
- Rumore shot (rappresentato come processo di Poisson).
- Decadimento nucleare.
- Effetto tunnel.

Tuttavia tali implementazioni presentano alcune problematiche: *a)* in generale, una volta creato un modello matematico per il fenomeno, esso perde la casualità; *b)* alcuni dei fenomeni fisici, in particolare quelli termici, possono essere vulnerabili ad attacchi fisici, ad esempio, la riduzione della temperatura dell'ambiente in cui avviene il fenomeno stesso, riducendone l'entropia; *c)* per alcuni fenomeni è possibile stimare l'entropia.
Ciò premesso, a tutt'oggi i migliori RNG restano gli *hardware random number generator*. [1][4][6][20][24]

3.2 Pseudorandom number generator

Per contro gli RNG più usati e più facilmente implementabili, sono quelli denominati *Pseudorandom number generator* (PRNG), in quanto costituiti solo da software. Tali generatori producono stringhe che sembrano casuali, ma in realtà sono calcolate tramite un algoritmo. La stringa risultante viene generata partendo da un valore specifico chiamato *seed* e si osserva la pseudo casualità quando partendo dallo stesso *seed* si ottengono gli stessi numeri. Inoltre ogni PRNG ad un certo punto inizierà a ripetere la sequenza generata; viene denominato periodo la quantità di numeri prodotti prima che la sequenza si ripeta.[11][14] Matematicamente un PRNG è:

Definizione 3.1. *Dati P distribuzione di probabilità su $(\mathbb{R}, \mathfrak{B})$, $\mathfrak{F}$ una collezione non vuota di insiemi boreliani $\mathfrak{F} \subseteq \mathfrak{B}$ e $A \subseteq \mathbb{R}$ un insieme non vuoto. Allora possiamo chiamare $f : \mathbb{N}^+ \to \mathbb{R}$ un PRNG per P dato $\mathfrak{F}$ a valori in A se e solo se:*

- $f(\mathbb{N}^+) \subseteq A$
- $\forall E \in \mathfrak{F} \qquad \forall \varepsilon > 0 \qquad \exists N \in \mathbb{N}^+ \qquad \forall n \geq N, \quad \left| \frac{\#\{i \in \{1,2,\ldots,n\} : f(i) \in E\}}{n} - P(E) \right| < \varepsilon$

dove $\#S$ denota il numero di elementi nell'insieme S.[29]

3.2.1 Middle-square method

Durante la sua ricerca a Los Alamos, Neumann si ritrovò a dover generare numeri casuali per implementare i metodi MC che stava utilizzando. Non avendo a disposizione una grande quantità di memoria sull'ENIAC, su cui memorizzare lunghe stringhe di numeri casuali, inventò il metodo middle-square (Figura 1). Questo funzionava nel modo seguente: partendo da un *seed* generato da un HRNG, dopo averlo elevato al quadrato, si estraevano le cifre al centro del numero come nuovo *seed*.

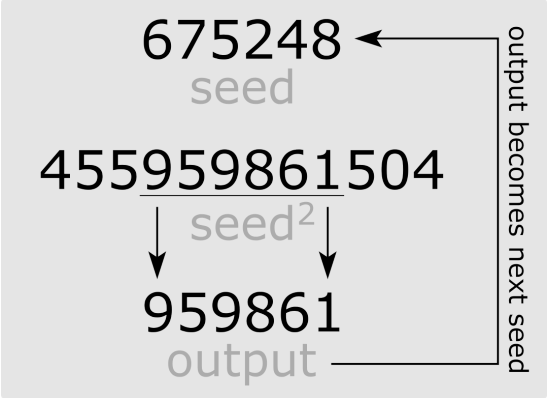

Figura 1: Metodo middle-square.

