

Laboratorio II – 1° modulo

Lezione 4

Metodi Monte Carlo

Indice

- Deviazione standard della media di numeri casuali estratti da una distribuzione di probabilità
- Metodi Monte Carlo:
 - Il metodo Hit or Miss per l'integrazione numerica di funzioni
 - Incertezza statistica associata alla stima di un integrale con il metodo Hit or Miss
- Come *debuggare* il codice
- Esercizi

Deviazione standard della media

Nella scorsa lezione abbiamo visto diverse tecniche per generare numeri casuali che seguano diverse distribuzioni di probabilità

Abbiamo anche calcolato la **media** e la **varianza** di Q numeri casuali estratti da una stessa distribuzione di probabilità

La **stima** di media e varianza può essere vista come una **variabile aleatoria**

Quanto vale la **deviazione standard della media** (m) di Q variabili aleatorie (y_i) indipendenti ed identicamente distribuite?

$$y_i \sim \text{pdf}(x); E[y_i] = \mu; \text{Var}[y_i] = \sigma^2; \forall y_i$$

$$E[m] = \mu; \text{Var}[m] = 1/Q^2 \cdot Q \text{Var}[y_i] = \sigma^2/Q$$

$$m = \frac{1}{Q} \sum_{i=1}^Q y_i$$

Il Teorema Centrale del Limite ci garantisce inoltre che m tende ad una distribuzione Normale di media μ e varianza σ^2/Q

Possiamo calcolare l'incertezza sulla media attraverso il seguente esercizio e confrontare il risultato numerico con quanto previsto dal Teorema Centrale del Limite

Deviazione standard della media

1. Estrarre $Q=10^4$ numeri casuali da una distribuzione uniforme nel range $[a, b]$ e calcolare la media dei Q numeri estratti
2. Ripetere la procedura al punto 1. per $N=10^4$ volte e inserire i valori medi ottenuti in un istogramma `TH1F`:

```
...
TH1F h1 ("histogram", "title", 100, 4.7, 5.3);
double mean, random, a = 0., b = 10.;
for (int i = 0; i < N; i++)
```

E' stata usata una distribuzione uniforme nell'intervallo $[0, 10]$

```
{
    mean = 0.;
    for (int j = 0; j < Q; j++)
```

