Relazione 2: Generatori di numeri Pseudocasuali

Antonio Michele Miti

16 gennaio 2010

Sommario

Lo scopo di questo articolo è dare una soluzione numerica del problema matematico detto *Ago di buffon*. Dopo averne realizzato un modello computazionale si intende determinare una legge che descriva l'andamento dell'errore da attribuire ad una eventuale singola misura della *probabilità di successo*. Un'analisi teorica del problema porta ad individuare 3 fattori determinanti per stabilire una stima dell'incertezza della misura: errore sistematico, errore statistico, errore computazionale. Il primo errore origina della natura della legge che si intende calcolare sperimentalmente, ovvero la distribuzione di probabilità che è un'astrazione matematica definita esatta solo su infiniti lanci. L'errore statistico dipende dalla casualità delle singole variabili (θ, x) che determinano il successo del lancio. Infine l'errore computazionale è quello che emerge dalla non perfetta uniformità della distribuzione dei numeri generati dal generatore pseudocasuale *rand* delle librerie *C++*.

1 Introduzione

Il problema dell Ago di Buffon : Si consideri un pavimento, suddiviso da un grande numero di righe paralle equispaziate da una distanza T, si immagini ora di lasciar cadere su di esso, in modo completamente casuale, un sottile ago di lunghezza R. Scopo del problema è determinare con quale probabilità l'ago, una volta caduto, si trovi a contatto con almeno una delle righe sul piano.

1.1 Analisi Teorica

In termini matematici i gradi di libertà che descrivono il sistema sono solo 2: le variabili (θ, x) che fissano la posizione del ago sul piano. La variabile θ fissa l'inclinzione dell ago rispetto alle righe parallele, mentre la variabile x esprime la distanza del centro di massa dell ago dalla riga più vicina.

Un lancio è considerato un successo quando l'ago, una volta fermo sul piano, tocca almeno una riga parallela. Da semplici considerazioni geometriche si ottiene che la condizione di successo è :

$$\text{condizione di successo :} \qquad x \leq \frac{R}{2}\sin(\theta) \tag{1}$$

Considerando la caduta del ago come un evento perfettamente casuale le coordinate di configurazione del sistema possono assumere equiprobabilmente qualsiasi valore compreso negli intervalli

$$0 \leq x \leq \frac{T}{2} \qquad \text{e} \qquad 0 \leq \theta \leq \frac{\pi}{2} \tag{2}$$

Più precisamente questa condizione equivale ad affermare che (θ, x) sono una coppia di variabili aleatorie con funzione densità di probabilità uniforme. Ciò significa che

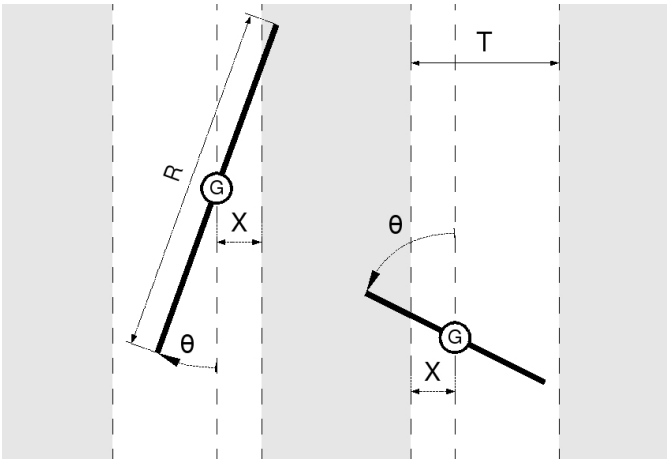


Figura 1: Parametri dell ago di buffon

le funzioni densità di probabilità delle 2 variabili dovranno essere delle costanti che intragrate sugli intervalli (2) diano la probabilità totale 1.

