

Passeggiate aleatorie

- Note

- Autore

Claudio Marsan
Liceo Cantonale di Mendrisio
Via Agostino Maspoli
CH-6850 Mendrisio (Switzerland)
e-mail: claudio.marsan@liceomendrisio.ch

- Versione

Versione 2.0, 15 marzo 2003
Maple V Release 6.02 for Windows 2000

```
> restart:  
with(plots):  
with(stats):  
with(stats[statplots]):  
Warning, the name changecoords has been redefined
```

- Definizione delle funzioni necessarie

Inizializzazione del randomizzatore.

```
[ > readlib(randomize)();  
1047749830
```

Funzione che restituisce casualmente 1 o -1.

```
[ > caso := (2 * rand(0..1)) - 1;  
caso := 2 (proc()  
local t;  
global _seed;  
_seed := irem(427419669081*_seed, 999999999989); t := _seed; irem(t, 2)  
end proc) - 1
```

Funzione che determina i singoli passi da compiere.

```
[ > passi := proc(N::posint)  
local i, posizione;  
posizione := [seq(caso(), i=1..N)];  
end;  
passi :=  
proc(N::posint) local i, posizione; posizione := [seq(caso( ), i = 1 .. N)] end proc
```

Funzione che calcola la quota dopo ogni passo.

```
[
```

```

> quota := proc(L::list)
  local N, y, i, k;
  N := nops(L);
  y[1] := L[1];
  for i from 2 to N do
    y[i] := y[i-1] + L[i];
  end do;
  y := [seq(y[i], i=1..N)];
end;

```

```

quota := proc(L::list)

```

```

local N, y, i, k;
N := nops(L);
y[1] := L[1];
for i from 2 to N do y[i] := y[i-1] + L[i] end do;
y := [seq(y[i], i = 1 .. N)]
end proc

```

Funzione che calcola la quota massima di una passeggiata aleatoria.

```

> quota_max := proc(L::list)
  local N, i;
  N := nops(L);
  max(seq(L[i], i=1..N));
end;

```

```

quota_max := proc(L::list) local N, i; N := nops(L); max(seq(L[i], i = 1 .. N)) end proc

```

Funzione che calcola la quota minima di una passeggiata aleatoria.

```

> quota_min := proc(L::list)
  local N, i;
  N := nops(L);
  min(seq(L[i], i=1..N));
end;

```

```

quota_min := proc(L::list) local N, i; N := nops(L); min(seq(L[i], i = 1 .. N)) end proc

```

Funzione che calcola il numero di ritorni all'origine.

```

> ritorni := proc(L::list)
  local N, cont, i;
  N := nops(L);
  cont := 0;
  for i from 1 to N do
    if L[i] = 0 then
      cont := cont + 1;
    end if;
  end do;
  cont;

```

```

    end;
    ritorni := proc(L::list)
local N, cont, i;
    N := nops(L);
    cont := 0;
    for i to N do if L[i] = 0 then cont := cont + 1 end if end do;
    cont
end proc

```

Funzione che conta i passaggi attraverso l'origine.

```

> passaggi := proc(L::list)
local N, i, cont;
    N := nops(L);
    cont := 0;
    for i from 2 to N-1 do
        if L[i] = 0 then
            if L[i-1]*L[i+1] = -1 then
                cont := cont + 1;
            end if;
        end if;
    end do;
    cont;
end;

passaggi := proc(L::list)
local N, i, cont;
    N := nops(L);
    cont := 0;
    for i from 2 to N-1 do
        if L[i] = 0 then if L[i-1]*L[i+1] = -1 then cont := cont + 1 end if end if
    end do;
    cont
end proc

```

Funzione che restituisce la quota finale della passeggiata.

```

> quota_finale := proc(L::list)
    L[nops(L)];
end;

    quota_finale := proc(L::list) L[nops(L)] end proc

```

Funzione che disegna una passeggiata di lunghezza N.

```

> passeggiata := proc(L::list)
    listplot(L, color=red);
end;

```

```
passeggiata := proc(L::list) listplot(L, color = red) end proc
```

Statistica sulla posizione dopo N passi (l'esperimento viene ripetuto M volte).

```
> statistica := proc(M::posint, N::posint)  
  local i, P, q, Q, intervalli, L;  
  Q := [];  
  for i from 1 to M do  
    P := passi(N);  
    q := quota(P);  
    Q := [op(Q), quota_finale(q)];  
  end do;  
  Q := sort(Q);  
  intervalli := [seq((i-0.5)..(i+0.5), i=Q[1]..Q[M])];  
  L := transform[tallyinto](Q, intervalli);  
  histogram(L);  
end;  
statistica := proc(M::posint, N::posint)  
local i, P, q, Q, intervalli, L;  
  Q := [ ];  
  for i to M do P := passi(N); q := quota(P); Q := [op(Q), quota_finale(q)] end do;  
  Q := sort(Q);  
  intervalli := [seq(i - .5 .. i + .5, i = Q[1] .. Q[M])];  
  L := transform[tallyinto](Q, intervalli);  
  histogram(L)  
end proc
```