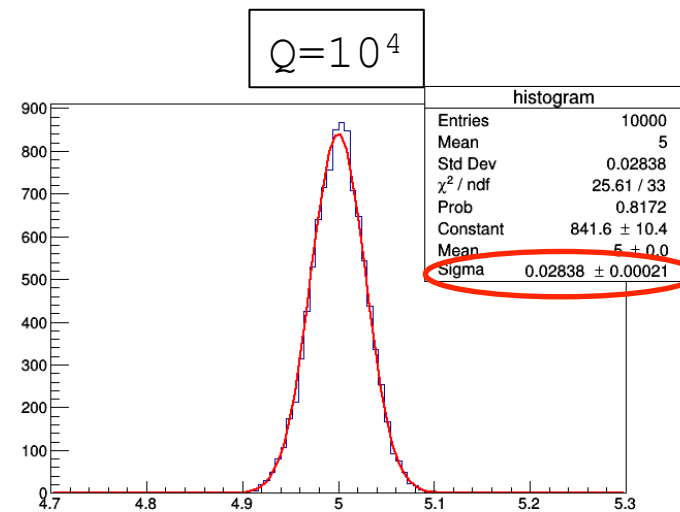
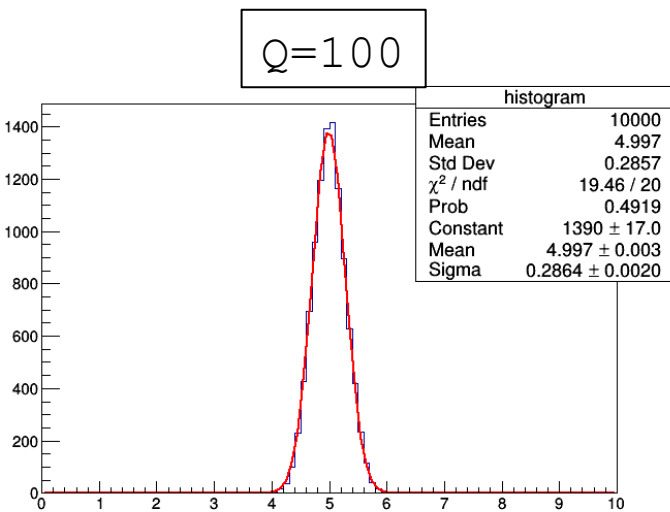
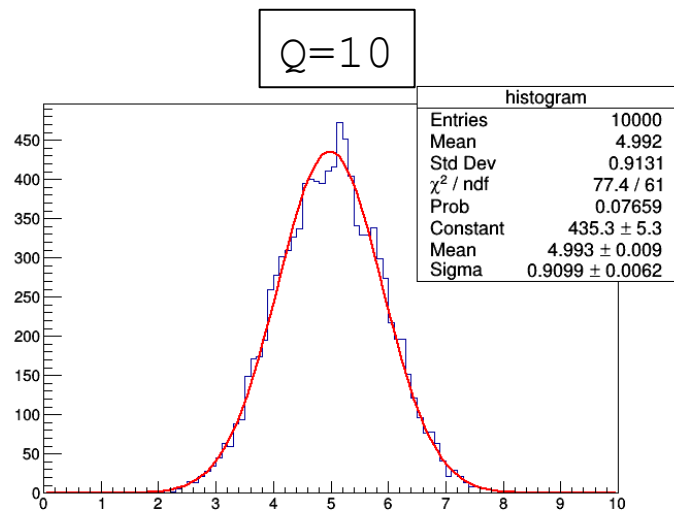
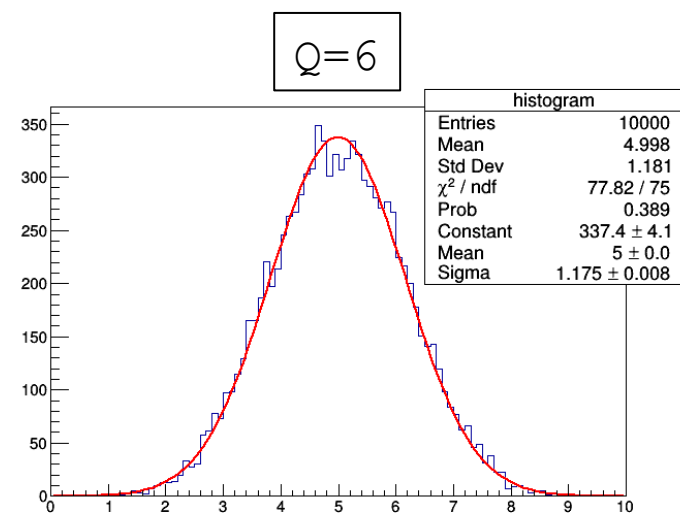
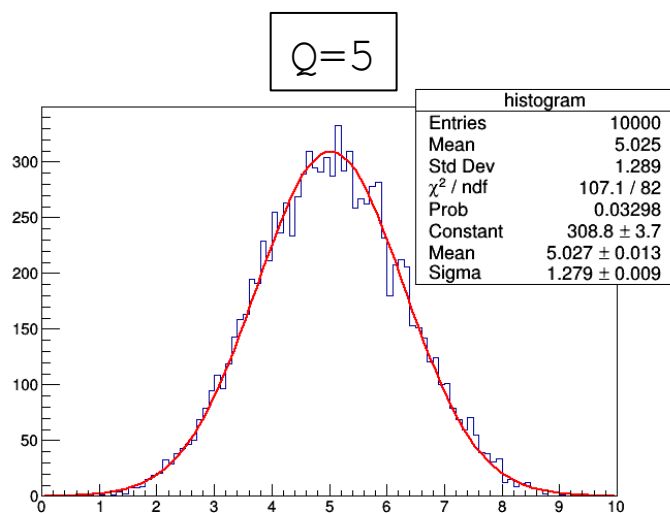