$$1 = \int_R f(t)dt = k\int_I dt = K(x1 - x0) \qquad \Longrightarrow$$

$$g(\theta) = \frac{2}{\pi} \qquad h(x) = \frac{2}{T}$$

L'indipendenza delle due variabile aleatorie permette di fattorizzare la funzione densità di probabilità $f(\theta, x)$ come il prodotto delle due singole funzioni.

$$f(\theta, x) = g(\theta)h(x) = \frac{4}{T\pi} \tag{3}$$

A questo punto è sufficiente integrare la funzione densità di probabilità sul dominio D, in cui le variabili aleatorie danno un successo, per ottenerè la probabilità che l'ago tocchi almeno una riga sul pavimento.

$$P_{successo} = \int_0^{\frac{\pi}{2}} \int_0^{\frac{R}{2}\sin(\theta)} \frac{4}{T\pi} dx d\theta = \frac{2R}{T\pi} \tag{4}$$

1.2 Simulazione del esperimento in C++

Il modello per la simulazione è molto semplice: si usa la funzione rand() delle librerie C++ per estrarre 2 numeri casuali negli intervalli (2), se la coppia di valori soddisfa la condizione (1) asi ha un successo.

Ripetendo la simulazione di un lancio un numero N di volte e ottenendo complessivamente un numero k di successi, si ottiene una stima della probabilità di successo:

$$P_{successo}(N) = \frac{k}{N} \tag{5}$$

Risulta la seguente funzione di C++;

```
int cont=0; //contatore delle volte che l'ago tocca una fessura
for(int i=0; i<N; i++)  {
        double x = ((double)rand()/RAND_MAX)*T/2;
        double s = ((double)rand()/RAND_MAX)*(M_PI/2);
        if(x<=(R/2)*sin(s))cont++ ;} // condizione di successo
return probabilita=((double)cont/N);
```

A questo punto bisogna stabilire con che incertezza un singolo *esperimento* di N lanci determina la stima della probabilità di successo, si cerca una leggi in funzione del numero dei lanci componenti l'esperimento.

1.3 Errore sistematico

Il concetto di probabilità relativo ad un particolare evento può essere definito come limite di (5) per un numero N di prove tendente all'infinito.

Qualsiasi simulazione computazionale o esperimento empirico invece prevede necessariamente un numero finito di prove, di fatto può solo stimare, con una certa approssimazione, il valore della probabilità.

Considerando una simulazione di N lanci, la casualità delle variabili aleatorie implica che anche il numero k di successi sia casuale (ovviamente con una distribuzione diversa, cambierà ogni volta che si ritenta la simulazione [vedere errore statistico]) e potrà in generale assumere ogni valore intero compreso tra 0 eK . Quindi se :

$$P = \frac{k}{N} \qquad e \qquad P' = \frac{k \pm 1}{N}$$

i vari valori di P risultanti dall'esperimento non potranno assumere valori continui ma tutti i possibili risultati saranno distanziati da un intervallo:

$$\triangle P = \mid P - P' \mid = \frac{1}{N}.$$

Dato che il valore atteso di P è un numero irrazionale (contiene π) sarà necessariamento compreso tra una certa coppia di valori (P, P') che invece, in quanto quozienti di numeri naturali, risulteranno sempre una quantità razionale.

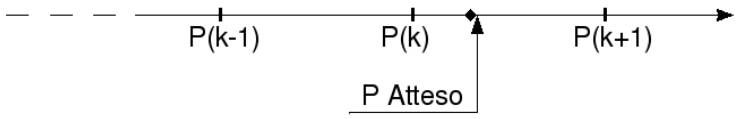


Figura 2: Errore sistematico: con P(K) si intende la probabilità risultante da k successi.

Per cui il valore di P risultante da una simulazione di N prove approssima la probabilità vera con un incertezza:

$$\delta_{sistematico} = \frac{\triangle P}{N} = \frac{1}{2N}. \tag{6}$$

1.4 Errore statistico

Come accennato nella sezione precedente, la casualità delle variabili aleatorie implica che anche il numero k di successi sia casuale. In linea di principio ogni simulazione darà un risultato di P leggermente diversa da quello precedente con una distribuzione gaussiana attorno al valore atteso.