Alcuni esempi

Passeggiata aleatoria di lunghezza 10

```
La passeggiata.  
> P := passi(10);  
P := [1, -1, -1, 1, 1, 1, 1, 1, -1, 1]  
Le quote raggiunte.  
> Q := quota(P);  
Q := [1, 0, -1, 0, 1, 2, 3, 4, 3, 4]  
La quota massima raggiunta.  
> quota_max(Q);  
4  
La quota minima raggiunta.  
> quota_min(Q);  
-1  
La quota finale raggiunta.  
> quota_finale(Q);
```

4

Numero di ritorni all'origine.

> `ritorni(Q)` ;

2

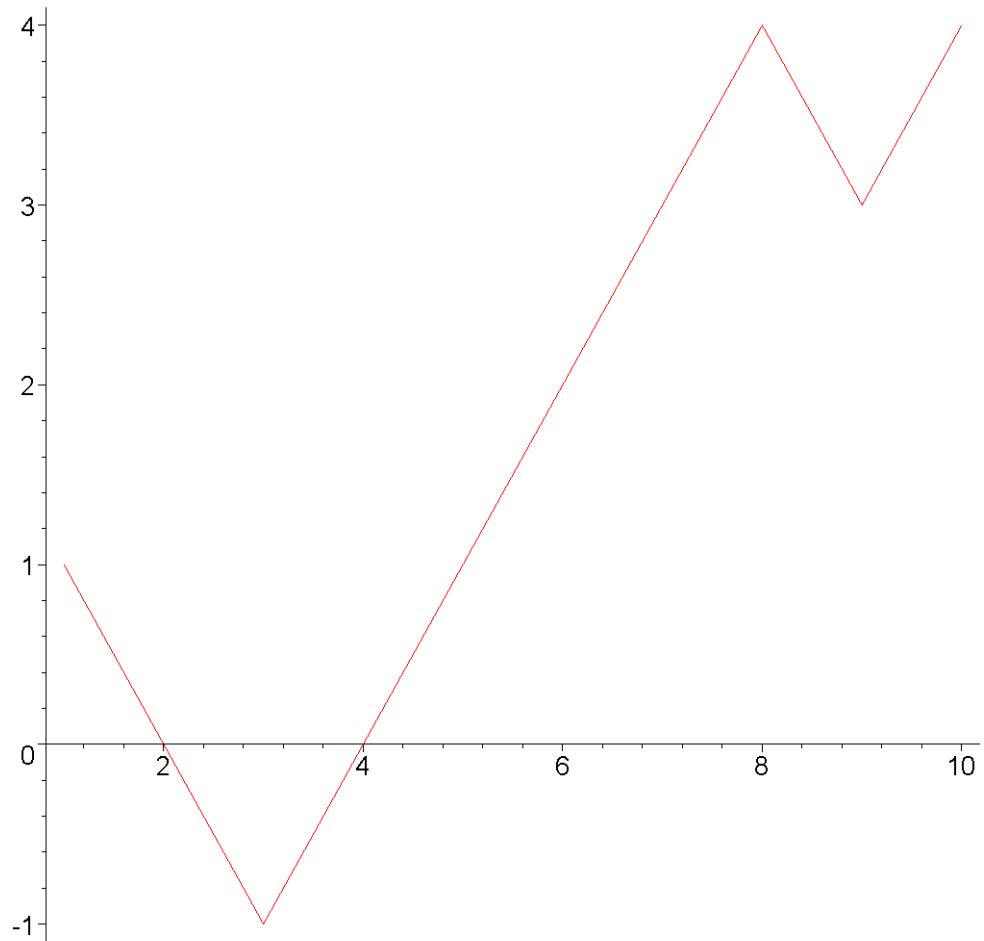
Numero di passaggi attraverso l'origine.

> `passaggi(Q)` ;