Questo metodo presenta problemi nel suo utilizzo in caso di importanti applicazioni in quanto *a)* se le cifre centrali sono 0 il generatore rimarrà su 0, *b)* il periodo è troppo breve (meno di 8^n numeri per un *seed* con n cifre), *c)* non si può partire con un valore iniziale dispari.[26]

3.2.2 Linear congruential generator

Una implementazione molto più popolare e utile è la *linear congruential generator* (LCG). Questo tipo di generatori si basa sulla formula

$$X_{n+1} = (aX_n + c) \bmod m$$

L'efficacia del LCG dipende dalla scelta dei parametri.[16] Esistono tre famiglie di scelte:

m primo, $c = 0$ Un LCG di questo tipo si chiama Lehmer RNG, risale al 1951 ed è il primo PRNG creato dopo il metodo middle-square. Il periodo è uguale a $m - 1$ se a è un elemento primitivo degli interi modulo n[2][21]

[2]per n primo, g è un elemento primitivo (mod n) se $\forall$ intero a co-primo ad n, $\exists k \in \mathbb{N} : g^k \equiv a \pmod{n}$, ovvero $g^k - a$ è divisibile per n

***m* potenza di 2, $c = 0$** Questa formulazione produce un LCG efficiente perché l'operazione di modulo può essere calcolata troncando la rappresentazione binaria. Il periodo può essere al massimo $m/4$ ottenibile se $a \equiv 3 \bmod 8$ oppure $a \equiv 5 \bmod 8$. Il problema più grande è che in rappresentazione binaria i bit meno significanti hanno periodi più corti, infatti il bit meno significativo non cambia, il secondo meno significativo ha periodo 2, il terzo ha periodo 4 e così via.

***c* ≠ 0** In questo caso il periodo è m se e solo se:

- m e c sono co-primi
- $a - 1$ è divisibile per tutti i fattori primi di m
- $a - 1$ è divisible per 4 se lo è anche m

Tuttavia mentre queste condizioni (Teorema di Hull-Dobell) assicurano il periodo massimo, non sono solo sufficienti per creare un buon LCG e altre accortezze saranno necessarie per assicurarsi un buon RNG.[9]

Questi sono alcuni valori dei parametri usati da applicazioni commerciali:

	m	a	c
java.util.Random[13]	2^{48}	25214903917	11
minstd_rand (C++11)[10]	$2^{31} - 1$	48271	0
Microsoft Visual Basic (6 e più vecchio)[19]	2^{24}	1140671485	12820163
Turbo Pascal[27]	2^{32}	134775813	1

3.2.3 Cryptographic PRNG (CPRNG)

I CPRNG sono la tipologia di generatori usata nei sistemi crittografici e sono più complessi e sicuri dei normali PRNG. Sono caratterizzati dal fatto che la conoscenza del *seed* iniziale ha un vantaggio trascurabile nel capire la sequenza generata, o in altre parole non deve esistere un algoritmo in tempo polinomiale che può prevedere il bit $k + 1$, conoscendo i bit fino a k, con probabilità di successo maggiore del 50%.[30]

4 Applicazioni del MC

4.1 Calcolo π

Uno dei primi e più famosi esempi di metodo Monte Carlo è il calcolo del valore di π. Si considera un quadrato $[-1, 1] \times [-1, 1]$ come dominio e un cerchio ad esso inscritto. Sul dominio si considerano distribuzioni uniformi in modo tale che le coordinate

$X, Y \sim U[-1, 1]$. Se ora consideriamo la probabilità che una coordinata sia all'interno del cerchio, si ottiene:

$$P(\text{punto interno al cerchio}) = \frac{\text{area cerchio}}{\text{area quadrato}} = \frac{\iint_{x^2+y^2<1} dxdy}{\iint_{-1<x,y<1} dxdy} = \frac{\pi}{4}$$

Ora, considerando n il numero totale di punti, il numero di punti S all'interno del cerchio è una variabile binomiale:

$$S \sim Bin(n, p) \qquad \text{con } p = P(\text{punto interno al cerchio})$$

Quindi abbiamo

$$\pi \approx \frac{4E(S)}{n}$$

e simulando n punti casuali si ottiene l'$E(S)$ della v.a. binomiale come frazione tra punti interni al cerchio e punti esterni al cerchio.[12] La simulazione sotto, creata con Matlab, mostra il valore di π ottenuto con 100, 10000 e 1000000 punti rispettivamente. La funzione *Rand* di Matlab usa un PRNG chiamato *Mersenne Twister* che produce float lunghi 53 bit con periodo di $2^{19937} - 1$. Tuttavia MATLAB ha disponibile una suite di PRNG tra cui: LFG, Philox e Threefry.[25]

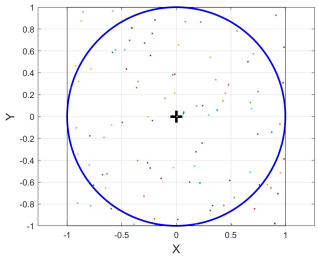

Figura 2: 100 punti. $\widehat{\pi} = 3.0800$.