L'errore statistico sarà dunque la *deviazione standard* del risultato di K simulazioni da N lanci ciascuna:

$$\delta_{statistico} = \sqrt{\mid \bar{x^2} - \bar{x}^2 \mid} = ... \tag{7}$$

A questo punto si sarebbe tentati ad utilizzare la media dei risultati degli esperimenti come stima di P, questo è chiaramente sensato ma computazionalmente è identico ad un singolo esperimento di K x N lanci:

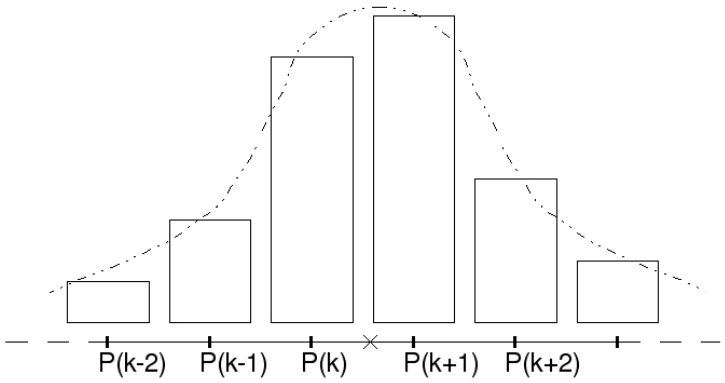


Figura 3: Andamento atteso a causa dell'errore statistico.

$$\bar{P} = \frac{\sum_{i=1}^{K} \frac{successi_i}{N}}{K} = \frac{\sum_{i=1}^{K} successi_i}{NK} \tag{8}$$

In sostanza questo errore esprime, nell'ipotesi di uniformità della funzione di probabilità del generatore rand(), la dispersione del valore ottenuto attorno al valore vero per colpa della casualità dei possibili risultati di un singola simulazione di esperimento.

1.5 Errore computazionale

Questo errore deriva dalla pseudo-casualità del generatore.

Non ci sono motivi di dubitare dell'efficienza della funzione rand() di *C++*, che è già stata immensemente testata, ma in ogni caso ogni algoritmo numerico presenta dei difetti: crea catene di valori interi, periodiche, finite, numerabili; sono deterministici: partendo da un seed si avrà ogni volta la stessa catena; nel complesso possono solo *approssimare* una distribuzione uniforme su un intervallo continuo...

Nello specifico non è possibile fare una stima a priori dell'incertezza causata da questi fattori, il solo concetto di casuale è problematico da definire per una catena finita di valori.

Questo però non impedisce di fare una stima di questa incertezza. Inanzi tutto è evidente che il generatore pseudo-random entra solo nel momento in cui la simulazione chiama una coppia di valori (θ, x), l'imperfezione del generatore implicherà che le distribuzioni di probabilità relative ai 2 parametri siano solo quasi uniformi, ovvero esisteranno due funzioni $(\eta(\theta), \upsilon(x))$ tali per cui le effettive funzioni densità di probabilità per le 2 variabili saranno:

$$g(\theta) = \frac{2}{\pi}(1 + \eta(\theta)) \qquad h(x) = \frac{2}{T}(1 + \upsilon(x))$$

la forma delle funzioni dipenderà dal tipo di generatore mentre il loro modulo sarà legato all'efficienza. A questo punto si potrebbe sfruttare l'indipendenza delle 2 variabili aleatorie e costruire la funzione densità di probabilità totale, il punto è che chiamando successivamente 2 rand() non si fa altro che prendere un elemento e il suo successivo nella stessa catena, data la natura linear congruent math del algoritmo l'indipendenza delle due variabili aleatorie non è garantita.