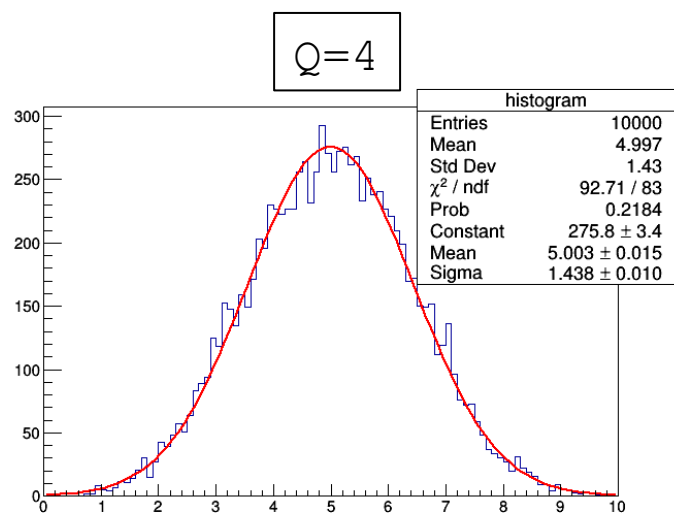
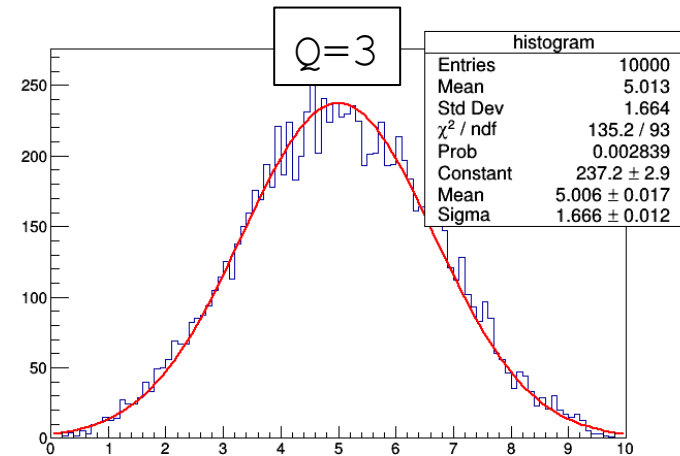
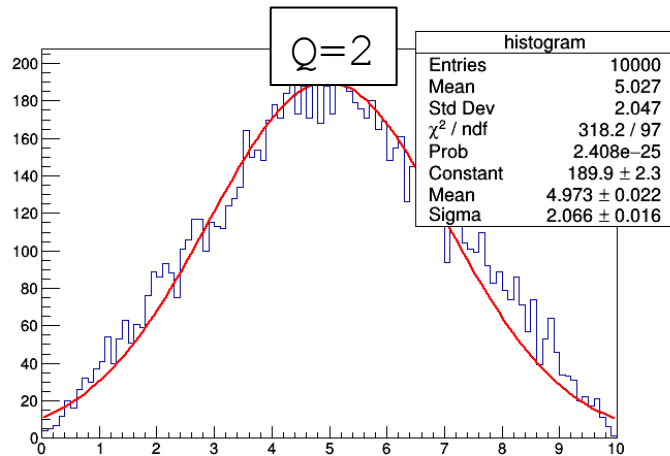
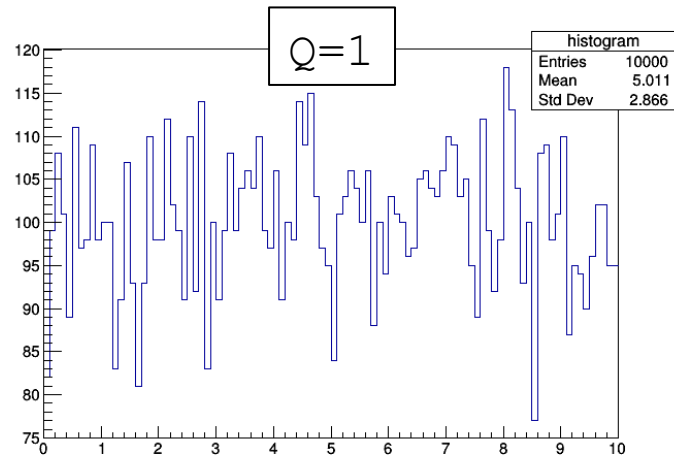
La varianza della distribuzione uniforme è:

$$\sigma^2 = \frac{(b-a)^2}{12} = \frac{100}{12}$$

```
    {
        random = rand_range(a, b);
        mean += random;
    }
    mean /= Q;
    h1.Fill(mean);
}
```

Quindi, la varianza che ci aspettiamo per la pdf dei valori medi è:

$$\sigma_m^2 = \frac{1}{Q} \frac{(b-a)^2}{12} = \frac{1}{1200} \Rightarrow \sigma_m = \sqrt{\frac{1}{1200}} \approx 0.029$$



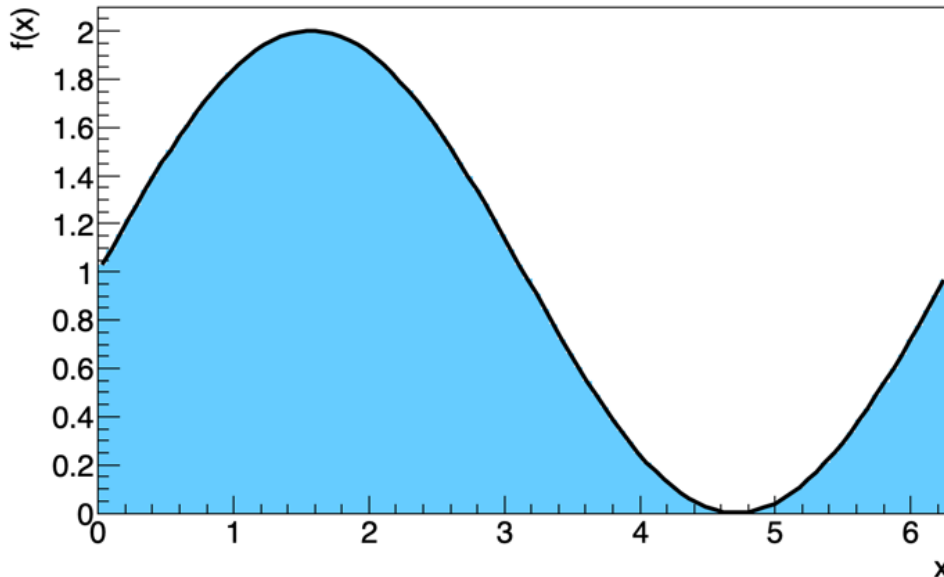
Metodi Monte Carlo

- Le tecniche di calcolo e simulazione basate sulla generazione di numeri casuali prendono il nome di metodi Monte Carlo (MC)
- Il nome deriva dal famoso casinò di Monte Carlo, i cui giochi sono basati, appunto, sull'estrazione di numeri casuali
- I metodi MC sono ampiamente utilizzati in fisica per simulare e risolvere problemi complessi:
 - per simulare la propagazione di fasci di particelle nei rivelatori
 - per l'integrazione sullo spazio fasi delle ampiezze di probabilità della meccanica quantistica per il calcolo, per esempio, delle sezioni d'urto
 - per l'integrazione delle densità di probabilità durante la procedura di regressione con il metodo della Likelihood
 - etc...
- Vediamo ora un semplice esempio di applicazione dei metodi Monte Carlo proprio al calcolo dell'integrale di una funzione