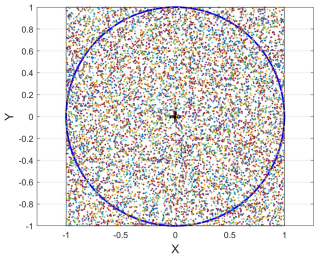

Figura 3: 10000 punti. $\widehat{\pi} = 3.1472$.

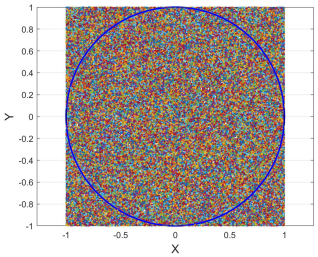

Figura 4: 1000000 punti. $\widehat{\pi} = 3.1416$.

4.2 Calcolo integrali uni e multi dimensionali

Un altro uso classico del MC è il calcolo di integrali. Qui riporto un veloce esempio per il calcolo di integrali uni dimensionali e poi l'adattamento a integrali di qualsiasi dimensione.[8]

Consideriamo una funzione $f : \mathbb{R} \to \mathbb{R}$ *e un intervallo* $[a, b]$ *su cui la vogliamo integrare. Allora*

$$I = \int_a^b f(x)dx = (b-a)\int_a^b f(x)\frac{1}{b-a}dx = (b-a)E(f(X))$$

con $X \sim U([a,b])$ *e per definizione di valore atteso di una v.a. uniforme. Quindi*

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

Esempio 1

Il seguente è il codice Python con cui implementare il metodo per il calcolo dell'integrale della funzione $f(x) = \cos(e^{\sin x})$ nell'intervallo $[5, 10]$.

```
# metodo MC per integrali unidimensionali su un intervallo [a,b]

import random as rd
import numpy as np

# Qui definiamo la funzione da integrare
def func(x):
  return np.cos(np.exp(np.sin(x)))

#Qui definiamo il metodo MC
def monte(n, a, b):
  su = 0
  i=0

  while (i<n):
    su += func(rd.uniform(a,b)) #generazione dei sample
    i+=1

  return ((su*(b-a))/n)
```

Il rislutato calcolato algebricamente è $I = 0.387$. Calcolando tramite MC si ottiene:

- Per $n = 100$, si ottiene $I = 0.313$. $|Errore| = 0.074$.
- Per $n = 10000$ si ottiene $I = 0.413$. $|Errore| = 0.026$.
- Per $n = 1000000$, si ottiene $I = 0.386$. $|Errore| = 0.001$.

Per gli integrali a più dimensioni si devono fare un po' di modifiche ma il funzionamento è essenzialmente uguale.

Consideriamo una funzione $f : \mathbb{R}^n \to \mathbb{R}$ e un dominio Ω su cui vogliamo integrare. Allora

$$I = \int_\Omega f(\overline{v})d\overline{v} = |\Omega| \int_\Omega f(\overline{v})\frac{1}{|\Omega|}d\overline{v} = |\Omega|E(f(\boldsymbol{X}))$$

dove $\overline{v} = (v_0, v_1, ..., v_n)$, $|\Omega|$ = misura di Ω e $\boldsymbol{X} = (X_0, X_1, ..., X_n)$ con $X_i \sim U$ definita sull'appropriato sottoinsieme. Quindi