La conseguenza di tutto questo si esprimerà con una funzione di correzione $\varepsilon * W(\theta, x)$ alla funzione distribuzione di probabilità del problema. In altre parole la distribuzione di probabilità vera risulta:

$$f(\theta, x) = \frac{4}{T\pi}(1 + \varepsilon W(\theta, x)) \tag{9}$$

Per cui la probabilità di avere un successo in questo modello imperfetto sarà:

$$P_{successo}^{simulazione} = \int_0^{\frac{\pi}{2}} d\theta \int_0^{\frac{R}{2}\sin(\theta)} \frac{4}{T\pi}(1+\varepsilon W(\theta,x))dx = P_{successo}^{teorico} - \varepsilon \int\int W(\theta,x)d\theta dx \tag{10}$$

quindi si arriva ad una stima del errore random:

$$\delta_{random} = \mid P_{successo}^{teorico} - P_{successo}^{simulazione} \mid \approx \varepsilon \tag{11}$$

che evidentemente non dovrebbe dipendere dal numero N di lanci della simulazione.

2 Metodologia

Scopo della relazione è dare una stima dell'incertezza sul risultato della simulazione in funzione del numero di lanci di ciascuna simulazione.

Errore Sistematico : L'andamendo dell'errore sistematico è noto a priori $\frac{1}{2N}$.

Errore Statistico : Vengono simulati, al variare del parametro N= numero dei lanci, 100 esperimenti di Ago di Buffon generando un insieme $\{P_i\}$ di 100 risultati in funzione di N. Per ogni set viene calcolata la media e la deviazione standard allo scopo di mostrare graficamente l'andamento dell'errore in funzione di N e di eseguirne un fit tramite gnuplot in modo da ricavare la legge cercata.

Errore Computazionale : Si esegue un'altra analisi sul set di valori ottenuti prima stampando l'andamento del modulo della differenza P(n)-P(atteso) in funzione del numero di lanci fatti nel esperimento.

Sapendo che teoricamente vale:

$$lim_{n\to\infty} P^{sperimentale}(n) = P^{teorico}$$

fittando i dati considerati con una funzione con parametro libero costante del tipo

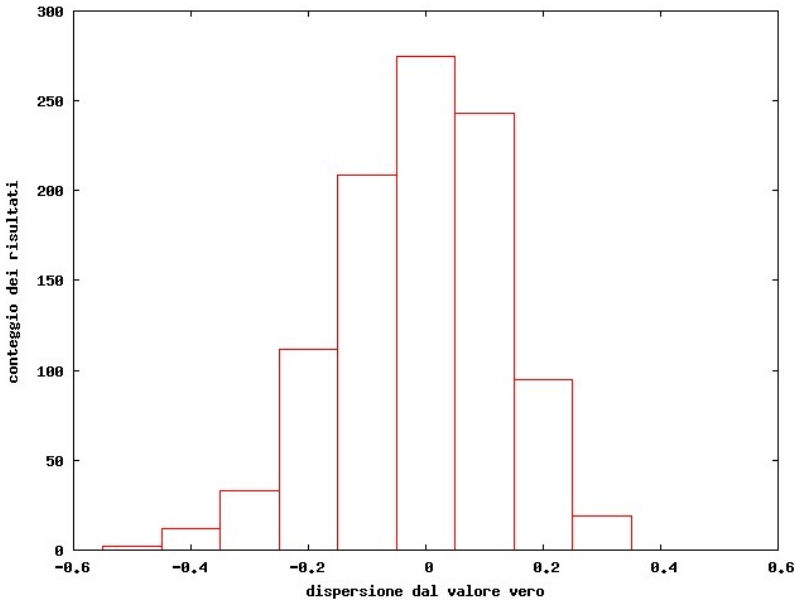
$$f(n) = A + g(n, B, C, D...)$$

dove $(A, B, C, D..)$ sono i parametri liberi del fit e $g(n)$ è una funzione evanescente a $+\infty$, il parametro A sarà la stima dell'errore computazionale causato dal generatore.