Integrazione con metodi Monte Carlo

- Assumiamo funzioni regolari (continue su un compatto)
- Consideriamo funzioni definite positive
- Utilizziamo come esempio la funzione:

$$f(x) = 1 + \sin(x)$$



Codice C++:

```
double fsin (double x)
{
    return 1. + sin(x);
}
```

Per questo semplice esempio sappiamo che:

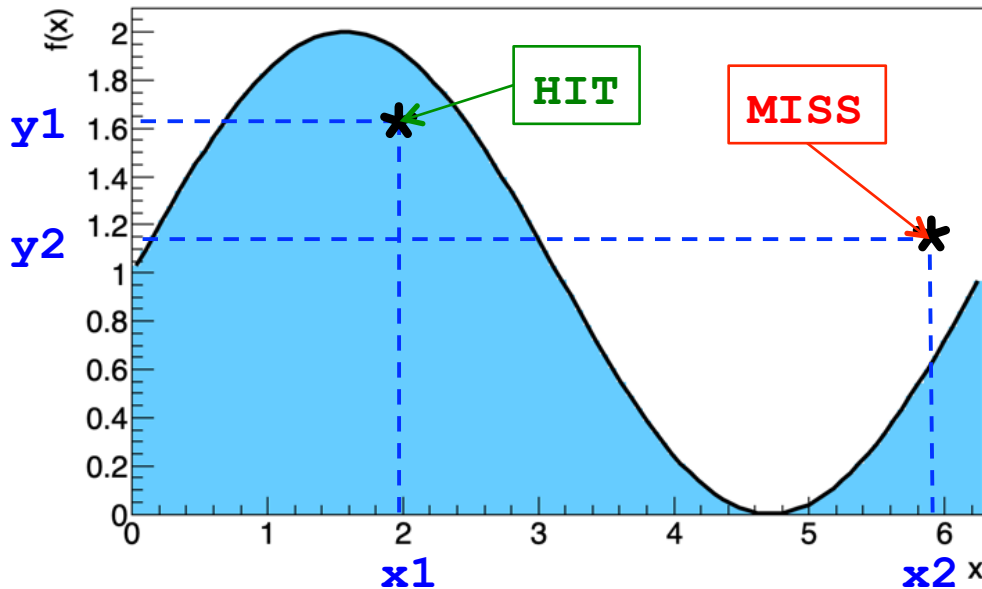
$$\int_0^{2\pi} 1 + \sin(x) dx = 2\pi$$

Integrazione con il metodo Hit or Miss

La logica è la stessa del metodo Try&Catch:

- Si generano coppie di numeri casuali che identificano un punto su una porzione rettangolare del piano che includa completamente la parte di funzione di cui si vuole calcolare l'integrale
- Si usa un contatore `n_hit` per contare quanti punti si trovano al di sotto della funzione

$$\frac{n_hit}{N} \xrightarrow{N \rightarrow \infty} \int_0^{2\pi} 1 + \sin(x) dx / A$$



- A è l'area del rettangolo in cui si sono generati i punti casuali
- Questo metodo di integrazione numerica fornisce una stima approssimata dell'integrale
- Tale stima è tanto più precisa quanto maggiore è il numero di punti campionati

Incertezza statistica di Hit or Miss

- L'integrale definito di $f(x)$ nell'intervallo $[a, b]$, che chiameremo I , stimato con il metodo MC, è a sua volta una variabile aleatoria che segue la stessa distribuzione di probabilità di n_{hit} :

$$\hat{I} = \frac{A}{N} n_{hit} \quad \Rightarrow \quad V[\hat{I}] = \frac{A^2}{N^2} V[n_{hit}] = \frac{A^2}{N} p(1-p)$$

dove A è l'area del rettangolo in cui si sono generati i punti casuali

- Infatti poiché Hit or Miss è un processo **binomiale**, n_{hit} ha i seguenti momenti:

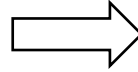
$$E[n_{hit}] = Np \quad V[n_{hit}] = Np(1-p)$$

dove p è la probabilità di ottenere un hit, che può essere stimata dal rapporto n_{hit}/N , quindi l'espressione finale per la varianza di I risulta:

$$\hat{V}[\hat{I}] = \frac{A^2}{N} \frac{n_{hit}}{N} \left(1 - \frac{n_{hit}}{N}\right)$$

Implementazione metodo Hit or Miss

File .cc
implementazione delle
funzioni esterne al main



```
int main()
{
    srand(time(NULL));
    int N      = 100000;
    int nHit   = 0;
    double xMin = 0., xMax = 2*M_PI;
    double yMin = 0., yMax = 2.;

    for (int i = 0; i < N; i++)
    {
        if (HitMiss(xMin, xMax, yMin, yMax) == true)
            nHit++;
    }

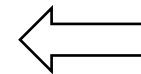
    double Area      = (xMax-xMin) * (yMax-yMin);
    double Integral  = nHit*Area / (double)N;
    double p         = nHit / (double)N;
    double Var       = Area*Area / (double)N * p * (1.-p);
    double StdDev    = sqrt(Var);

    std::cout << "Integral = " << Integral
               << " +/- " << StdDev << std::endl;

    return 0;
}
```

```
double fsin (double x)
{
    return 1. + sin(x);
}

bool HitMiss (double xMin, double xMax,
              double yMin, double yMax)
{
    double x = 0., y = 0.;
    x = rand_range(xMin, xMax);
    y = rand_range(yMin, yMax);
    if (y < fsin(x)) return true;
    else return false;
}
```



File .cpp

Come *debuggare* il codice

- Programmi sempre più complicati richiedono tecniche di *debug* sempre più sofisticate
- Nella prima lezione abbiamo visto come leggere i messaggi che il compilatore ci fornisce in caso di errore durante la fase di compilazione. Il messaggio racchiude in sé la riga in cui il compilatore ha trovato una inconsistenza con la sintassi del linguaggio, in maniera tale che il programmatore sappia immediatamente dove andare a guardare per rimediare
- Errori logici, procedurali, sono più difficili da trovare perché in genere si verificano a *run time*. Per individuare questi errori è opportuno mettere delle stampe nel codice in maniera tale da individuare esattamente quale riga è responsabile dell'arresto del programma. Le stampe possono essere di questo tipo:

```
std::cout << __PRETTY_FUNCTION__  
          << "\tROW: " << __LINE__ << std::endl;
```

__PRETTY_FUNCTION__ fornisce il nome della funzione in cui è stato inserito il `cout`

__LINE__ fornisce il numero di riga del codice sorgente in cui è inserita

Attenzione: anche se individuate la riga responsabile dell'arresto del programma non è detto che l'errore sia in quella riga. In questo modo però avete ulteriori indizi per capire dove potrebbe essere il problema

Esercizi

Svolgere gli esercizi seguenti organizzando le funzioni esterne al `main` in un'unica libreria, composta da un file con l'implementazione delle funzioni (`.cc`) e da un header file (`.h`) contenente i prototipi delle funzioni

Esercizio 1: Completare l'esercizio sull'incertezza della media

1. Estrarre $Q=10^4$ numeri casuali da una distribuzione uniforme nel range $[a, b]$ e calcolare la media dei Q numeri estratti
2. Ripetere la procedura al punto 1. per $N=10^4$ volte e inserire i valori medi ottenuti in un istogramma `TH1F`
3. Calcolare la media e la deviazione standard dei valori medi calcolati al punto 2.
4. Confrontare il risultato ottenuto con quanto previsto dal Teorema Centrale del Limite
5. Ripetere l'esercizio variando il valore di Q (comunicato al programma tramite l'istruzione `std::cin`) e verificare che la deviazione standard della media è proporzionale a $1/\sqrt{Q}$

Esercizi

Esercizio 2: Ripetere l'esercizio 1, estraendo numeri casuali da una distribuzione Normale (usate il metodo Try&Catch) invece che da una distribuzione uniforme

Quale è la relazione tra la deviazione standard della pdf Normale di partenza e la deviazione standard della media?