$$I \approx \frac{|\Omega|}{n}\sum_{i=1}^{n} f(\overline{X}_i)$$

Esempio 2

Il codice Python per l'implementazione del metodo Monte Carlo per il calcolo dell'integrale della funzione $f(x, y, z) = xyz+(x^2+y^2)z$ su $\Omega = \{(x, y, z) : x^2+y^2 < 4, 0 < z < 3\}$ è riportato di seguito.

```
#metodo MC per integrali multidimensionali.

import random as rd
import numpy as np

# Qui definiamo la funzione da integrare
def func(x,y,z):
  return (x*y*z + (x*x+y*y)*z)

#Qui definiamo il metodo MC
def mul_monte(n, meas, x0, y0, z, r):
  su = 0
  i=0
  while (i<n):
    x = rd.uniform(x0-r,x0+r)    #generazione sample
    y = rd.uniform(y0-r,y0+r)
    if ((x-x0)*(x-x0)+(y-y0)*(y-y0) <= r*r):  #per fare in modo che
                                              #sia all'interno del domninio
    su += func(x,y,rd.uniform(0,z))
    i+=1

  return ((su*(meas))/n)
```

Il rislutato calcolato algebricamente è $I = 36\pi = 113.097$. Calcolando tramite MC si ottiene:

- Per $n = 100$, si ottiene $I = 106.491$. $|Errore| = 6.606$.
- Per $n = 10000$ si ottiene $I = 114.949$. $|Errore| = 1.852$.
- Per $n = 1000000$, si ottiene $I = 113.057$. $|Errore| = 0.040$.

Anche la funzione *random.rand* usa il *Mersenne Twister*.[22]

4.3 Ripiegamento delle proteine

Le proteine sono molecole complesse essenziali per i processi vitali degli esseri viventi. Sono composte da catene di amminoacidi (*struttura primaria*) che si ripiegano su se stesse creando strutture denominate *strutture secondarie*. Le strutture più comuni sono o a forma di spirale chiamata *α-helix* o a forma piatta e larga chiamata *β-sheet* (Figura 5) che a loro volta, ripiegandosi creano forme uniche e incredibilmente complesse, chiamate *strutture terziarie*. Il ripiegamento non è un fenomeno casuale ed è necessario per il funzionamento della specifica proteina, sia perché la forma deve essere adatta a creare legami con altre molecole, sia per motivi fisici ed in particolare per la distribuzione di cariche all'interno della molecola.

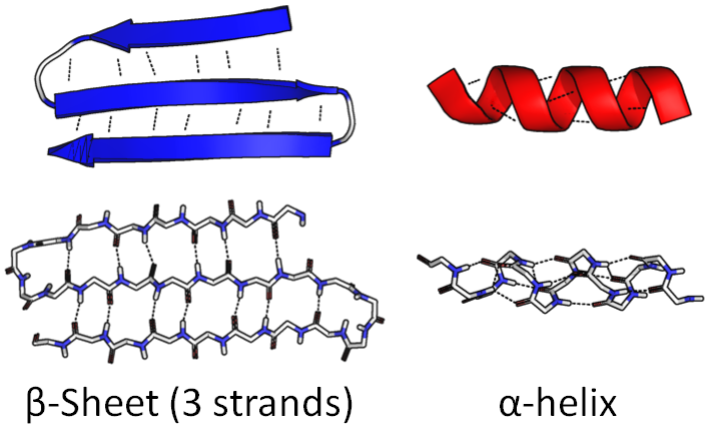

Figura 5: Le due *strutture secondarie* più comuni.

Le informazioni sulla *struttura primaria* sono contenute nel DNA, mentre non vi è modo diretto e semplice di capire la *struttura secondaria* e *struttura terziaria*, cosa essenziale per capire la funzione della proteina. L'individuazione di queste strutture viene denominata "problema del ripiegamento delle proteine" ed esistono due metodologie per affrontarlo: misura e previsione. L'approccio tramite la "misura" prevede l'individuazione della struttura con metodi sperimentali in osservazione diretta. Per contro l'approccio tramite la "previsione" prevede la simulazione della forma che potrebbe assumere la struttura variando particolari parametri.[3][23]

Earl et al[2] hanno sviluppato una simulazione MC per affrontare il problema la cui implementazione necessita 3 elementi: un modello della proteina, un campo di forze e un RNG.

- **Modello della proteina** Per ottimizzare tempo e risorse necessarie per la computazione vengono usati modelli approssimativi chiamati *coarse-grained* al posto del modello *all-atom* (cioè modellando ogni atomo). In particolare nel testo citato viene usato il modello *coarse-grained CABS* in cui catene di atomi, presenti nella molecola, che partecipano agli stessi processi sono semplificati con singoli pseudo-atomi (Figura 6). Inoltre dal modello *CABS*, pur semplificato, si è comunque in grado di risalire ad un modello *all-atom* tramite interpolazione statistica, caratteristica che lo rende un modello particolarmente potente.