3 Risultati

Errore statistico Per evedenziare l'effetto del errore statistico è possibile costruire un istogramma [Figura4] che conti il risultato di un certo numero di simulazioni da 10 lanci. Come si vede dal grafico la distribuzione dei risultati è piccata attorno al valore

Figura 4: Istogramma della dispersione, dal valore atteso, dei risultati di K simulazioni da 10 lanci ciascuna.



vero ma presenta in ogni caso una certa dispersione sui valori. Per questo è necessario prendere la media di questi valori per ottenere una stima accettabile della probabilità cercato.

Simulando, al variare del parametro N= numero dei lanci, 100 esperimenti di Ago di Buffon è possibile costruire il grafico della stima della probabilità in funzione del numero dei lanci :[Figura5].

L'errore rappresentato è l'effetto complessivo degli errori statitistici e sistematici quindi il grafico converge al valore $P^{simulazione}_{stimato}$.

Inoltre è possibile dare una stima dell'errore statistico stampando la deviazione standard di 100 esperimenti al variare di N, numero dei lanci:[Figura6].

A questo punto è possibile fare un fit dei valori, ci si aspetta un andamento del tipo:

$$\delta_{statistico}(N) = AN^B \tag{12}$$

quindi fittando, tramite gnuplot, secondo la funzione *f(x) =AxB*, si ottiene il seguente output:

```
After 265 iterations the fit converged.
final sum of squares of residuals : 1.57567e-08
rel. change during last iteration : -3.2102e-08

degrees of freedom    (FIT_NDF)                        : 97
rms of residuals      (FIT_STDFIT) = sqrt(WSSR/ndf)    : 1.27452e-05
variance of residuals (reduced chisquare) = WSSR/ndf   : 1.6244e-10

Final set of parameters            Asymptotic Standard Error
=======================            ==========================

A               = 0.0211063        +/- 0.0002628    (1.245%)
B               = -0.501652        +/- 0.001838     (0.3664%)
```

Figura 5: Grafico del andamento del valore medio di $P(n)$ con rappresentato la somma degli errori statistico e sistematico in funzione del numero dei lanci. Output di ago_1.c