2

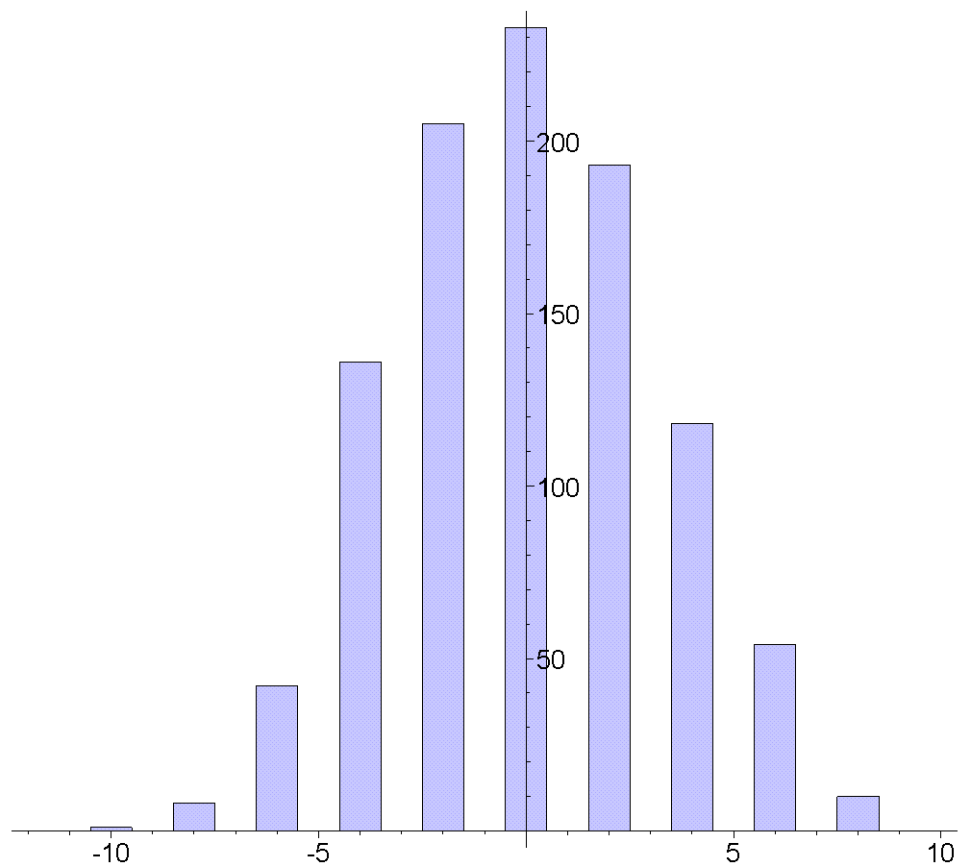
Il grafico della passeggiata aleatoria.

> `passeggiata(Q)` ;



Statistica sul valore finale dopo 10 passi (1000 esperimenti)

> `statistica(1000, 10)` ;



— Passeggiata aleatoria di lunghezza 100

La passeggiata.

```
> P := passi(100);
```

```
P := [-1, 1, -1, -1, 1, 1, 1, -1, 1, -1, -1, -1, -1, 1, -1, 1, -1, -1, 1, 1, -1, 1, 1, -1, -1, 1,
      -1, 1, -1, 1, 1, 1, 1, 1, -1, 1, -1, -1, -1, 1, 1, 1, 1, -1, -1, -1, -1, -1, 1, -1, -1, -1, -1, -1,
      1, 1, 1, -1, 1, 1, -1, -1, 1, -1, -1, 1, 1, 1, -1, -1, -1, -1, 1, -1, -1, 1, -1, 1, 1, -1, -1, 1,
      -1, -1, 1, 1, 1, 1, 1, -1, -1, 1, 1, -1, 1, 1]
```

Le quote raggiunte.

```
> Q := quota(P);
```

```
Q := [-1, 0, -1, -2, -1, 0, 1, 0, 1, 0, -1, -2, -3, -4, -3, -4, -3, -4, -5, -4, -3, -4, -3, -2, -3, -4,
      -3, -4, -3, -4, -3, -2, -1, 0, 1, 2, 1, 2, 1, 0, -1, 0, 1, 2, 3, 2, 1, 0, -1, -2, -1, -2, -3, -4, -5,
      -6, -5, -4, -3, -4, -3, -2, -3, -4, -3, -4, -5, -4, -3, -2, -3, -4, -5, -6, -5, -6, -7, -6, -7, -6, -7,
      -6, -5, -6, -7, -6, -7, -8, -7, -6, -5, -4, -3, -4, -5, -4, -3, -4, -3, -2]
```