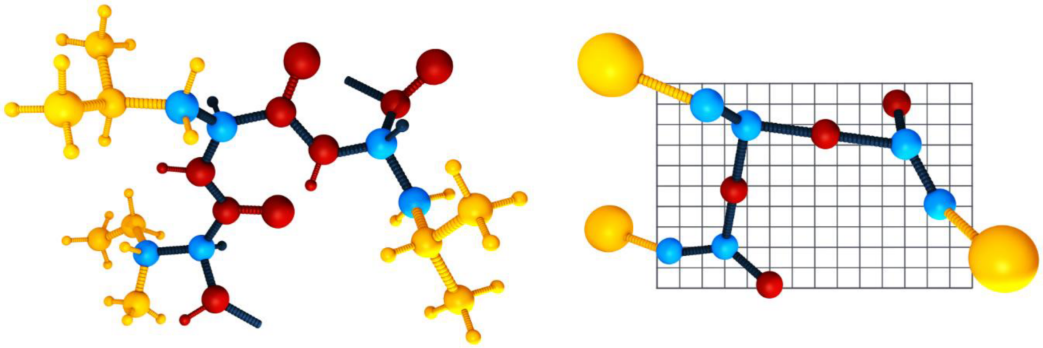

Figura 6: A sinistra il modello *all-atom*, a destra modello *CABS*. Atomi dello stesso colore svolgono le stesse funzioni.

- **Campo di forze** Per definire l'energia potenziale di ogni configurazione della molecola è necessario un campo di forze definito su di essa. Questo campo può essere basato sulle relazioni fisiche della specifica molecola, per esempio la sua geometria o le forze coloumbiane e i fenomeni quantistici all'interno. In alternativa il campo può essere statistico cioè ricavato dalle regolarità statistiche delle strutture proteiche già modellate. Un semplice esempio si può ricavare usando il teorema di Bayes. L'energia del sistema si definisce:

$$E = \ln(P(C \mid X, A)) = \ln\left(\frac{P(X, A \mid C)}{P(X, A)}\right) + c$$

Dove X è un vettore di caratteristiche strutturali (distanze, contatti, angoli, etc.), A è la catena di amminoacidi e $\ln(P(C \mid X, A))$ è la probabilità che la conformazione sia "corretta" dati dei particolari valori per X e A. Per il teorema di Bayes

questa è uguale alla distribuzione di X, A sulle proteine già modellate divisa per una distribuzione a priori di proteine con X, A. Il testo citato usa un modello del secondo tipo.

- **RNG** Per l'implementazione del metodo MC è necessaria la generazione di numeri casuali. Tipicamente viene usato un PRNG semplice.[15]

La simulazione effettiva consiste in cicli innestati di tre processi (Figura 7):

- **Passi del MC**
  I passi del MC consistono nella generazione di valori casuali per controllare il tipo, la direzione e l'intensità di movimenti casuali di pezzi della proteina (anche il pezzo è scelto casualmente).

- **Cicli del MC**
  I cicli MC sono blocchi composti da passi MC, di default 50.

- **Cicli di annealing**
  Infine i cicli di annealing sono il blocco in cui viene calcolata l'energia del sistema; l'idea generale è scegliere la configurazione con energia minore. Si confronta l'energia della configurazione ottenuta con quella precedente, poi si adopera il processo di annealing, necessario per evitare minimi locali di energia, e si riparte dall'inizio del processo con una nuova configurazione. Il processo di annealing consiste nello scegliere probabilisticamente se cambiare configurazione usando l'energia del precedente e successivo stato per calcolare le probabilità.