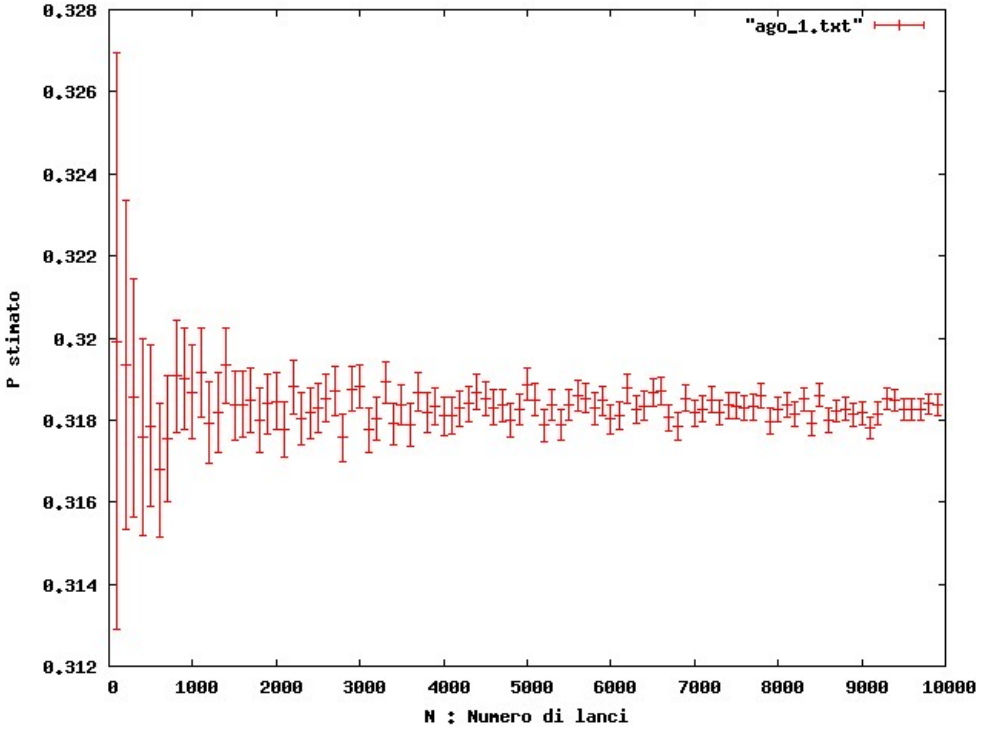
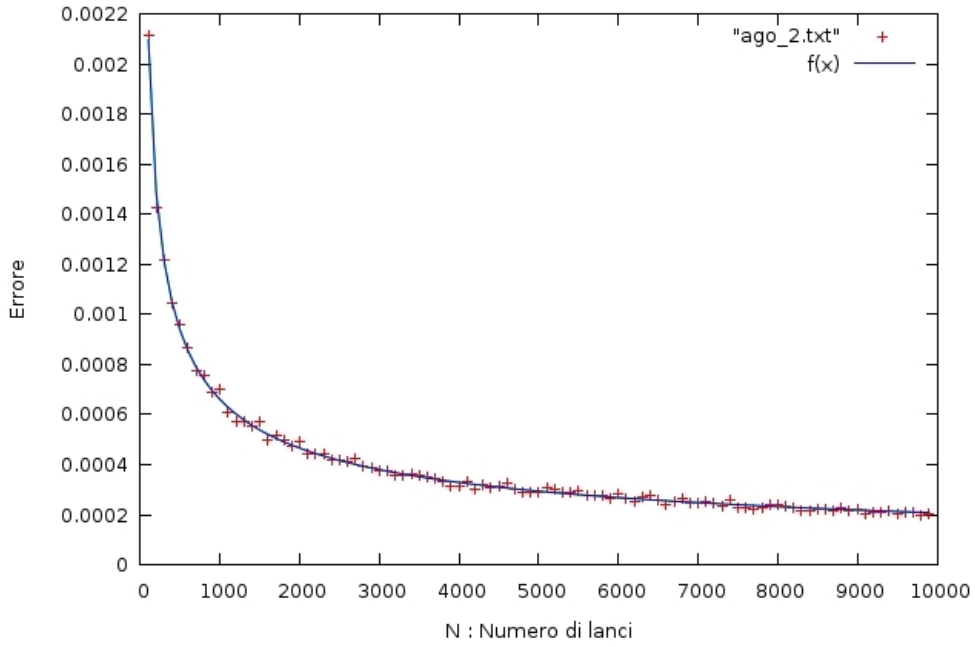


Figura 6: Grafico del andamento della deviazione standard sulla media di $P(n)$ in funzione del numero dei lanci



In conclusione l'andamento di $\delta_{statistico}(N)$ è del tipo:

$$\delta_{statistico}(N) \sim \frac{1}{\sqrt{N}}$$

Errore computazionale Per stimare l'errore computazionale è necessario studiare la convergenza della simulazione. Quindi si stampa il valore assoluto della differenza tra valore atteso $\frac{2R}{T\pi}$ e la media di 1000 simulazioni, ottenendo il grafico :[Figura6].

