03/12/2013

Informatica per la Finanza
Informatica per le Scienze Economiche
Informatica e Programmazione per il Turismo

Metodi MonteCarlo

- Sono procedure di calcolo basate sull'uso di generatori di numeri casuali per ottenere il risultato.
- Descritti da Stanislaw Ulam sono nati per risolvere problemi complessi. Hanno diverse applicazioni fra cui:
  - integrazione numerica,
  - soluzione numerica di (sistemi di) equazioni differenziali,
  - simulazione di processi stocastici,
  - modellazione di sistemi fisico-statistici (sono intrinsecamente dei processi stocastici),
  - ...

Passeggiate a caso

- Un problema tipico è quello della simulazione di passeggiate a caso (nel discreto).
  - Un punto si muove a partire da una posizione:
    - di un numero di passi (al limite casuale)
    - in una direzione scelta a caso
    - indipendentemente dai passi precedenti
    - il movimento continua fino ad una barriera dove il punto può essere:
      - assorbito
      - rimbalzato
  - Esempi tipici sono:
    - Problema della rovina del giocatore
    - Moto browniano

Rovina del giocatore

- Un giocatore possiede **M** monete e partecipa ad un gioco di lancio di una moneta. Ad ogni lancio se esce testa vince **V** monete, se esce croce perde **P** monete.
- Il giocatore si ritirerà quando
  - Ha perso tutte le monete
  - Ha vinto **C** monete
  - Nel caso più semplice P=V

La rovina del giocatore

- Per **V=P** si può dimostrare che la probabilità di vittoria (tornare a casa con M+C monete) vale

M/(M+C)

- Ovvero più si vuole vincere più è probabile che si resti senza una moneta

La rovina del giocatore

- Implementiamo questo problema in MatLab
  - Dobbiamo modellare una partita di questo gioco partendo dai valori di M, C, P e V
    - Una partita termina quando il giocatore ha raggiunto una cifra complessiva M+C o ha perso tutto.
  - Giocando N partite contando quante se ne vincono (indichiamo con **`vinte`** questo valore)
    - Lo scarto fra il valore teorico (M/(M+C) se P=V) e quello sperimentale decresce come

$$1/\sqrt{N}$$

La rovina del giocatore

```
function [prob, vinte, totale] = rovinaGiocatore(M, C, N, P, V)
% gestisce la rovina del giocatore
if (nargin == 4)
    V = P;
else
    if (nargin < 4)
        error('pochi argomenti')
    end
end

vinte = 0;

for i=1:N
    if partita (M, C, P, V) == 1
        vinte = vinte + 1;
    end
end

totale = N;
prob = vinte/totale;
end
```

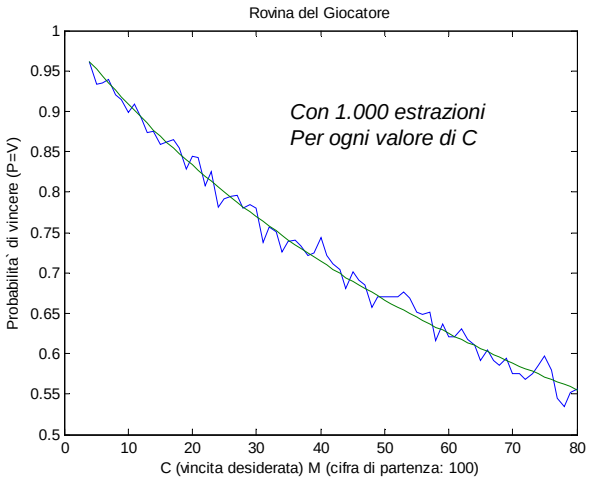

La rovina del giocatore

```
function r = partita(M, C, P, V)
   somma = M;
   while somma < M+C & somma > 0
       if rand > 0.5
           somma = somma + V;
       else
           somma = somma - P;
       end
   end

   r = somma >= M+C;
end
```