La quota massima raggiunta.

```
> quota_max(Q);
```

3

La quota minima raggiunta.

```
> quota_min(Q);
```

-8

La quota finale raggiunta.

```
> quota_finale(Q);
```

-2

Numero di ritorni all'origine.

```
> ritorni(Q);
```

8

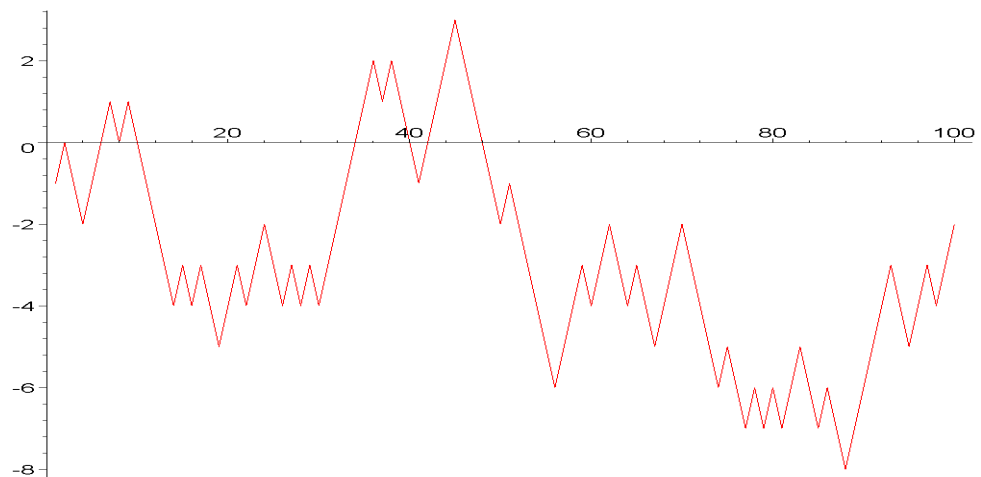
Numero di passaggi attraverso l'origine.

```
> passaggi(Q);
```

6

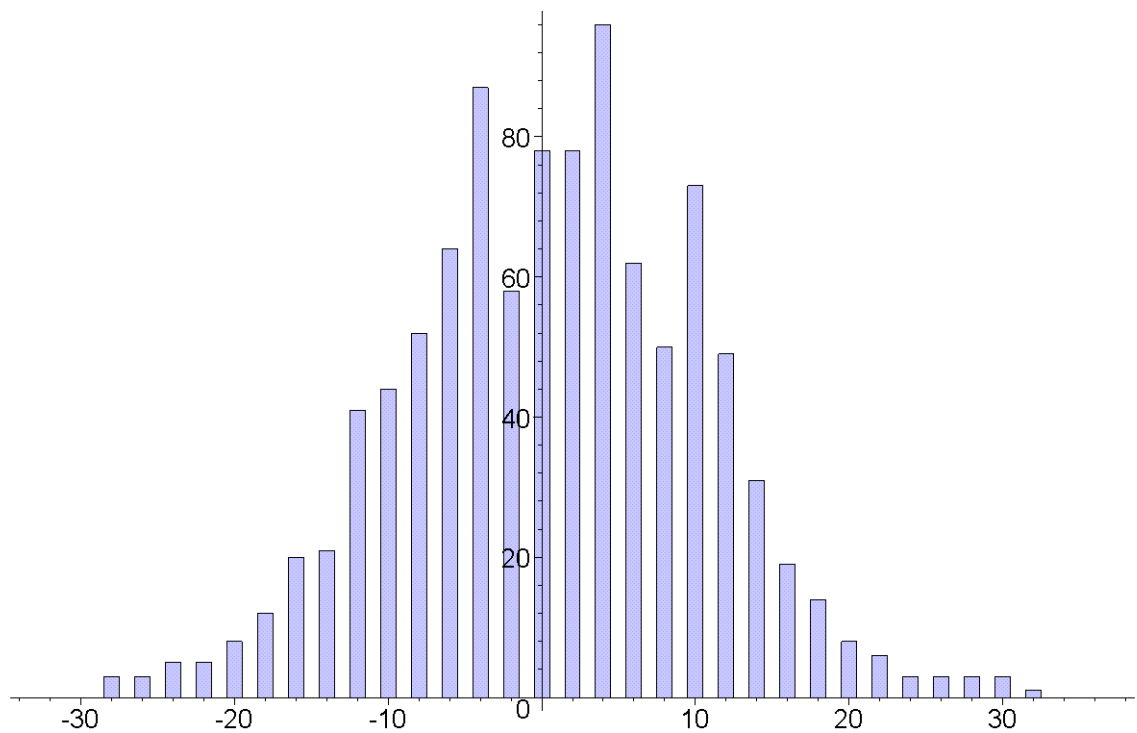
Il grafico della passeggiata aleatoria.

```
> passeggiata(Q);
```



Statistica sul valore finale dopo 100 passi (1000 esperimenti)

```
> statistica(1000, 100);
```



— Passeggiata aleatoria di lunghezza 1000

```

[ La passeggiata.
  > P := passi(1000) :
[ Le quote raggiunte.
  > Q := quota(P) :
[ La quota massima raggiunta.
  > quota_max(Q) ;
                                     14
[ La quota minima raggiunta.
  > quota_min(Q) ;
                                     -48
[ La quota finale raggiunta.
  > quota_finale(Q) ;
                                     -48
[ Numero di ritorni all'origine.
  > ritorni(Q) ;
                                     29
[ Numero di passaggi attraverso l'origine.
  > passaggi(Q) ;
                                     15
Il grafico della passeggiata aleatoria.
  > passeggiata(Q) ;

```

[
[
[
v