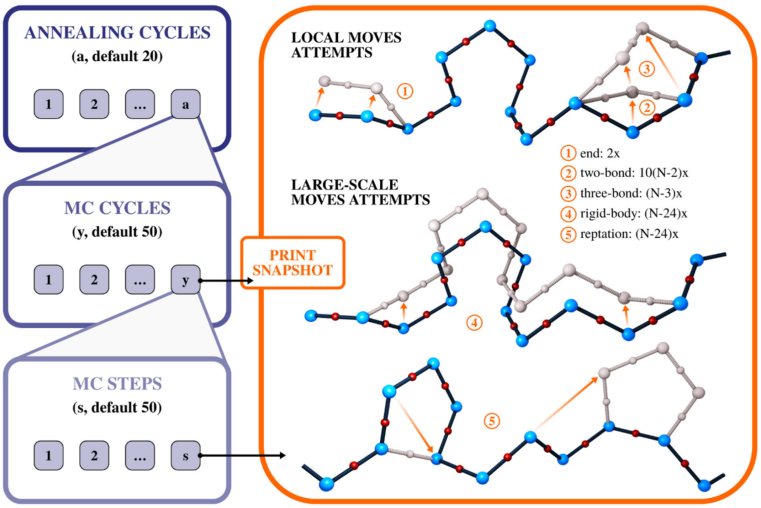

Figura 7: A sinistra il processo usato nella simulazione delle proteine, a destra i possibili movimenti/ripiegamenti della proteina.

L'uso del metodo MC ha dimostrato risultati accurati e promettenti, pur dovendo competere, al giorno d'oggi, con approcci di machine-learning con reti neurali.[2][3]

5 Conclusione

Il metodo MC, a distanza di un secolo, mantiene una forte valenza quale potente strumento per diverse implementazioni in quanto caratterizzato dalla sua relativa semplicità e flessibilità. La tendenza a richiedere poche condizioni a priori per il suo funzionamento insieme alla sua efficienza computazionale gli conferiscono una versatilità capace di renderlo utile al calcolo di una miriade di risultati: dalla cromodinamica quantistica agli schemi di processo per la lavorazione dei minerali, dalla costruzione di alberi filogenetici tramite inferenza bayesiana alla valutazione del rischio associato a derivate finanziarie, dal corretto rendering della luce in un ambiente virtuale tridimensionale all'individuazione di persone a rischio in operazioni di ricerca e soccorso tramite analisi probabilistiche, le conseguenze del metodo MC permeano la società moderna. Inoltre è di particolare interesse il modo in cui unisce il mondo teorico e assiomatico della matematica deterministica a quello più reale e sperimentale della matematica stocastica mettendo in luce come la divisione tra queste discipline sia spesso più una semplificazione umana piuttosto che una realtà. Lo studio di questi metodi continua in una grande quantità di discipline e nella risoluzione di molti problemi risulta ancora la migliore, se non l'unica, scelta.

Riferimenti bibliografici

[1] R. Bernardo-Gavito, I. E. Bagci, J. Roberts, J. Sexton, B. Astbury, H. Shokeir, T. McGrath, Y. J. Noori, C. S. Woodhead, M. Missous, U. Roedig, and R. J. Young. Extracting random numbers from quantum tunnelling through a single diode. *Scientific Reports*, 7(1):17879, Dec 2017. ISSN 2045-2322. doi: 10.1038/s41598-017-18161-9. URL `https://doi.org/10.1038/s41598-017-18161-9`.

[2] M. P. Ciemny, A. E. Badaczewska-Dawid, M. Pikuzinska, A. Kolinski, and S. Kmiecik. Modeling of disordered protein structures using monte carlo simulations and knowledge-based statistical force fields. *Int. J. Mol. Sci.*, 20(3):606, Jan. 2019.

[3] J. Crawford. What is the "protein folding problem"? a brief explanation, 2020. URL `https://rootsofprogress.org/alphafold-protein-folding-explainer`.

[4] R. Davies. Hardware random number generators, October 2000. URL `http://www.robertnz.net/hwrng.htm`.

[5] L. Devroye. Methods for generating random variates with polya characteristic functions. *Statistics Probability Letters*, 2(5):257–261, 1984. ISSN 0167-7152. doi: https://doi.org/10.1016/0167-7152(84)90061-0.

[6] R. Dube. *Hardware-based Computer Security Techniques to Defeat Hackers: From Biometrics to Quantum Cryptography*. Wiley, 2008. ISBN 9780470425473.