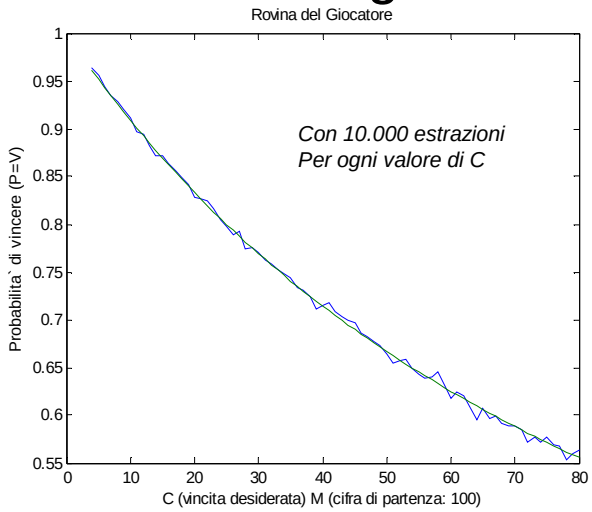
La rovina del giocatore

```
% grafico teorico vs sperimentale
i = int16(1);
C=4:1:80;
for vincita=C
  r(i) = rovinaGiocatore(100,vincita,1000,1);
  i=i+1;
end

teo = 100./(100+C);

plot(C,r,C,teo)
title('Rovina del Giocatore');
xlabel('C (vincita desiderata) M (cifra di partenza: 100)');
ylabel('Probabilita` di vincere (P=V)');
```


La rovina del giocatore



La rovina del giocatore



La rovina del giocatore

- Implementare con VBA il problema della rovina del giocatore

Modello di Ehrenfest

- Modella un sistema di diffusione di due sostanze gassose contenute in due urne separate da una membrana che viene rimossa
  - Lo modelliamo con una ulteriore astrazione:
    - Supponiamo di avere due urne A e B ciascuna con n palline;
      - al tempo **t=0**
        - » tutte le palline in A sono bianche
        - » tutte le palline in B sono rosse.
      - Al tempo t=1
        - » si preleva una pallina a caso da A e la si mette B
        - » contemporaneamente si prende una pallina da B e la si mette in A
      - Si ripete il passo precedente X volte
    - Si calcola ad ogni passo quante palline bianche sono nell'urna A

Ehrenfest

Non varia la composizione delle urne

eA < frPbA

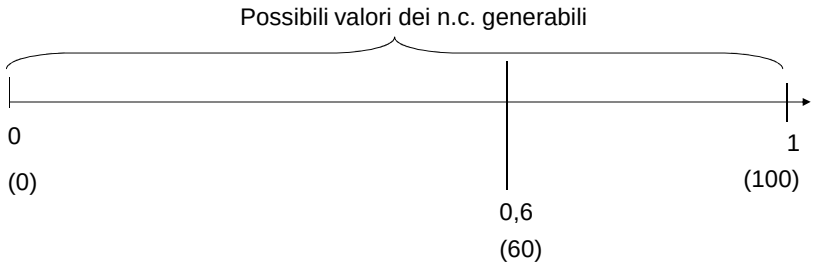
eB < frPrB

Estraggo	Bianca inA	Rossa inA	Bianca inB	Rossa in B
	V		V	
		V		V
	V			V
		V	V	

Varia la composizione delle urne

Ehrenfest

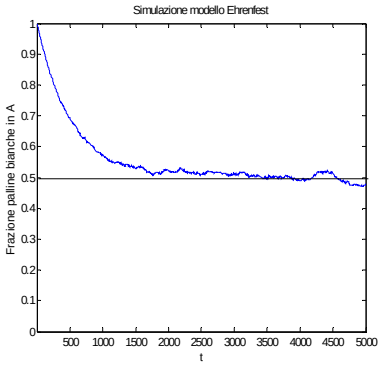
- Come modellare la probabilità di estrazione:
  - Esempio: probabilità Bianca 60% (0,6)

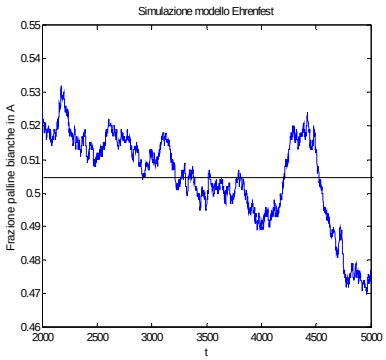


Modello di Ehrenfest

```
function [pbA, st] = Ehrenfest(n,X)
   if (nargin==0)
        n=1000;
        X=5000;
   elseif (nargin == 1)
        X=5000;
   end

   pbA=n;  % bianche in A
   frpbA=pbA/n;  %Freq in A
   prB=n;  % rosse in B
   frprB=prB/n;  %Freq in B

   for i=1:X
       st(i)=frpbA;
       eA = rand;
       eB = rand;
       if (eA<frpbA & eB<frprB )
                %A bianca, B rossa
            pbA=pbA-1;
            prB=prB-1;
       elseif(eA>=frpbA & eB>=frprB)
                %A rossa, B bianca
            pbA=pbA+1;
            prB=prB+1;
       end
        frpbA=pbA/n;
        frprB=prB/n;
   end

   plot(st);
   axis([1 X 0 1]);
   xlabel('t');
  ylabel('% palline bianche in A');
title('Simulazione modello Ehrenfest');

end
```