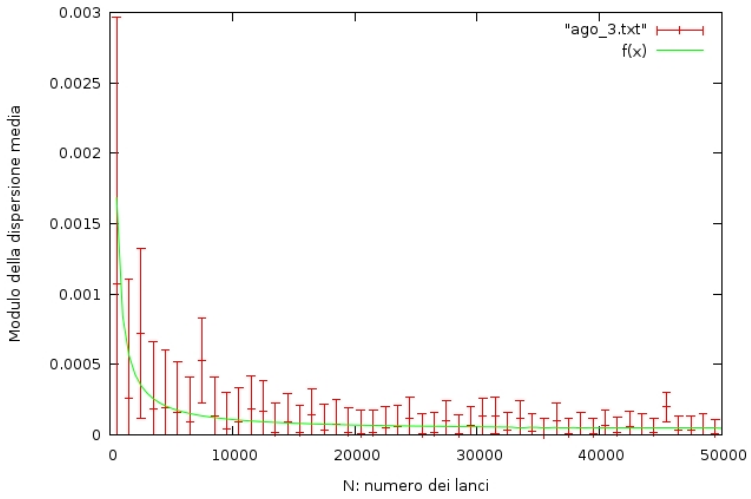


Figura 7: Grafico del andamento del modulo della dispersione della media rispetto al valore atteso in funzione dei lanci considerando gli errori precedentemente stimati.

A questo punto è possibile fittare i dati trovati con una funzione del tipo:

$$f(N) = A + N^B \tag{13}$$

il parametro A restituirà una stima dell'incertezza derivante dal *errore computazionale* mentra il parametro B indicherà la convergenza del modello al valore atteso.

```
After 46 iterations the fit converged.
final sum of squares of residuals : 8.30531
rel. change during last iteration : -7.1159e-07

degrees of freedom    (FIT_NDF)                        : 48
rms of residuals      (FIT_STDFIT) = sqrt(WSSR/ndf)    : 0.415965
variance of residuals (reduced chisquare) = WSSR/ndf   : 0.173027

Final set of parameters            Asymptotic Standard Error
=======================            ==========================

A               = 3.37606e-05      +/- 1.319e-05    (39.08%)
B               = -1.02927         +/- 0.03679      (3.574%)
```

Risulta che il modello converge al valore atteso come $1/N$ mentre

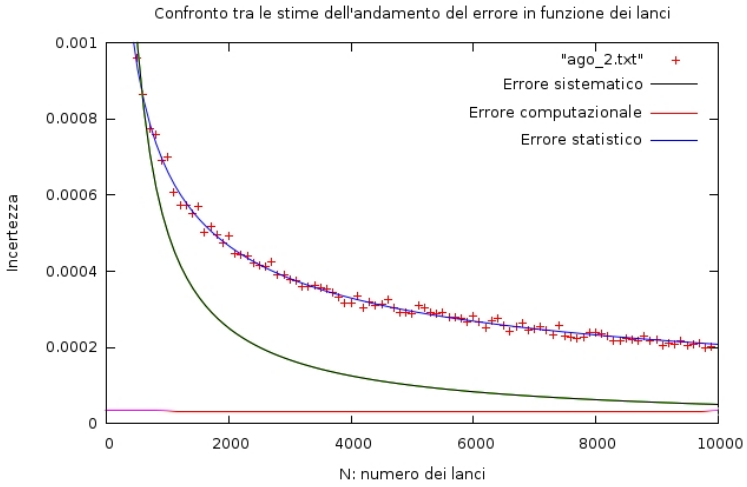
$$\delta_{computazionale} \simeq 4 \times 10^{-5} \tag{14}$$

4 Conclusione

Si ottengono le seguenti leggi per l'errore:

$\delta_{sistematico}$	$\sim \frac{1}{2N}$
$\delta_{statistico}$	$\sim 0,02\frac{1}{\sqrt{N}}$
$\delta_{computazionale}$	$\sim 4 \times 10^{-5}$

Mettendole a confronto si nota che per simulazioni con pochi lanci domina su tutti l'errore sistematico, la situazione si inverte superando i 500 lanci da dove comincia a dominare l'errore statistico. Per la totalità dei casi pratici l'incertezza computazionale risulterà completamente trascurabile rispetto alle altre fonti di errore.



Riferimenti bibliografici

[1] *Numerical recipes in C++*

[2] Knuth D. *The Art of Computer Programming*

[3] Hildebrand F. *Introduction to Numerical Analysis*, 2nd.ed.