Esercizio 3:

1. Generare $N=10^4$ numeri casuali da una pdf Normale con $\mu=0$, $\sigma=0.5$. Inserire questi numeri in un istogramma TH1F
2. Generare $N=10^4$ numeri casuali da una pdf Normale con $\mu=0$, $\sigma=2$. Inserire questi numeri nello stesso istogramma definito al punto 1.
3. Disegnare l'istogramma risultante
È una distribuzione Normale? Perché?

Esercizi

Esercizio 4:

1. Generare $N=10^4$ numeri casuali da una pdf Normale con $\mu=0$, $\sigma=0.5$
2. Generare $N=10^4$ numeri casuali da una pdf Normale con $\mu=0$, $\sigma=2$
3. Sommare ogni coppia di numeri estratti ai punti 1. e 2. ed inserire il risultato in un istogramma
4. Disegnare l'istogramma risultante

È una distribuzione Normale? Perché?

Quanto vale la sua deviazione standard?

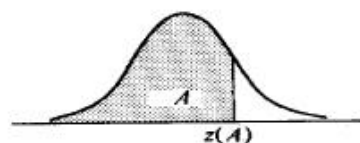
Esercizi

Esercizio 5: Utilizzare il metodo Hit or Miss per stimare l'integrale sotteso ad una pdf Normale con $\mu=0$, $\sigma=1$ in un generico intervallo $[a, b]$:

$$\int_a^b dx \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$$

Calcolare l'integrale su diversi intervalli $[a, b]$ con $a=-5$ e b **variabile**, e confrontare i risultati ottenuti con quelli riportati nella tabella accanto