Modello di Ehrenfest





Ricavato con:

```
[f s]=Ehrenfest
plot([2000:length(s)],s(2000:length(s)))
```


Modello di Ehrenfest

- Questo esempio evidenzia due caratteristiche tipiche dei metodi MonteCarlo
  - Necessità di un certo numero di esecuzioni per raggiungere lo stato stazionario
  - Oscillazioni intorno allo stato stazionario
- I rimedi
  - Effettuare più run di simulazione e quindi eseguire una media sui risultati
  - Non considerare i primi valori nel calcolo della media

Calcolo di aree

– Altra applicazione dei metodi MonteCarlo è nel calcolo delle aree di figure.
  - poiché la convergenza verso il risultato è piuttosto lenta questa metodologia si applica per:
    – integrazioni su più dimensioni
    – Integrazioni di funzioni complicate per i metodi tradizionali (Newton)
  - Esemplificato per l'integrazione a 2 dimensioni

Integrazione MonteCarlo

- Integrazione MonteCarlo con il metodo accetta-rifiuta
  – Richiede che:
    - La funzione sia definita nell'intervallo [0,1]

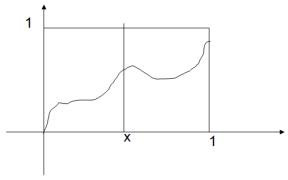
$$\int_a^b f(x)\cdot dx = (b-a)\int_0^1 f(t)\cdot dt \quad x = (b-a)\cdot t + a$$

    - Abbia lo stesso segno nell'intervallo di integrazione
      – Ci si può riportare in questa condizione dividendo l'intervallo in partizioni nelle quali la funzione ha lo stesso segno e quindi si sommano algebricamente i risultati
    - Abbia l'ordinata compresa fra [0,1]
      – Ci si porta in questa condizione dividendo per f_{max} il valore di f(x).

Integrazione MonteCarlo

– Fatte queste premesse la funzione viene racchiusa in un quadrato 1 x 1
  - si generano N punti interni a questo quadrato.
  - Si conta quanti punti cadono sotto la funzione (N_{amm})
  - il rapporto N_{amm} /N è una stima dell'area della funzione tanto migliore quanto più è alto il valore di N
  - Se l'ascissa della funzione era in [a,b] e l'ordinata in [0,f_{max}] l'area è data da:

$$(N_{amm} / N)\ f_{max}\ (b-a)$$



Integrazione MonteCarlo

- Calcolare l'area della funzione

$$x^3 - 3x^2 - x + 3$$

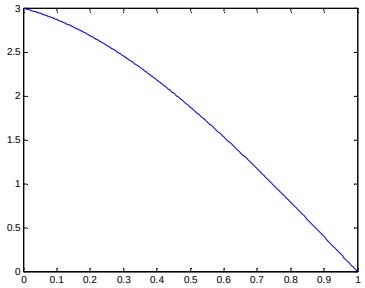
fra 0 ed 1 utilizzando il metodo Montecarlo

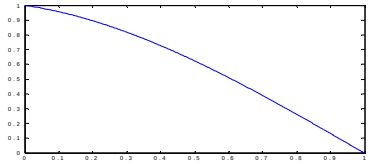
Il valore dell'integrale è 1,75

Il massimo è in 0 e vale 3

Il minimo è in 1 e vale 0

Integrazione MonteCarlo





```
x=linspace(0,1,10^6);
y=polyval(poli,x);
plot(x,y);
```

```
x=linspace(0,1,10^6);
y=polyval(poli,x)./3;
plot(x,y);
```

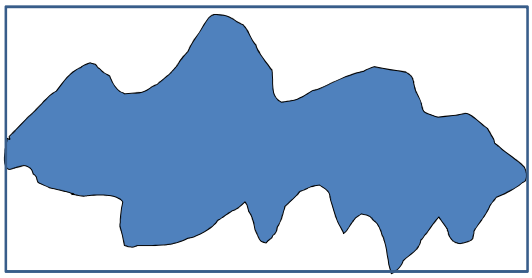

Integrazione MonteCarlo

```
punti=int32(10^5);
area=0;
poli=[1 -3 -1 3];
tic
for i=1:punti
    x=rand;    %0<=x<=1
    y=rand;    %0<=y<=1
    q=polyval(poli,x)./3;
    if q>y
        area=area+1;
    end
end
tT=toc
area=area/double(punti)*3
```

```
punti=int32(10^5);
area=0;
poli=[1 -3 -1 3];
tic
x=rand(1,punti);
y=rand(1,punti);
q=polyval(poli,x)./3;
cf=q>y;
area=sum(cf);
tT=toc
area=area/double(punti)*3
```