[7] R. Durrett. *Probability: Theory and Examples*. Cambridge University Press, January 2019. ISBN 9781139491136.

[8] L. Finesso. *Lezioni di probabilità*. Libreria Progetto, 2017. ISBN 9788896477915.

[9] T. Hull and A. Dobell. Random number generators. *SIAM Review*, 4(3):230–254, July 1962.

[10] ISO. Iso/iec 14882:2011. 2011.

[11] F. James. A review of pseudorandom number generators. *Computer Physics Communications*, 60(3):329–344, 1990. ISSN 0010-4655. doi: https://doi.org/10.1016/0010-4655(90)90032-V.

[12] A. M. Johansen and L. Evers. Monte carlo methods, lecture notes, November 2007.

[13] katleman. view src/share/classes/java/util/random.java @ 9107:687fd7c7986d, 2014. URL `http://hg.openjdk.java.net/jdk8/jdk8/jdk/file/tip/src/share/classes/java/util/Random.java`.

[14] Khan Academy. Pseudorandom number generators, 2012. URL https://www.khanacademy.org/computing/computer-science/cryptography/crypt/v/random-vs-pseudorandom-number-generators.

[15] S. Kmiecik, D. Gront, M. Kolinski, L. Wieteska, A. E. Dawid, and A. Kolinski. Coarse-grained protein models and their applications. *Chem. Rev.*, 116(14):7898–7936, July 2016.

[16] D. E. Knuth. *Art of Computer Programming, Volume 2: Seminumerical Algorithms*. Addison-Wesley Professional, 1997. ISBN 9780201896848.

[17] D. P. Kroese, T. Brereton, T. Taimre, and Z. I. Botev. Why the monte carlo method is soimportant today. *WIREs Comput Stat*, 6:386–392, 2014. doi: 10.1002/wics.1314.

[18] N. Metropolis. The beginning of the monte carlo method. *Los Alamos Science Special Issue*, 15:125–130, 1987.

[19] Microsoft Corporation. Info: How visual basic generates pseudo-random numbers for the rnd function, 2004. URL https://www.betaarchive.com/wiki/index.php/Microsoft_KB_Archive/231847.

[20] J. Niemczuk. Shot noise-based quantum random number generator. In E. Diamanti, S. Ducci, N. Treps, and S. Whitlock, editors, *Quantum Technologies 2020*, volume 11347, page 1134717. International Society for Optics and Photonics, SPIE, 2020. doi: 10.1117/12.2554898. URL https://doi.org/10.1117/12.2554898.

[21] W. H. Payne, J. R. Rabung, and T. P. Bogyo. Coding the lehmer pseudo-random number generator. *Commun. ACM*, 12(2):85–86, feb 1969. ISSN 0001-0782. doi: 10.1145/362848.362860. URL https://doi.org/10.1145/362848.362860.

[22] Python Software Foundation. random — generate pseudo-random numbers, 2022. URL https://docs.python.org/3/library/random.html.

[23] I. Rehman, M. Farooq, and S. Botelho. *Biochemistry, Secondary Protein Structure*. StatPearls Publishing, 2021.

[24] C. Shaw. Hardware random number generators. *Cerberus Security Laboratories Ltd*, 2020.

[25] The MathWorks. Randstream.list, 2022. URL https://it.mathworks.com/help/matlab/ref/randstream.randstream.list.html.

[26] J. von Neumann. Various techniques used in connection with random digits. *Monte Carlo Method, National Bureau of Standards Applied Mathematics Series*, 12:36–38, 1951.

[27] wiki.freepascal.org. Delphi compatible lcg random, 2019. URL `https://wiki.freepascal.org/Delphi\_compatible\_LCG\_Random`.

[28] Wikipedia. Riemann-lebesgue lemma, 2021. URL `https://en.wikipedia.org/wiki/Riemann\%E2\%80\%93Lebesgue\_lemma`.

[29] Wikipedia. Pseudorandom number generator, 2022. URL `https://en.wikipedia.org/wiki/Pseudorandom\_number\_generator`.

[30] A. C. Yao. Theory and application of trapdoor functions. In *23rd Annual Symposium on Foundations of Computer Science (sfcs 1982)*, pages 80–91, 1982. doi: 10.1109/SFCS.1982.45.