Entry is area A under the standard normal curve from $-\infty$ to $z(A)$



z	.00	.01	.02	.03	.04	.05	.06	.07	.08	.09
.0	.5000	.5040	.5080	.5120	.5160	.5199	.5239	.5279	.5319	.5359
.1	.5398	.5438	.5478	.5517	.5557	.5596	.5636	.5675	.5714	.5753
.2	.5793	.5832	.5871	.5910	.5948	.5987	.6026	.6064	.6103	.6141
.3	.6179	.6217	.6255	.6293	.6331	.6368	.6406	.6443	.6480	.6517
.4	.6554	.6591	.6628	.6664	.6700	.6736	.6772	.6808	.6844	.6879
.5	.6915	.6950	.6985	.7019	.7054	.7088	.7123	.7157	.7190	.7224
.6	.7257	.7291	.7324	.7357	.7389	.7422	.7454	.7486	.7517	.7549
.7	.7580	.7611	.7642	.7673	.7704	.7734	.7764	.7794	.7823	.7852
.8	.7881	.7910	.7939	.7967	.7995	.8023	.8051	.8078	.8106	.8133
.9	.8159	.8186	.8212	.8238	.8264	.8289	.8315	.8340	.8365	.8389
1.0	.8413	.8438	.8461	.8485	.8508	.8531	.8554	.8577	.8599	.8621
1.1	.8643	.8665	.8686	.8708	.8729	.8749	.8770	.8790	.8810	.8830
1.2	.8849	.8869	.8888	.8907	.8925	.8944	.8962	.8980	.8997	.9015
1.3	.9032	.9049	.9066	.9082	.9099	.9115	.9131	.9147	.9162	.9177
1.4	.9192	.9207	.9222	.9236	.9251	.9265	.9279	.9292	.9306	.9319
1.5	.9332	.9345	.9357	.9370	.9382	.9394	.9406	.9418	.9429	.9441
1.6	.9452	.9463	.9474	.9484	.9495	.9505	.9515	.9525	.9535	.9545
1.7	.9554	.9564	.9573	.9582	.9591	.9599	.9608	.9616	.9625	.9633
1.8	.9641	.9649	.9656	.9664	.9671	.9678	.9686	.9693	.9699	.9706
1.9	.9713	.9719	.9726	.9732	.9738	.9744	.9750	.9756	.9761	.9767
2.0	.9772	.9778	.9783	.9788	.9793	.9798	.9803	.9808	.9812	.9817
2.1	.9821	.9826	.9830	.9834	.9838	.9842	.9846	.9850	.9854	.9857
2.2	.9861	.9864	.9868	.9871	.9875	.9878	.9881	.9884	.9887	.9890
2.3	.9893	.9896	.9898	.9901	.9904	.9906	.9909	.9911	.9913	.9916
2.4	.9918	.9920	.9922	.9925	.9927	.9929	.9931	.9932	.9934	.9936
2.5	.9938	.9940	.9941	.9943	.9945	.9946	.9948	.9949	.9951	.9952
2.6	.9953	.9955	.9956	.9957	.9959	.9960	.9961	.9962	.9963	.9964
2.7	.9965	.9966	.9967	.9968	.9969	.9970	.9971	.9972	.9973	.9974
2.8	.9974	.9975	.9976	.9977	.9977	.9978	.9979	.9979	.9980	.9981
2.9	.9981	.9982	.9982	.9983	.9984	.9984	.9985	.9985	.9986	.9986
3.0	.9987	.9987	.9987	.9988	.9988	.9989	.9989	.9989	.9990	.9990
3.1	.9990	.9991	.9991	.9991	.9992	.9992	.9992	.9992	.9993	.9993
3.2	.9993	.9993	.9994	.9994	.9994	.9994	.9994	.9995	.9995	.9995
3.3	.9995	.9995	.9995	.9996	.9996	.9996	.9996	.9996	.9996	.9997
3.4	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9997	.9998

Esercizi

Esercizio 6: Implementare il seguente metodo Monte Carlo per l'integrazione di una funzione $f(x)$ (e.g. una pdf Normale) definita nell'intervallo $[a, b]$:

1. Estrarre una variabile aleatoria, x , secondo una distribuzione di probabilità uniforme nell'intervallo $[a, b]$
2. Valutare $y = f(x)$

Ripetere il procedimento $N=10^4$ volte e calcolare, al termine del ciclo di estrazione, la media e la varianza delle y ottenute

Valutare l'integrale tramite:

$$I = (b - a) \left(E[y] \pm \sqrt{\text{Var}[y] / N} \right)$$

N.B.: questo metodo è anche chiamato “Crude Monte Carlo”

Il metodo Crude Monte Carlo

Per come è definita, la y è una variabile aleatoria di cui possiamo quindi calcolare il valore di aspettazione:

$$E[y] = \int_{a'}^{b'} yh(y) dy = \int_a^b yU(x) dx = \frac{1}{b-a} \int_a^b y dx = \frac{1}{b-a} \int_a^b f(x) dx = \frac{I}{b-a}$$

Inoltre la variabile aleatoria “media delle y ” (M) ha il seguente valore di aspettazione e varianza:

Per la proprietà di invarianza
della probabilità elementare

$$E[M] = \frac{I}{b-a} \quad Var[M] = \frac{Var[y]}{N}$$

Quindi I può essere scritto come $I = (b-a) E[M] = (b-a) E[y]$ e la sua deviazione standard come $(b-a) \sqrt{Var[y]/N}$, da cui:

$$I = (b-a) \left(E[y] \pm \sqrt{Var[y]/N} \right)$$

Nella pratica stimiamo $E[y]$ e $Var[y]$ tramite la media e la varianza campionaria delle y estratte