Integrazione MonteCarlo

- Avendo a disposizione un metodo per verificare se un punto (x,y) appartiene alla figura è possibile calcolare l'area con il metodo Montecarlo



Integrazione Montecarlo

- Un altro metodo per ottenere il valore dell'integrale di una funzione è usare il teorema del valor medio:
  - Data una funzione f continua su di un intervallo [a,b] esiste un valore c tale che:

$$J = \int_a^b f(x)\,dx = f(c)\cdot(b-a)$$

Dove f(c) è il valor medio della funzione in [a,b]

Integrazione Montecarlo

- L'approccio Montecarlo prevede di campionare la funzione in N punti indipendenti X1, ...,XN secondo una distribuzione di probabilità p(x), ad esempio:

$$p(x) = \begin{cases} 1/b-a & a \le x \le b \\ 0 & altrove \end{cases}$$

- Questo permette di ricavare una stima per J:

$$\hat{J} = F_N = \frac{1}{N}\sum_{i=1}^{N} f(x_i)\cdot(b-a)$$

Integrazione Montecarlo

- F_N è una variabile casuale di media $E[F_N]=J$
- Supponendo che i campioni siano indipendenti si dimostra che:

$$\sigma[F_N] = \frac{1}{\sqrt{N}}\sigma(Y)$$

- Dove $Y_i = f(X_i)/p(X_i)$
- Applicando il teorema di Chebicev e, per N grande, il teorema centrale limite si ha:

$$\Pr\{|F_N - J| \ge t\cdot\sigma[F_N]\} = \sqrt{2/\pi}\int_t^{\infty} e^{-x^2/2}\cdot dx$$

(t=2 Pr{...}=0,04554; t=3 Pr{...}=0,0026; t=4 Pr{...}=0,0001)

Integrazione Montecarlo

- Utilizzando la funzione **`std()`**
  - Richiede come parametri:
    - » La matrice di punti sui ricavare la deviazione standard
    - » quale deviazione usare (corretta: 0; non corretta 1);facoltativo
    - » Su quale dimensione eseguire il calcolo (vale per le matrici); facoltativo
  - Restituisce
    - » Deviazione standard di ogni colonna della matrice
- si può scrivere una funzione che ha in ingresso:
  - Il puntatore alla funzione di cui si vuole calcolare l'integrale (fz)
  - Il limite inferiore dell'intervallo di integrazione (a)
  - Il limite superiore dell'intervallo di integrazione (b)
  - Il numero di punti in cui valutare la funzione (N)
  - Il valore del parametro t

Integrazione Montecarlo

```
function [vInt scarto diff] = …
                    integraVM(fz,a,b,N,t)
% calcola con il metodo del VM l'integrale di
% una funzione

  Xi = rand(1,N) .* (b-a)  + a;
  Yi = fz(Xi);
  vInt = sum(Yi)/N*(b-a);
  scarto=std(Yi,1)/sqrt(N);

  diff=t*scarto;
end
```

Metodi Montecarlo

- Generalizziamo il risultato in modo che l'utente fornisca invece del valore N un valore per:

$$t \cdot \sigma[F_N] = \frac{t}{\sqrt{N}} \sigma(Y)$$

- La funzione dovrà restituire oltre all'integrale il valore di N per cui è stato ottenuto lo scarto voluto o se ci sono più di 10 milioni di punti terminare comunque

Metodi Montecarlo

```
function [integr, scarto, prob, N] = ...
                 valorMedioStima(fz, a, b, stm, t)
   %N=1000;

   while true
      rd = rand(1,N)*(b-a)+a;
      fYi = fz(rd);
      integr = sum(fYi)/N*(b-a);
      scarto = std(fYi,1)/sqrt(N);
      prob = t * scarto;
      if (prob > stm & N<10^7)
          N=N*10;
      else
        break
      end
   end
```

Esempio di richiamo

```
>> integraVM(@fz,0,1,10^5,3)
```

Funzione richiamata

```
function [y] = fz( x )
   y = polyval([1 -3 -1 3],x);
end
```